



Test Schedule and Test Procedure (TSTP)

**ITSAR name: Session Management
Function (SMF) of 5G**

ITSAR No: ITSAR111092401

Disclaimer: This TSTP is intended to serve as a guide for testing various clauses outlined in the relevant ITSAR. The test methods described in this TSTP are not exhaustive and should not be considered as the sole approach to test any clause. The actual testing process and results may vary depending on how the OEMs implemented the clause in their equipment.

© रा.सं.सु.कें., २०२४

© NCCS, 2024

MTCTE के तहत जारी:

Issued under MTCTE by:

राष्ट्रीय संचार सुरक्षा केंद्र (रा.सं.सु.कें.)

दूरसंचार विभाग, संचार मंत्रालय

भारत सरकार

सिटी दूरभाष केंद्र, एसआर नगर, बैंगलोर-५६००२७, भारत

National Centre for Communication Security (NCCS)

Department of Telecommunications

Ministry of Communications

Government of India

City Telephone Exchange, SR Nagar, Bangalore-560027, India

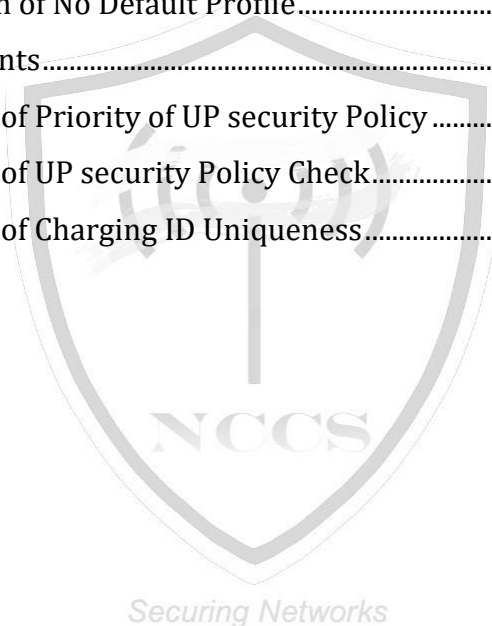
Table of Contents

Common Security Requirements	5
2.1.1 TSTP for Evaluation of Management Protocols Mutual Authentication	5
2.1.2 TSTP Report for Evaluation of Management Traffic Protection	12
2.1.3 TSTP For Role-based access control policy 5G ITSAR	20
2.1.4 TSTP for Evaluation of user Authentication – Local /Remote	26
2.1.5 TSTP For Remote login restrictions for privileged users	33
2.1.6 TSTP Report for Evaluation of Authorization Policy	39
2.1.7 TSTP for Evaluation of Unambiguous identification of the user & group accounts removal	44
2.2.1 TSTP for Evaluation of Authentication Policy	49
2.2.2 TSTP Report for Evaluation of Authentication Support – External	55
2.2.3 TSTP for Protection against brute force and dictionary attacks	61
2.2.4 TSTP Report for Evaluation for Enforcement of Strong Password under 5G ITSAR	78
2.2.5 TSTP For Inactive Session timeout under 5G ITSAR	83
2.2.6 TSTP for Evaluation of Password Changes	89
2.2.7 TSTP Report for Evaluation of Protected Authentication feedback (2.2.7 of CSR)	96
2.2.8 TSTP for Evaluation of Removal of predefined or default authentication attributes.	100
2.2.9 TSTP for Evaluation of Logout function	104
2.2.10 TSTP Report for Evaluation of Policy regarding consecutive failed login attempts.	109
2.3.1 TSTP Report for Evaluation of Secure Update	116
2.3.2 TSTP Report for Evaluation of Secure Upgrade	121
2.3.3 TSTP for evaluation of Source Code Security Assurance	126
2.3.4 TSTP Report for Known Malware and backdoor Check	130
2.3.5 TSTP For No Unused Software	136
2.3.6 TSTP for Evaluation of Unnecessary Services Removal	143
2.3.7 TSTP for Evaluation of Restricting System Boot Source	150
2.3.8 TSTP for Evaluation of Secure Time Synchronizations	155
2.3.9 TSTP Report for Evaluation of Restricted reachability of services	162
2.3.10 TSTP for Evaluation of Self Testing	167
2.4.1 TSTP For No Unused Functions	173

2.4.2 TSTP For No Unsupported Components	179
2.4.3 TSTP Report for Evaluation of Avoidance of Unspecified mode of Access	185
2.5.1 TSTP Report for Evaluation of Audit trail storage and protection.....	190
2.5.2 TSTP for Evaluation of Audit Event Generation	194
2.5.3 TSTP for Evaluation of Secure Log Export.....	205
2.5.4 TSTP for Evaluation of Logging access to personal data.....	211
2.6.1 TSTP for Cryptographic Based Secure Communication	216
2.6.2 TSTP for Cryptographic Module Security Assurance	223
2.6.3 TSTP for Cryptographic Algorithms implementation Security Assurance	226
2.6.4 TSTP For Protecting data and information – Confidential System Internal Data	229
2.6.5 TSTP for Evaluation of Protecting data and information in storage	235
2.6.6 TSTP Report for Evaluation of Protection against Copy of Data.....	243
2.6.7 TSTP for Protection against Data Exfiltration - Overt Channel	250
2.6.8 TSTP for Protection against Data Exfiltration - Covert Channel	255
2.7.1 TSTP for Traffic Filtering – Network Level Requirement.....	260
2.7.2 TSTP for Evaluation of Traffic Separation (2.7.2 of CSR)	270
2.7.3 TSTP Report for Traffic Protection –Anti-Spoofing	275
2.7.4 TSTP for Evaluation of GTP-C Filtering (2.7.4 of CSR)	282
2.7.5 TSTP for Evaluation of GTP-U Filtering.....	288
2.8.1 TSTP For Network Level and application-level DDoS under 5G ITSAR.....	294
2.8.2 TSTP For Excessive Overload Protection under 5G ITSAR	300
2.8.3 TSTP for Evaluation of Manipulated packets that are sent to an address of the network device shall not lead to an impairment of availability.....	306
2.9.1 TSTP for Evaluation of Fuzzing – Network and Application Level	320
2.9.2 TSTP for Evaluation of Port Scanning	329
2.9.3 TSTP Report for Evaluation of Vulnerability Testing Requirements	334
2.10.1 TSTP Report for Evaluation of Growing Content Handling.....	339
2.10.2 TSTP for Evaluation of Handling of ICMP.....	344
2.10.3 TSTP For Authenticated Privilege Escalation only	353
2.10.4 TSTP for Evaluation of System account identification	358
2.10.5 TSTP for Evaluation of OS Hardening - Minimized kernel network functions	362

2.10.6 TSTP for Evaluation of No automatic launch of removable media.....	372
2.10.7 TSTP Report for Evaluation of Protection from buffer overflows under 5G ITSAR .	379
2.10.8 TSTP for Evaluation of External file system mount restrictions.....	382
2.10.9 TSTP for Evaluation of File-system Authorization Privileges	387
2.10.10. TSTP for Syn Flood Prevention.....	393
2.10.11 TSTP for Handling of IP options and extensions.....	398
2.10.12 TSTP for Evaluation of Restriction on Running Scripts/Batch Processes	405
2.10.13 TSTP for Evaluation of Restrictions on Soft-Restart	411
2.11.1 TSTP for Evaluation of HTTPS.....	415
2.11.2 TSTP for Evaluation of Webserver logging.....	423
2.11.3 TSTP for Evaluation of HTTPS Input Validation.....	430
2.11.4 TSTP for No System Privileges	438
2.11.5 TSTP for No unused HTTPS methods	443
2.11.6 TSTP for No unused add-ons.....	450
2.11.7 TSTP for No compiler, interpreter, or shell via CGI or other server-side scripting..	455
2.11.8 TSTP for No CGI or other scripting for uploads	461
2.11.9 TSTP for No execution of system commands with SSI.....	467
2.11.10 TSTP for Access rights for web server configuration	473
2.11.11 TSTP for Evaluation of No default content.....	479
2.11.12 TSTP for Evaluation of No directory listings.....	485
2.11.13 TSTP for Evaluation of Web server information in HTTPS headers	492
2.11.14 TSTP for Evaluation of Web server information in error pages.....	498
2.11.15 TSTP for Minimized file type mappings	505
2.11.16 TSTP for Evaluation of Restricted file access	511
2.11.17 TSTP for Execute rights exclusive for CGI/Scripting directory.....	521
2.11.18 TSTP for HTTP User session	530
2.12.1 TSTP for Evaluation of No code execution or inclusion of external resources by JSON parsers.....	538
2.12.2 TSTP Report for Evaluation of Validation of the unique key values in IEs	544
2.12.3 TSTP Report for Validation of the IEs limits	548
2.12.4 TSTP Report for Evaluation of Protection at the transport layer	552

2.12.5 TSTP for Evaluation of Authorization token verification failure handling within one PLMN.....	559
2.12.6 TSTP for Evaluation of Authorization token verification failure handling in different PLMNs.....	565
2.13.1 TSTP For Remote Diagnostic Procedure - Verification under 5G ITSAR	570
2.13.2 TSTL Report for No System Password Recovery	576
2.13.3 TSTL Report for Secure System Software Revocation	581
2.13.4 TSTP Software Integrity Check –Installation.....	586
2.13.5 TSTP Software Integrity Check – Boot	591
2.13.6 TSTP Report for Evaluation of Unused Physical and Logical Interfaces Disabling...	595
2.13.7 TSTP for Evaluation of No Default Profile.....	600
Specific Security Requirements.....	605
3.1.1 TSTP for Evaluation of Priority of UP security Policy	605
3.2.1 TSTP for Evaluation of UP security Policy Check.....	610
3.3.1 TSTP for Evaluation of Charging ID Uniqueness.....	614



Common Security Requirements

2.1.1 TSTP for Evaluation of Management Protocols Mutual Authentication

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.1.1
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 1: Access and Authorization
2. **<Security Requirement No & Name >**2.1.1 Management Protocols Mutual Authentication
3. **<Requirement Description: >**

The network product management shall support mutual authentication mechanisms, the mutual authentication mechanism can rely on the protocol used for the interface itself or other means. Secure cryptographic controls prescribed in Table1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” shall only be used for SMF management and maintenance.

[Ref: TEC 25848:2022 /TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.4.1]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5. DUT Configuration:

Check OEM documentation to find all the supported Mutual Authentication Protocols.

Run NMAP or any port scanning tool to get running services and verify the version and conf. of the services:

Command used: `nmap -p- 172.18.0.8 -sC -sV` (Ip address here is that of DUT)

```

[ashwini@pc]~$ nmap -p- 127.0.0.1 -sV -sC
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-08 15:31 IST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00011s latency).
Not shown: 65531 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 9.3 (protocol 2.0)
| ssh-hostkey:
|_  256 55315abe056f1d086cb7c8c4a52a1a17 (ECDSA)
|_  256 340dd525014dfdae0c7ddc4f4b58bf1 (ED25519)
631/tcp   open  ipp          CUPS 2.4
| http-robots.txt: 1 disallowed entry
|_/
|_http-title: Home - CUPS 2.4.2
|_http-server-header: CUPS/2.4 IPP/2.1
7070/tcp  open  ssl/realserver?
|_ssl-date: TLS randomness does not represent time
|_ssl-cert: Subject: commonName=AnyDesk Client
| Not valid before: 2023-04-27T14:55:35
|_Not valid after: 2073-04-14T14:55:35
36886/tcp open  tcpwrapped

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 18.90 seconds

```

List of Example Management Protocols used as per (NMAP and OEM Documentation):

- SSH
- TLS
- SNMP
- And so on...

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30  AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

a. For SSH:

command used: **ssh -V** (To get version information)

```

amf@localhost $ ssh -V
OpenSSH_8.9p1 Ubuntu-3ubuntu0.1, OpenSSL 3.0.2 15 Mar 2022

```

Command used: **\$ cd ~/.ssh**

\$ ls

(Screenshot of the DUT consisting of public/private key pair generated by DUT. The public key of DUT is used for authenticating the DUT on the tester side)

```
amf@localhost $ ls
.rw-rw-r-- 106 ashwini 22 May 13:52 config
.rw----- 2.6k ashwini 20 Sep 2022 id_rsa
.rw-rw-r-- 577 ashwini 20 Sep 2022 id_rsa.pub
.rw----- 4.5k ashwini 19 May 15:47 known_hosts
```

Verify that the key pair used by the DUT is in accordance with the latest Document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)”

Command used: **\$ ssh-keygen -l -f ~/.ssh/id_rsa.pub**

```
amf@localhost $ ssh-keygen -l -f ~/.ssh/id_rsa.pub
3072 SHA256:5xY44sMqyViE0o3qXKvQ+KjZh4QP8Yt6PpkIwsFswbc amf@localhost (RSA)
```

(Screenshot of inside of known_hosts file that contains public key of tester used for authentication of tester)

```
1 github.com ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOMqknVzrm0SdG6U0oqKLsagH5C9okWi0dh2L9GKJL
2 github.com ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCj7ndNxQowgcQnjsHcLrqPEiiphnt+VTTvDP6mHBL9j1aNUkY4Ue1gvwnGLVl0h
3 github.com ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlkdHh0NTYAAAAIbmlkdHh0NTYAAABBBEmKSENjQEezOmxkZMy7opKgwF89n
4 10.200.14.121 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlkdHh0NTYAAAAIbmlkdHh0NTYAAABBBCHNV4Z5LicsbpWmp3pfpRkNCg
5 192.168.122.73 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIA1soJXmWSq4LZpMPatU2XUjrd/CVIOAtHQw+igB0nJ
6 192.168.122.73 ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGC7+6pV35fLnLy6rE5aUEKgfVVoY/J0tre+vvCHCZMkbgYWZ0nTHxapre67
7 192.168.122.73 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlkdHh0NTYAAAAIbmlkdHh0NTYAAABBB0QhWn0iou8VZd004E85MyiGh
8 192.168.126.209 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIAyQi/rGHWSelz2/RnL5UXaPBW0NjbraWPsJXTtEzFDV
```

Verify that Authentication Methods is set to "publickey,password" (2FA)

Command used: **\$ sudo cat /etc/ssh/sshd_config**

```
#PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

AuthenticationMethods "publickey,password"

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts yes
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes
```

b. Similarly, and so on for other protocols used.

6. Preconditions

- Documentation that lists each of the management protocols and describes the authentication mechanism used for each one.
- IP address of the DUT should be provided, either automatically detected and assigned by OEM, or assigned manually by Tester
- The DUT has the public key of the tester device pre-installed

7. Test Objective

To verify that the DUT permits access through supported management protocols only after successful mutual authentication.

8. **Test Plan**

8.1. **Number of Test Scenarios:**

8.1.1. Test Scenario for SSH:

- This test scenario is regarding SSH Management Protocol (Additional Test scenarios based on the OEM document)

8.2. **Test Bed Diagram**



8.3. **Tools Required:** Wireshark, OpenSSH, SNMP and so on..

8.4. **Test Execution Steps**

- Launch the Wireshark app on the tester device.
- Login into to the DUT and check the supported algorithms for mutual authentication
- Ensure DUT can be reached using ping packets
- Ensure that you can capture the ping request and reply packets between tester and DUT, with wireshark running on Tester Device.
- Use the Tester device to access the DUT over management protocol using right credentials. Verify through wireshark that Mutual Authentication is successfull. (Case 1)
- Use the Tester device to access the DUT over management protocol using wrong credentials. Verify through wireshark that Mutual Authentication is not successfull. (Case 2)
- Repeat the above steps for all the supported protocols.

9. **Expected Results for Pass:**

- **Case 1:** Tester and DUT are able to mutually authenticate each other successfully, so Tester is allowed to access DUT.
- **Case 2:** DUT is unable to mutually authenticate so should not allow tester to connect.

10. **Expected Format of Evidence:** Screenshots of Wireshark or Terminal

11. Test Execution:

Test Case Number: 01

a. **Test Case Name:** TC1_MUTUAL_AUTHENTICATION_FOR_SSH

b. **Test Case Description:** DUT should allow access to tester device through SSH only after successful mutual authentication.

c. **Execution Steps:**

- The tester tries to contact DUT using following commands

• ***ssh -o StrictHostKeyChecking=yes <hostname_DUT>@<DUT IP address>***

```
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-72-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

7 updates can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable

45 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '22.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu May 18 20:39:26 2023 from 192.168.110.73
```

- In order to test the 2FA of DUT, the tester must input incorrect password while logging in the DUT. (Note that the valid key pair is present in the DUT)

```
pratik@pratik-HP-Laptop-15g-br0xx:~$ ssh pratik@192.168.127.173
pratik@192.168.127.173's password:
Permission denied, please try again.
```

- In order to test server authentication, the tester must erase the known host information in the tester device which is present in .ssh/known_hosts directory and then try to connect to the server using following command

• ***ssh -o StrictHostKeyChecking=yes <hostname_DUT>@<DUT IP address>***

```
pratik@pratik-HP-Laptop-15g-br0xx:~$ ssh -o StrictHostKeyChecking=yes pratik@192.168.127.173
No ED25519 host key is known for 192.168.127.173 and you have requested strict checking.
Host key verification failed.
```

- The tester shall restore the known_host file and then in order to test client authentication, the tester must generate a new pair of keys which will remove the original key pairs using following command

• ***rm .ssh/id_rsa***

• ***rm .ssh/id_rsa.pub***

• ***ssh-add -D***

- And then generate new keys using

• ***ssh-keygen***

- The tester tries to contact DUT using following commands

• ***ssh -o StrictHostKeyChecking=yes <hostname_DUT>@<DUT IP address>***

```
pratik@pratik-HP-Laptop-15g-br0xx:~$ ssh pratik@192.168.127.173
pratik@192.168.127.173: Permission denied (publickey).
```

d. **Test Observations:**

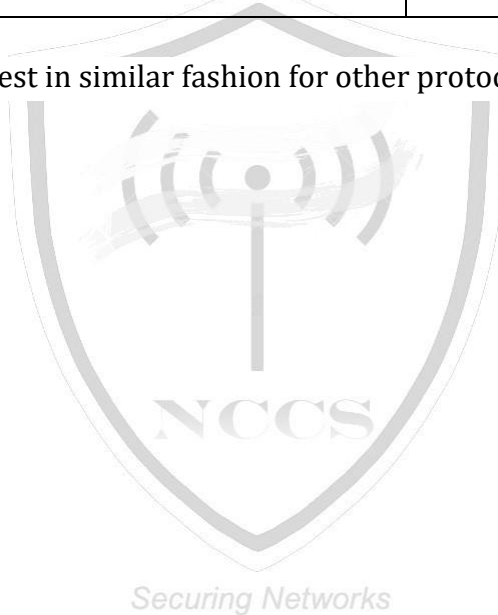
- **Case 1 (Positive Test Case):** Tester and DUT are able to mutually authenticate each other successfully, so Tester is allowed to access DUT.
- **Case 2 (Negative Test Case):** Tester and DUT are not able to communicate with each other if DUT authentication fails, Tester authentication fails or 2FA fails

e. **Evidence Provided** Screenshot of Terminal

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_MUTUAL_AUTHENTICATION_FOR_SSH		

(Tester has to execute the test in similar fashion for other protocols)



2.1.2 TSTP Report for Evaluation of Management Traffic Protection

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.1.2
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 1: Access and Authorization
2. **<Security Requirement No & Name >** 2.1.2 Management Traffic Protection
3. **<Requirement Description: >** SMF management traffic shall be protected strictly using secure cryptographic controls prescribed in Table1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.2.4]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5. DUT Configuration:

Check OEM documentation to find all the supported Mutual Authentication Protocols. Run NMAP or any port scanning tool to get running services and verify the version and conf. of the services:

Command used: `nmap -p- <IP Address of DUT> -sC -sV`

```

[ashwini@pc]~$ nmap -p- 127.0.0.1 -sV -sC
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-08 15:31 IST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00011s latency).
Not shown: 65531 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 9.3 (protocol 2.0)
| ssh-hostkey:
|_  256 55315abe056f1d086cb7c8c4a52a1a17 (ECDSA)
|_  256 340dd525014dfdae0c7ddc4f4b58bf1 (ED25519)
631/tcp   open  ipp          CUPS 2.4
| http-robots.txt: 1 disallowed entry
|_/
|_ http-title: Home - CUPS 2.4.2
|_ http-server-header: CUPS/2.4 IPP/2.1
7070/tcp  open  ssl/realserver?
|_ ssl-date: TLS randomness does not represent time
|_ ssl-cert: Subject: commonName=AnyDesk Client
| Not valid before: 2023-04-27T14:55:35
|_ Not valid after: 2073-04-14T14:55:35
36886/tcp open  tcpwrapped

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 18.90 seconds

```

List of Example Management Protocols used as per (NMAP and OEM

- Documentation):
- SSH
- TLS
- SNMP
- And so on...

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

a. For SSH:

command used: **ssh -V** (To get version information)

```

amf@localhost $ ssh -V
OpenSSH_8.9p1 Ubuntu-3ubuntu0.1, OpenSSL 3.0.2 15 Mar 2022

```

Command used: **\$ cd ~/.ssh**

\$ ls

(Screenshot of the DUT consisting of public/private key pair generated by DUT. The public key of DUT is used for authenticating the DUT on the tester side)

```
amf@localhost $ ls
.rw-rw-r-- 106 ashwini 22 May 13:52 ~ config
.rw----- 2.6k ashwini 20 Sep 2022 ~ id_rsa
.rw-r--r-- 577 ashwini 20 Sep 2022 ~ id_rsa.pub
.rw----- 4.5k ashwini 19 May 15:47 ~ known_hosts
```

(Screenshot of inside of known hosts file that contains public key of tester used for authentication of tester)

```
1 github.com ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOMqknkVzrm0SdG6U0oqKLsbgH5C9okWi0dh2L96KJL
2 github.com ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCj7ndNxQowgcQnjsHcLrqPEiiphnt+VTTvDP6mHBL9j1aNuK4Ue1gvwn6LVl0h
3 github.com ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlkdHh0NTYAAAAIbmlkdHh0NTYAAABBBEmKSENjQEezOmxkZMy7opKgwFB9n
4 10.200.14.121 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlkdHh0NTYAAAAIbmlkdHh0NTYAAABBBCHNV4Z5licsbpWmp3pfpRkNCg
5 192.168.122.73 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIA1soJXNWSq4LZpMPatU2XuJrD/CVIOAtHQw+16B0nJ
6 192.168.122.73 ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGC7+6pV3SfLnLy6rE5aUEKgfgvVoY/J0tre+vvCHCZMkbgYWZ0nTHxapre67
7 192.168.122.73 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlkdHh0NTYAAAAIbmlkdHh0NTYAAABBBQhWn0iou8VZd004E85MyiGh
8 192.168.126.209 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIAyQi/rGHWSeLz2/RnL5UXaPBW0NjbraWPsJXTtEzFDV
```

Command used: `cat /etc/ssh/sshd.config` (to get the configuration information)

	File: /etc/ssh/sshd_config
1	
2	# This is the sshd server system-wide configuration file. See
3	# sshd_config(5) for more information.
4	
5	# This sshd was compiled with PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr
6	
7	# The strategy used for options in the default sshd_config shipped with
8	# OpenSSH is to specify options with their default value where
9	# possible, but leave them commented. Uncommented options override the
10	# default value.
11	
12	Include /etc/ssh/sshd_config.d/*.conf
13	
14	#Port 8959
15	#AddressFamily any
16	#ListenAddress 0.0.0.0
17	#ListenAddress ::
18	
19	#HostKey /etc/ssh/ssh_host_rsa_key
20	#HostKey /etc/ssh/ssh_host_ecdsa_key
21	#HostKey /etc/ssh/ssh_host_ed25519_key
22	
23	# Ciphers and keying
24	#RekeyLimit default none
25	
26	# Logging
27	#SyslogFacility AUTH
28	#LogLevel INFO
29	

Verify that the ciphers supported by the DUT are in accordance with the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only. The configuration file (as Mentioned above) may have a field enlisting the ciphers the DUT supports


```

#ClientAliveCountMax 3
#UseDNS no
#PidFile /var/run/sshd.pid
#MaxStartups 10:30:100
#PermitTunnel no
#ChrootDirectory none
#VersionAddendum none

# no default banner path
#Banner none

# Allow client to pass locale environment variables
AcceptEnv LANG LC_*

# override default of no subsystems
Subsystem          sftp          /usr/lib/openssh/sftp-server

# Example of overriding settings on a per-user basis
#Match User anoncvs
#      X11Forwarding no
#      AllowTcpForwarding no
#      PermitTTY no
#      ForceCommand cvs server
Ciphers aes128-ctr,aes192-ctr,aes256-ctr,aes128-cbc
HostKeyAlgorithms ssh-ed25519

```

(Note: The configuration file may not have the Ciphers field or it may be commented which means that the configurations here follow default settings as mentioned here)

b. Similarly, and so on for other protocols used.

6. **Preconditions:**

- Network product documentation containing information about supported OAM protocol is provided by the vendor.
- A peer/tester device configured for communication with DUT over the protocol to be tested (e.g. SSH client supporting SSHv2 or HTTPS client) shall be available.
- Network product documentation stating which security protocols for protection of data in transit are implemented and which profiles in TS 33.310 and TS 33.210 are applicable is provided by the vendor for TLS, the tester shall base the tests on the profile defined by 3GPP in TS 33.310. For IKE and IPsec, the tester shall base the tests on the profile defined by 3GPP in TS 33.210. For protocols, for which 3GPP did not define a security profile, e.g. SSH, the tester shall base the tests on a widely recognized and publicly available security profile.

7. **Test Objective:-** To verify if the DUT management traffic shall be protected strictly using secure cryptographic controls prescribed in Table1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)”.

8. **Test Plan**

8.1. Number of Test Scenarios:

- Test Scenario for SSH:
- This test scenario is regarding SSH Management Protocol

8.2. TEST BED DIAGRAM:



8.3. **Tools Required:-** Wireshark, OpenSSH, SNMP and so on.

8.4. **Test Execution Steps:**

- Launch the Wireshark app on the tester device.
- Ensure DUT can be reached using ping packets
- Ensure that you can capture the ping request and reply packets between tester and DUT, with Wireshark running on Tester Device.
- Use the Tester device to access the DUT over management protocol using any of the encryption protocols mentioned in Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR). Verify through Wireshark that handshake of the selected management protocol is completed and data exchanged is protected using the algorithms mentioned in the handshake (Case 1)
- Use the Tester device to access the DUT over management protocol using any of the unsupported encryption protocols (3DES, DES and so on). Verify through Wireshark that handshake of the selected management protocol is not completed. (Case 2)
- Repeat the above steps for all the protocols.

Securing Networks

9. Expected Results for pass:

- **Case 1:** Handshake of the selected management protocol is completed between DUT and Tester device and data exchanged between DUT and tester device is encrypted using the selected encryption algorithm
- **Case 2:** Handshake of the selected management protocol is not complete between DUT and Tester device and communication between DUT and tester device fails

10. **Expected Format of Evidence:** Screenshots of Wireshark and Pcap files capturing the protocol handshake & data exchange between DUT and Tester device and the algorithm used for encryption

11. **Test Execution**

➤ **Test Case Number:** 01

- a. **Test Case Name:** TC1_PROTECT_DATA_INFO_TRANSFER_USING_SSH
- b. **Test Case Description:** DUT and tester device should support traffic protection through SSH only through provided cryptographic methods
- c. **Execution Steps:**
 - Tester tries to contact DUT using any of the supported encryption protocol (say aes256-ctr) using following command
 - `ssh -c aes256-ctr <DUT ADDRESS>`
 - Capture the communication through Wireshark.

4 0.028281293	10.10.28.200	192.168.127.173	93 SSHv2	Client: Protocol (SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.5)
6 0.068511727	192.168.127.173	10.10.28.200	93 SSHv2	Server: Protocol (SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.5)
8 0.069656465	10.10.28.200	192.168.127.173	1364 SSHv2	Client: Key Exchange Init
9 0.096169324	192.168.127.173	10.10.28.200	1108 SSHv2	Server: Key Exchange Init
12 0.142623431	10.10.28.200	192.168.127.173	100 SSHv2	Client: Diffie-Hellman Key Exchange Init
14 0.181858719	192.168.127.173	10.10.28.200	552 SSHv2	Server: Diffie-Hellman Key Exchange Reply, New Keys, Encrypte...
16 0.214137183	10.10.28.200	192.168.127.173	68 SSHv2	Client: New Keys
18 0.285240939	10.10.28.200	192.168.127.173	96 SSHv2	Client: Encrypted packet (len=44)
20 0.318982548	192.168.127.173	10.10.28.200	96 SSHv2	Server: Encrypted packet (len=44)
21 0.319199092	10.10.28.200	192.168.127.173	112 SSHv2	Client: Encrypted packet (len=60)
22 0.353367839	192.168.127.173	10.10.28.200	96 SSHv2	Server: Encrypted packet (len=44)
23 0.353608828	10.10.28.200	192.168.127.173	544 SSHv2	Client: Encrypted packet (len=492)
24 0.388569871	192.168.127.173	10.10.28.200	96 SSHv2	Server: Encrypted packet (len=44)

```

encryption_algorithms_client_to_server length: 10
encryption_algorithms_client_to_server string: aes256-ctr
encryption_algorithms_server_to_client length: 10
encryption_algorithms_server_to_client string: aes256-ctr
mac_algorithms_client_to_server length: 213
mac_algorithms_client_to_server string [truncated]: umac-64-etm@openssh.com,umac-128-etm@openssh.cc
mac_algorithms_server_to_client length: 213
mac_algorithms_server_to_client string [truncated]: umac-64-etm@openssh.com,umac-128-etm@openssh.cc
compression_algorithms_client_to_server length: 26
compression_algorithms_client_to_server string: none,zlib@openssh.com,zlib
compression_algorithms_server_to_client length: 26

```

- Tester tries to contact DUT using any of the unsupported encryption protocol (say 3des-cbc) using following command
 - `ssh -c 3des-cbc <DUT ADDRESS>`
- Capture the communication through Wireshark.

10.10.28.200	192.168.127.173	93 SSHv2	Client: Protocol (SSH-2.0-OpenSSH_8
192.168.127.173	10.10.28.200	93 SSHv2	Server: Protocol (SSH-2.0-OpenSSH_8
10.10.28.200	192.168.127.173	1364 SSHv2	Client: Key Exchange Init
192.168.127.173	10.10.28.200	1108 SSHv2	Server: Key Exchange Init

```

encryption_algorithms_client_to_server length: 8
encryption_algorithms_client_to_server string: 3des-cbc
encryption_algorithms_server_to_client length: 8
encryption_algorithms_server_to_client string: 3des-cbc
mac_algorithms_client_to_server length: 213
mac_algorithms_client_to_server string [truncated]: umac-64-etm@openssh.com,umac-128-et
mac_algorithms_server_to_client length: 213
mac_algorithms_server_to_client string [truncated]: umac-64-etm@openssh.com,umac-128-et
compression_algorithms_client_to_server length: 26
compression_algorithms_client_to_server string: none,zlib@openssh.com,zlib
compression_algorithms_server_to_client length: 26

```

d. **Test Observations**

- **Case 1:** Handshake of SSH is completed between DUT and Tester device and data exchange between DUT and tester device is encrypted using the selected encryption algorithm
- **Case 2:** Handshake of SSH is not complete between DUT and Tester device and communication between DUT and tester device fails

- e. **Evidence Provided:** Screenshot and the pcap file.

12. **Test Case Result**

SL. No	TEST CASE NAME	PASS/FAIL	REMARKS
1	TC1_PROTECT_DATA_INFO_TRANSFER_USING_SSH		

(Tester has to execute the test in similar fashion for other protocols)



2.1.3 TSTP For Role-based access control policy 5G ITSAR

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.1.3
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<TSTP Document ID:>

<Applicant Name:> Ex: XYZ

<Application Number>

<DUT Details: > Ex: Router

<DUT Software Version:>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 1 – Access and Authorization
2. **<Security Requirement No & Name >** 2.1.3 Role-based access control policy
3. **<Requirement Description: >** SMF shall support Role-Based Access Control (RBAC). A role-based access control system uses a set of controls that determines how users interact with domains and resources. The RBAC system controls how users or groups of users are allowed access to the various domains and what type of operation they can perform, i.e., the specific operation command or command group (e.g View, Modify, Execute). SMF supports RBAC with minimum of 3 user roles, in particular, for OAM privilege management for SMF Management and Maintenance, including authorization of the operation for configuration data and software via the network product console interface.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.6.2]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf

```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:** Documentation describing the role-based access control system including details on which user roles are defined.

7. **Test Objective/ Purpose:** Verify that users are granted access with role-based privileges.

8. **Test Plan:**

8.1 **Number of Test Scenarios:**

8.1.1 **Test Scenario for Role Based Access Control:**

This test scenario checks whether users can access the various files and can perform the type of operation they are allowed to.

8.2 **Test Bed Diagram:**



8.3 **Tools required:** Command Line Interface of the DUT.

8.4 **Test Execution Step:**

1. User accounts which are assigned to different access roles are created.
2. Operations that are allowed by different roles (as defined in the network product documentation), are attempted via the different user accounts.

9. **Expected Results:**

- Users that are assigned to a role that is not allowed to execute an operation are prevented from executing the operation.
- Users that are assigned to a role that is allowed to execute an operation can successfully execute the operation.

10. **Expected Format of Evidence:** Evidence suitable for the interface, e.g. screenshot containing the operational results.

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC1_ROLE_BASED_ACCESS_CONTROL_POLICY

b. Test Case Description: Test Needs to be conducted to ensure users are granted access with role-based privileges.

c. Execution Steps:

1. Tester needs to create accounts with varying privileges based on OEM Documentation.
2. The tester needs to check Users that are assigned to a role which is allowed to execute an operation can successfully execute the operation and users who do not have the permission cannot perform the operation.

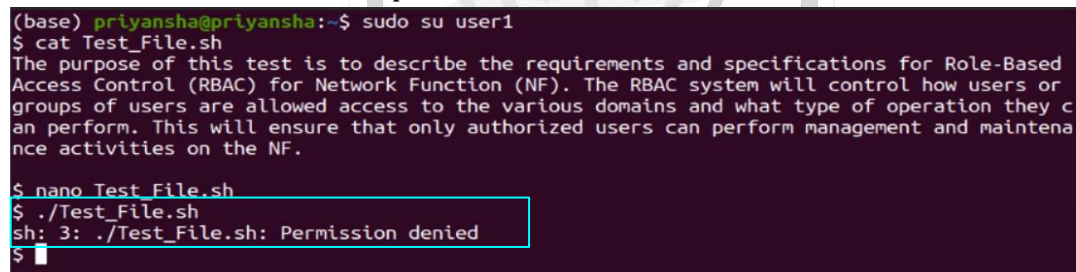
For Example,

We consider one file Test_File.sh and three users who belong to different groups.

- User 1 has View permission
- User 2 has Modify permission
- User 3 has Execute permission

➤ **Case 1(User1):** User 1 has view permission but does not have modify and execute permission. In the below screenshot we can see that when user1 performs the command `cat Test_File.sh`, it is able to view the content of the file.

When user1 performs, `./Test_File.sh` command to execute the script file, it gets Permission denied error as it does not have permission to execute.



```
(base) priyansha@priyansha:~$ sudo su user1
$ cat Test_File.sh
The purpose of this test is to describe the requirements and specifications for Role-Based
Access Control (RBAC) for Network Function (NF). The RBAC system will control how users or
groups of users are allowed access to the various domains and what type of operation they c
an perform. This will ensure that only authorized users can perform management and maintena
nce activities on the NF.

$ nano Test_File.sh
$ ./Test_File.sh
sh: 3: ./Test_File.sh: Permission denied
$
```

When User1 tries to modify it by writing “User1 Tryin to Modify....”, it will get the Error Permission Denied, as User1 doesn’t have permission to modify.

Securing Networks


```
GNU nano 4.8                                Test_File.sh                                Modified
The purpose of this test is to describe the requirements and specifications>
User1 Trying to Modify.....
[ Error writing Test_File.sh: Permission denied ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell
```

- **Case2(User2):** User 2 has permission to view and modify, but it does not have permission to execute. In the below screenshot we can see that User2 tries to modify the file by writing “-----Modified by User2-----”, this successfully gets modified. And when User2 views the file using cat command, the modification done by it can be seen.

When User2 performs, ./Test_File.sh command to execute the script file, it gets Permission denied error as it does not have permission to execute.

```
(base) priyansha@priyansha:~$ sudo su user2
$ nano Test_File.sh
$ cat Test_File.sh
-----Modified by User2-----
The purpose of this test is to describe the requirements and specifications for
Role-Based Access Control (RBAC) for Network Function (NF). The RBAC system will
control how users or groups of users are allowed access to the various domains
and what type of operation they can perform. This will ensure that only authoriz
ed users can perform management and maintenance activities on the NF.

$ ./Test_File.sh
sh: 3: ./Test_File.sh: Permission denied
$
```

- **Case3(User3):** User 3 has permission to view modify as well as execute the file. In the below screenshot we can see that User3 tries to modify the file by writing “-----Modified by User3-----”, this successfully gets modified. And when User3 views the file using cat command, the modification done by it can be seen.

When User3 performs, ./Test_File.sh command to execute the script file, it does not get Permission denied error as it has permission to execute.

The highlighted section shows that when User3 tries to execute the file, it gets details of syntax error etc which means it is successfully able to execute the file.

```

(base) priyansha@priyansha:~$ sudo su user3
$ nano Test_File.sh
$ cat Test_File.sh
-----Modified by User3-----
The purpose of this test is to describe the requirements and specifications for
Role-Based Access Control (RBAC) for Network Function (NF). The RBAC system will
control how users or groups of users are allowed access to the various domains
and what type of operation they can perform. This will ensure that only authoriz
ed users can perform management and maintenance activities on the NF.

$ ./Test_File.sh
./Test_File.sh: 1: -----Modified: not found
./Test_File.sh: 2: Syntax error: "(" unexpected
$

```

d. Test Observation:

- **Case 1:** Tester verifies users that are assigned to a role that is allowed to execute an operation can successfully execute the operation.
- **Case 2:** Tester verifies users that are assigned to a role that is not allowed to execute an operation are prevented from executing the operation.

e. Evidence Provided:

A testing report which will consist of the following information:

- Screenshot of the output of the terminal of the PC through which DUT logins.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_ROLE_BASED_ACCESS_CONTROL_POLICY		

2.1.4 TSTP for Evaluation of user Authentication – Local /Remote

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.1.4
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 1: Access and Authorization

2. **<Security Requirement No & Name >** 2.1.4 User Authentication – Local/Remote

3. **<Requirement Description: >** The various user and machine accounts on a system shall be protected from misuse. To this end, an authentication attribute is typically used, which, when combined with the username, enables unambiguous authentication and identification of the authorized user. Authentication attributes include

- Cryptographic keys
- Token
- Passwords

This means that authentication based on a parameter that can be spoofed is not permitted. Exceptions are attributes that cannot be faked or spoofed by an attacker. Minimum two of the above Authentication attributes shall be mandatorily combined for protecting all the accounts from misuse. An exception to this requirement is local access and machine accounts where at least one authentication attribute shall be supported.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.2.1]

Note: Local interface may not be applicable here for GVNP Models of Type 1& 2

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No

- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

Check OEM documentation to find all the supported remote connection protocol. Run NMAP or any port scanning tool to get running services and verify the version and conf. of the services:

Command used: **nmap -p- 172.18.0.8 -sC -sV** (Ip address here is that of DUT)

```

[ashwini@pc]~$
$ nmap -p- 127.0.0.1 -sV -sC
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-08 15:31 IST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00011s latency).
Not shown: 65531 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 9.3 (protocol 2.0)
| ssh-hostkey:
|_ 256 55315abe056f1d086cb7c8c4a52a1a17 (ECDSA)
|_ 256 340dd525014dfdae0c7ddc4f4b58bf1 (ED25519)
631/tcp   open  ipp          CUPS 2.4
| http-robots.txt: 1 disallowed entry
|_/
|_http-title: Home - CUPS 2.4.2
|_http-server-header: CUPS/2.4 IPP/2.1
7070/tcp  open  ssl/realserver?
|_ssl-date: TLS randomness does not represent time
|_ssl-cert: Subject: commonName=AnyDesk Client
| Not valid before: 2023-04-27T14:55:35
|_Not valid after: 2073-04-14T14:55:35
36886/tcp open  tcpwrapped

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 18.90 seconds

```

List of Example Management Protocols used as per (NMAP and OEM Documentation):

- SSH
- RDP
- And so on...

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edeaf1844429d30  AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

a. For SSH

command used: **ssh -V** (To get version information)

```

amf@localhost $ ssh -V
OpenSSH_8.9p1 Ubuntu-3ubuntu0.1, OpenSSL 3.0.2 15 Mar 2022

```

(Screenshot of inside of known_hosts file that contains public key of tester used for authentication of tester)

```

1 github.com ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAI0MqknVzrm0SdG6U0oqKLSabgH5C9okWi0dh2L9GKJL
2 github.com ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCj7ndNxQowgcQnjsHcLrQPEiiphnt+VTTvDP6mHBL9j1aNUKY4Ue1gvwn6LVl0h
3 github.com ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlkdHh0NTY1AAAAIbmlkdHh0NTY1AAABBBEmKSENjQEez0mxkZMy7opKgwFB9n
4 10.200.14.121 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlkdHh0NTY1AAAAIbmlkdHh0NTY1AAABBBCHNV4Z5LicsbpWmp3pfpRkNCg
5 192.168.122.73 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIA1soJXHWsq4LzPmPatU2XuJrD/CVIOAtHQW+i6BDnJ
6 192.168.122.73 ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGC7+6pV35fLnLy6rE5aUEKfgvVoY/J0tre+vvCHCZMkbgYWZ0nThxapre67
7 192.168.122.73 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlkdHh0NTY1AAAAIbmlkdHh0NTY1AAABBB0QhWn0iou8VZd004E85MyiGh
8 192.168.126.209 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIAyqi/rGHWSeLz2/RnL5UXaPBW0NjbraWPSJXTtEzFDV

```

Command used: **cat /etc/ssh/sshd.config** (to get the configuration information) Verify that AuthenticationMethods is set to "publickey,password" (2FA)

```
#PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

AuthenticationMethods "publickey,password"

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts yes
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes
```

b. **Similarly check for other user databases for which user authentication is needed for Local Access/ Machine Accounts**

command used: **sudo cat /etc/shadow** (To get content of shadow file)

```
ftp*:19039:0:99999:7:::
debian-tor*:19460:0:99999:7:::
sshd*:19545:0:99999:7:::
iith:$6$ahNC0TfJb5xtZac4$DH80o90cOq3JsdEIQvNcAdtNZTfAxG/mrIRXzenJxL17HkU0aSnBbnNViVW9gcIQgsZARdhJra61v0PyVBqHr.:19545:0:99999:7:::
machine_acc*:19545:0:99999:7:::
```

Verify that field after the username is not empty. It should have a hash value (for user accounts) or * or x (conventionally for machine accounts) (The mentioned configuration of machine accounts is for standard machine accounts of various services in linux. The configuration of machine accounts may vary according to the system and context.)

6. **Precondition:**

- All predefined accounts are identified in the documentation accompanying the Network Product.
- Instructions of how to create new accounts are provided in the documentation accompanying the Network Product.
- Instructions of how administrator user can view all existing accounts in the database are provided in the documentation accompanying the Network Product.
 - **NOTE:** No test is provided here for finding undocumented hard coded accounts as such tests may be impossible to define in a general way.

7. **Test Objective:** To verify that minimum two of the above authentication attributes shall be mandatorily combined for protecting all the accounts from misuse and for local access and machine accounts, at least one authentication attribute shall be supported

8. **Test Plan:**

8.1. **Number of Test Scenarios:**

8.1.1. Test Scenario for SSH

- This test scenario is regarding SSH Protocol (Additional Test scenarios based on the OEM document)

8.1.2. Test Scenario for Local Access

- This test scenario is regarding local access of user accounts

8.2. Test Setup Diagram



8.3. Tools Required NULL

8.4. Test Execution Steps

- Use the Tester device to access the DUT remotely using right credentials. (Case 1)
- Use the Tester device to access the DUT remotely using wrong credentials.(Case 2)
- Tester tries to access DUT locally using terminal access with correct credentials (Case 3)
- Tester tries to access DUT locally using terminal access with incorrect credentials (Case 4)
- Tester tries to access machine accounts in DUT using terminal access with correct credentials (Case 5)
- Tester tries to access machine accounts in DUT using terminal access with incorrect credentials (Case 6)

9. Expected Results for Pass:

- **Case 1:** DUT is able to authenticate tester successfully, so Tester is allowed to access DUT.
- **Case 2:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access DUT.
- **Case 3:** DUT is able to authenticate tester successfully, so Tester is allowed to access DUT.
- **Case 4:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access DUT.
- **Case 5:** DUT is able to authenticate tester successfully, so Tester is allowed to access DUT.
- **Case 6:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access DUT.

10. Expected Format of Evidence:- Screenshot of Terminal

11. Test Execution:

➤ **Test Case Number: 1**

a. **Test Case Name:** TC_ACCOUNT_PROTECTION_PWD_KEY_1

b. **Test Case Description:** Tester should be able to login in DUT if and only if correct password and cryptographic keys are used

c. **Execution Steps:**

- The tester tries to contact DUT using following commands (Case 1)

• **ssh <hostname_DUT>@<DUT IP address>**

```
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-72-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

7 updates can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable

45 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '22.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu May 18 20:39:26 2023 from 192.168.110.73
```

- In order to test the 2FA of DUT, the tester must input incorrect password while logging in the DUT. (Note that the valid key pair is present in the DUT (Case 2))

```
Permission denied, please try again.
```

- The tester must generate a new pair of keys which will remove the original key pairs using following command (Case 2)

• **rm .ssh/id_rsa**

• **rm .ssh/id_rsa.pub**

• **ssh-add -D**

- And then generate new keys using

• **ssh-keygen**

- The tester tries to contact DUT using following commands

• **ssh <hostname_DUT>@<DUT IP address>**

```
Permission denied (publickey).
```

d. **Test Observations:**

➤ **Case 1:** DUT is able to authenticate tester successfully, so Tester is allowed to access DUT.

➤ **Case 2:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access DUT.

e. **Evidence Provided:** Screenshot of Terminal

➤ **Test Case Number: 2**

- a. **Test Case Name:** TC_ACCOUNT_PROTECTION_PWD_KEY_2
- b. **Test Case Description:** Tester should be able to login in DUT if and only if correct password is used
- c. **Execution Steps:**
 - The tester tries to login into DUT using incorrect credentials (Case 4)
 - The tester tries to login into DUT using correct credentials (Case 3)
- d. **Test Observations:**
 - **Case 3:** DUT is able to authenticate tester successfully, so Tester is allowed to access DUT.
 - **Case 4:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access DUT.
- e. **Evidence Provided:** Screenshot of Terminal
 - **Test Case Number:** 3
 - a. **Test Case Name:** TC_ACCOUNT_PROTECTION_PWD_KEY_3
 - b. **Test Case Description:** Tester should be able to login in DUT machine accounts if and only if correct password is used
 - c. **Execution Steps:**
 - The tester tries to login into DUT machine accounts using incorrect credentials (Case 5)
 - The tester tries to login into DUT machine accounts using correct credentials (Case 6)
 - d. **Test Observations:**
 - **Case 5:** DUT is able to authenticate tester successfully, so Tester is allowed to access DUT machine accounts.
 - **Case 6:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access DUT machine accounts.
- e. **Evidence Provided:** Screenshot of Terminal

12. Test Case Result:

Securing Networks

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_ACCOUNT_PROTECTION_PWD_KEY_1		
2	TC_ACCOUNT_PROTECTION_PWD_KEY_2		
3	TC_ACCOUNT_PROTECTION_PWD_KEY_3		

(Tester has to execute the test in similar fashion for other databases)

2.1.5 TSTP For Remote login restrictions for privileged users

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.1.5
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 1 – Access and Authorization
2. **<Security Requirement No & Name >** 2.1.5 Remote login restrictions for privileged users
3. **<Requirement Description: >** Login to Network Product as root or equivalent highest privileged user shall be limited to the system console only. Root users will not be allowed to login to Network Product remotely. This remote root user access restriction is also applicable to application software's / tools such as TeamViewer, desktop sharing which provide remote access to the Network Product.

[Reference TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.3.2.6]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

For SSH:

command used: **ssh -V** (To get version information)

```
amf@localhost $ ssh -V
OpenSSH_8.9p1 Ubuntu-3ubuntu0.1, OpenSSL 3.0.2 15 Mar 2022
```

Locate the line that starts with **PermitRootLogin** in the SSH server configuration file. By default, it is set to yes, allowing root login. Make sure this field is set to No.

Ensure **DenyGroups** is set to the name of a high privilege group, for example: sudo(root privilege group) which will make sure that SSH login for all users belonging to the sudo group

```
Include /etc/ssh/sshd_config.d/*.conf

#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
DenyGroups sudo
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

is denied.

Similarly, and so on for other protocols used.

6. **Preconditions:** A document that describes the interfaces to the network product and how the tester can login to them remotely.

7. **Test Objective/ Purpose:** Verify that the root or equivalent highest privileged user will not be allowed to login to the system remotely.

8. **Test Plan:**

8.1 **Number of Test Scenarios:**

8.1.1 **Test Scenario for SSH:**

This test scenario is regarding SSH Management Protocol (Additional Test scenarios based on the OEM document)

8.2 **TestBed Diagram:**



8.3 **Tools required:**

Command Line Interface of the DUT, ssh, snmp or other remote access protocols.

8.4 **Test Execution Step:**

1. The tester tries to remotely login to the network product using the credentials of the root or equivalent highest privileged user via the interfaces as described in the documentation.
2. The tester tries to login to the network product using the credentials of the root or equivalent highest privileged user from the physical console of the system.

9. **Expected Results:**

- The tester is not able to login to the system remotely using the root credentials.
- The tester is able to login to the system from the physical console using the root credentials.

10. **Expected Format of Evidence:** Evidence suitable for the interface, e.g. screenshot containing the operational results.

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC1_REMOTE_LOGIN_RESTRICTIONS_PRIVILEGED_USERS

b. Test Case Description: Test Needs to be conducted that any privileged or root user should not be able to login over remote medium.

c. Execution Steps:

- Remote Login Test

Attempt to remotely log in to the network product using the root or equivalent highest privileged user's credentials. Use the appropriate command based on the remote login method described in the documentation.

Note: There can be many ways through which remote access can be performed like ssh, snmp, telnet, VNC (Virtual Network Computing), RDP (Remote Desktop Protocol) etc. Below steps are for ssh.

In case of ssh, ***ssh root@<network_product_ip>***

If the login is successful, record a failure. If the login is unsuccessful and you receive an authentication error or connection refused.

- Remote Login Test for Application such as Team Viewer

Determine which specific applications or tools are installed on your system that enable desktop sharing and remote access. Common examples include TeamViewer, VNC (Virtual Network Computing), AnyDesk, Chrome Remote Desktop, etc.

These applications have the option to disable remote access or desktop sharing, access its settings and disable the feature. This will prevent remote connections from being established.

- Physical Console Login Test

The tester attempts to log in to the network product using the credentials of the root or equivalent highest privileged user from the physical console of the system. Move to the physical console of the Linux system. Log in to the system using the root or equivalent highest privileged user's credentials. Enter the following command at the console prompt: **su-** Then provide the root user's password.

If the login is successful, record a pass.

If the login is unsuccessful and you receive an authentication error, record a failure.

d. Test Observation:

➤ **Case 1:** Root/Privileged User gets the access to the DUT on remote (**Negative Testcase**)

The below figure is from the PC through which the root/privileged user tries to login to the DUT. After entering the password, the user gets access to the DUT.

```

amf@localhost $ ssh root@192.168.126.132
root@192.168.126.132's password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-46-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

99 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

14 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

*** System restart required ***
Last login: Sat Jul 29 01:03:20 2023 from 192.168.126.157

```

- **Case 2: Root/Privileged User does not get the access to the DUT on remote (Positive Testcase)** The Below figure shows that the root user tries to connect remotely. Then a prompt to enter password appears after successfully establishing the connection. Then the permission denied message is popped up on the command line of the PC through which DUT is accessed remotely.

```

amf@localhost $ ssh root@192.168.126.132
root@192.168.126.132's password:
Permission denied, please try again.

```

e. Evidence Provided:

A testing report which will consist of the following information:

- Screenshot of the output of the terminal of the PC through which DUT logs in.

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_REMOTE_LOGIN_RESTRICTIONS_PRIVILEGED_USERS		

2.1.6 TSTP Report for Evaluation of Authorization Policy

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.1.6
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 1 - Access & Authorization
2. **<Security Requirement No & Name >** 2.1.6 Authorization policy
3. **<Requirement Description: >**

The authorizations for accounts and applications shall be reduced to the minimum required for the tasks they have to perform. Authorizations to a system shall be restricted to a level in which a user can only access data and use functions that he needs in the course of his work. Suitable authorizations shall also be assigned for access to files that are components of the operating system or of applications or that are generated by the same (e.g. configuration and logging files). Alongside access to data, execution of applications and components shall also take place with rights that are as low as possible. Applications should not be executed with administrator or system rights.

[Reference: TS/DSI STD T1.3GPP 33.117 -16.7.0 V.1.0.0. Section 4.2.3.4.6.1]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```


6. **Preconditions:**

- The tester shall be provided access to the administrative account as well as other user accounts (for verifying access rights).
- The Vendor/OEM must provide Documentation describing the steps to set the authorization policy for a particular user.

7. **Test Objective:** Verify that authorization policy is in place and that data access and application execution in the system are according to the authorization policy.

8. **Test Plan:**

a. **Test Setup Diagram:**



b. **Tools Used:** NULL

c. **Test Execution Steps:**

1. The tester must login into an account which has been given some specific privileges (RBAC as defined in Vendor documentation).
2. The tester must perform the permitted operations on the respective file/directories. (Case 1)
3. Then the tester must perform the non-permitted/denied operations on the respective file/directories. (Case 2)
4. The tester must try to run an application with administrator/system rights (Case 3)

9. **Expected Result for Pass:** The tester can perform operations on the data files/directories. (Case 1)

- The tester is not allowed to perform operations on the data files/directories. (Case 2)
- The application is not allowed to run with administrator/system rights. (Case 3)

10. **Expected Format of Evidence:** Screenshot of terminal/GUI containing the outcome of the performed operations on the data & application executables.

11. **Test Execution:**

➤ **Test Case Number: 1**

a. **Test Case Name:** TC_VERIFY_AUTH_POLICY

b. Test Case Description: Verify that the authorization policy is in place and that data access and application execution in the system are according to the authorization policy.

c. Execution Steps:

NOTE: The commands in the following description are considering Ubuntu 20.04 as the Operating System. The commands may vary for different Operating Systems.

1. The tester must login into an account which has been given some specific privileges (RBAC as defined in Vendor documentation). Here, we consider a case wherein we have an account named *siddhesh*.

The account has privileges set such that it is only allowed to read a file (say *test2.txt*) and is denied accessing a directory (say *test_dir*).

```
root@stark99:/home/siddhesh# ls -lrt test2.txt
-r--rw-r-- 1 siddhesh siddhesh 19 May 31 2022 test2.txt
root@stark99:/home/siddhesh#
root@stark99:/home/siddhesh# ls -lrt | grep test_dir
d-wxrw-x 2 siddhesh siddhesh 4096 Jul 23 22:15 test_dir
root@stark99:/home/siddhesh#
```

2. The tester must perform the permitted operations on the respective file/directories. (Case 1) Here, user *siddhesh* is allowed to read the file *test2.txt*

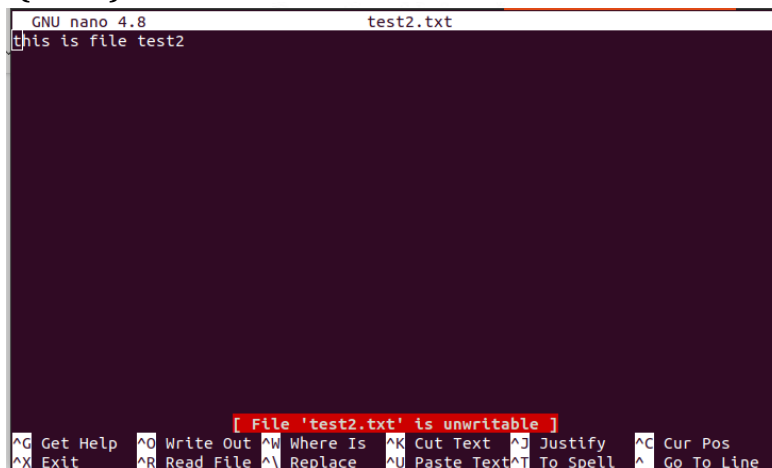
Read allowed:

```
siddhesh@stark99:~$ cat test2.txt
this is file test2
siddhesh@stark99:~$
```

3. Then the tester must perform the non-permitted/denied operations on the respective file/directories. (Case 2)

Here, user *siddhesh* is NOT allowed to modify/write into the file *test2.txt* and read contents of *test_dir* as we set the access rights accordingly

Modification(write) Denied:



The screenshot shows the GNU nano 4.8 text editor interface. The title bar indicates the file being edited is 'test2.txt'. The content of the file is 'this is file test2'. At the bottom of the editor, a red error message is displayed: '[File 'test2.txt' is unwritable]'. Below the error message, the standard nano editor shortcuts are listed: ^G Get Help, ^O Write Out, ^W Where Is, ^K Cut Text, ^J Justify, ^C Cur Pos, ^X Exit, ^R Read File, ^\ Replace, ^U Paste Text, ^T To Spell, and ^_ Go To Line.

Reading contents of directory *test_dir* denied:

```
siddhesh@stark99:~$ cd test_dir/
siddhesh@stark99:~/test_dir$ ls
ls: cannot open directory '.': Permission denied
siddhesh@stark99:~/test_dir$
```

4. The tester must try to run an application with administrator/system rights (Case 3)
Here the application *test_app.py* is not allowed to run with *sudo*(root/administrator) rights.

```
$ sudo test_app.py
[sudo] password for tom:
tom is not in the sudoers file. This incident will be reported.
$
```

d. Test Observations:

- (Case 1) Tester verifies that the permitted operations on the particular file/directories are allowed.
- (Case- 2) Tester verifies that the non-permitted/denied operations on the particular file/directories are denied.
- (Case- 3) Tester verifies that the application is not allowed to run with administrator/system rights.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_VERIFY_AUTH_POLICY		

NCCS
Securing Networks

2.1.7 TSTP for Evaluation of Unambiguous identification of the user & group accounts removal

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.1.7
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name:>** Section 1: Access and Authorization
2. **<Security Requirement No & Name:>** 2.1.7 Unambiguous identification of the user & group accounts removal
3. **<Requirement Description:>** Users shall be identified unambiguously by the SMF. SMF shall support the assignment of individual accounts per user, where a user could be a person, or, for Machine Accounts, an application, or a system. SMF shall not enable the use of group accounts or group credentials or sharing of the same account between several users.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.1.2]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. **DUT Configuration:** Configuration needed for setting one login at a time. This is done on Linux for the user “iith”. Using the command “*sudo gedit /etc/security/limits.conf*” we can check is this is set or not.

Note: this is done for Linux distribution and will vary for different distributions. It should be provided in the documentation how the OEM has done this and how we can check this functionality.

```

limits.conf
/etc/security

19 # - "soft" for enforcing the soft limits
20 # - "hard" for enforcing hard limits
21 #
22 #<item> can be one of the following:
23 # - core - limits the core file size (KB)
24 # - data - max data size (KB)
25 # - fsize - maximum filesize (KB)
26 # - memlock - max locked-in-memory address space (KB)
27 # - nofile - max number of open file descriptors
28 # - rss - max resident set size (KB)
29 # - stack - max stack size (KB)
30 # - cpu - max CPU time (MIN)
31 # - nproc - max number of processes
32 # - as - address space limit (KB)
33 # - maxlogins - max number of logins for this user
34 # - maxsyslogins - max number of logins on the system
35 # - priority - the priority to run user process with
36 # - locks - max number of file locks the user can hold
37 # - sigpending - max number of pending signals
38 # - msgqueue - max memory used by POSIX message queues (bytes)
39 # - nice - max nice priority allowed to raise to values: [-20, 19]
40 # - rtprio - max realtime priority
41 # - chroot - change root to directory (Debian-specific)
42 #
43 #<domain> <type> <item> <value>
44 iith hard maxlogins 1
45 # soft core 0

```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2dea1d08af1844429d30 AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

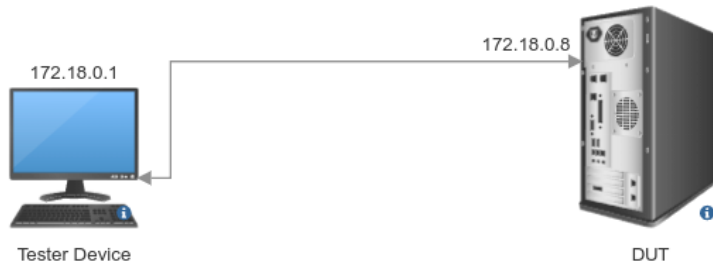
6. Preconditions:

- ⌘ All user and group data bases for names and credentials supported by the network product are identified in the documentation accompanying the network product.
- ⌘ The OEM document should provide the details of the password policy used.
- ⌘ All predefined accounts and groups are identified in the documentation accompanying the Network Product.
- ⌘ Instructions of how administrator users can add accounts, groups, and credentials to the database(s) are provided in the documentation accompanying the Network Product.
- ⌘ The operations manual describes OAM user and group concepts supported by the network product.
- ⌘ The tester should have sudo access.

7. Test Objective: To ensure that all accounts are uniquely identifiable, and no group account is allowed.

8. Test Plan:

8.1 Test Setup Diagram:



8.2 Tools Used: Command line

8.3 Test Execution Steps

- All the preconditions should be met.
- To check if all the accounts are unambiguously identified, we can check using this command: -
awk -F: 'seen[\$3]++ {print "Duplicate UID:", \$3}' /etc/passwd

9. Expected results for Pass:

- Preconditions should meet.
- All the user accounts should be uniquely identifiable.
- Simultaneous login should not be allowed.

Note1: All the databases should be checked for each of these requirements. The details of these databases should be present in the OEM documentation.

10. Expected Format of Evidence: Screenshots showing there are no duplicate accounts present and there is no group account.

11. Test Execution:

➤ Test Case Number: 01

a. Test Case Name:

TC_UNAMBIGUOUS_IDENTIFICATION_OF_THE_USER_AND_GROUP_ACCOUNT_REMOVAL

b. Test Case Description:

To ensure that the accounts are identified unambiguously.

c. Execution Steps:

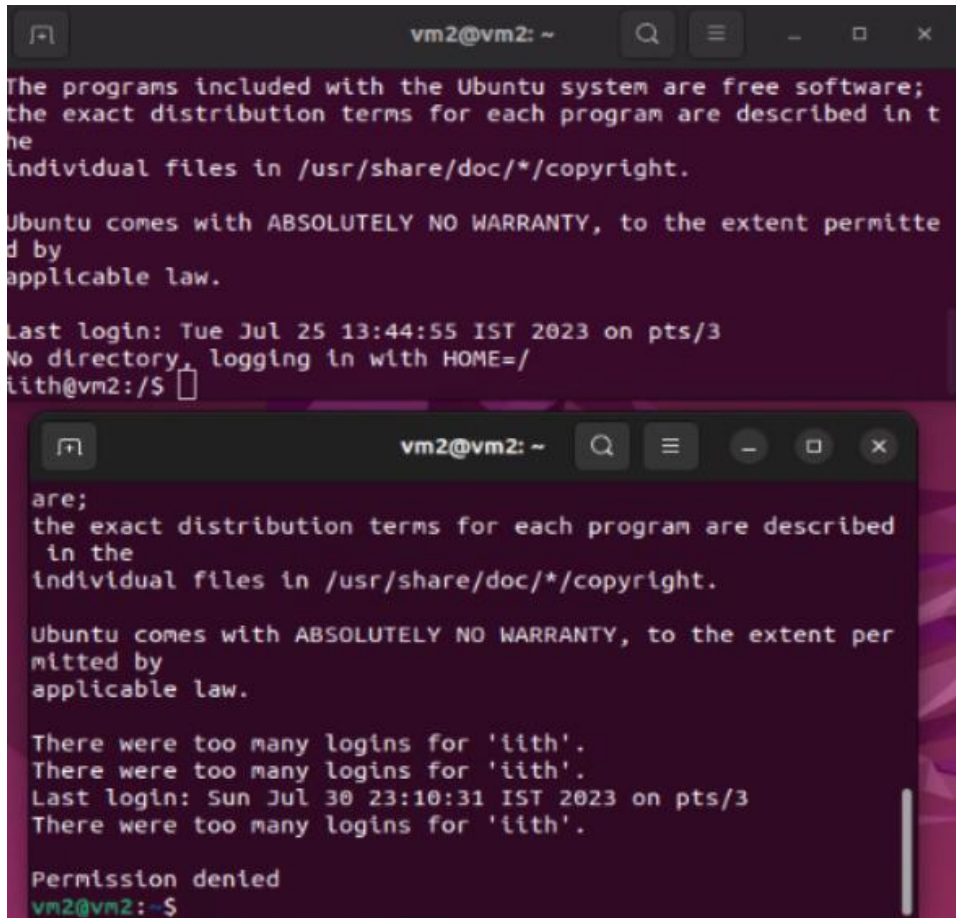
Success: Unambiguous identification of account

```
osboxes@osboxes:~$ awk -F: 'seen[$3]++ {print "Duplicate UID:", $3}' /etc/passwd
osboxes@osboxes:~$
```


Failure

```
osboxes@osboxes:~$ awk -F: 'seen[$3]++ {print "Duplicate UID:", $3}' /etc/passwd
Duplicate UID: 1001
osboxes@osboxes:~$
```

Screenshots showing multiple attempts to login by same user is restricted.



```
vm2@vm2: ~
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in t
he
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitte
d by
applicable law.

Last login: Tue Jul 25 13:44:55 IST 2023 on pts/3
No directory, logging in with HOME=/
iith@vm2:/S$

vm2@vm2: ~
are;
the exact distribution terms for each program are described
in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent per
mitted by
applicable law.

There were too many logins for 'iith'.
There were too many logins for 'iith'.
Last login: Sun Jul 30 23:10:31 IST 2023 on pts/3
There were too many logins for 'iith'.

Permission denied
vm2@vm2:~$
```

d. **Test Observations:** It should be ensured that the users are identified uniquely.

12. Test Case Result:

SL. No	OUTCOME OF RUNNING THE SCRIPT (CASE 1/CASE2)	PASS/FAIL	Remarks
1	TC_UNAMBIGUOUS_IDENTIFICATION_OF_TH E_USER_AND_GROUP_ACCOUNT_REMOVAL		

2.2.1 TSTP for Evaluation of Authentication Policy

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.2.1
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 2: Authentication Attribute Management

2. **<Security Requirement No & Name >** 2.2.1 Authentication Policy

3. **<Requirement Description: >**

The usage of a system function without successful authentication on basis of the user identity and at least two authentication attributes (e.g. password, certificate) shall be prevented. For machine accounts and local access one authentication attribute will be sufficient. System functions comprise, for example network services (like SSH, SFTP, Web services), local access via a management console, local usage of operating system and applications. This requirement shall also be applied to accounts that are only used for communication between systems.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.1.1]

Note: The reference to 'Local accesses and 'Console' may not be applicable here for GVNP Models of Type 1& 2

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5. **DUT Configuration:** Check OEM documentation to find all the supported remote connection protocol. Run NMAP or any port scanning tool to get running services and verify the version and conf. of the services:

Command used: `nmap -p- 172.18.0.8 -sC -sV` (Ip address here is that of DUT)

```

[ashwini@pc]~$ nmap -p- 127.0.0.1 -sV -sC
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-08 15:31 IST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00011s latency).
Not shown: 65531 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 9.3 (protocol 2.0)
| ssh-hostkey:
|_  256 55315abe056f1d086cb7c8c4a52a1a17 (ECDSA)
|_  256 340dd525014dfdae0c7ddc4f4b58bf1 (ED25519)
631/tcp   open  ipp          CUPS 2.4
| http-robots.txt: 1 disallowed entry
|_/
|_http-title: Home - CUPS 2.4.2
|_http-server-header: CUPS/2.4 IPP/2.1
7070/tcp  open  ssl/realserver?
|_ssl-date: TLS randomness does not represent time
|_ssl-cert: Subject: commonName=AnyDesk Client
| Not valid before: 2023-04-27T14:55:35
|_Not valid after: 2073-04-14T14:55:35
36886/tcp open  tcpwrapped

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 18.90 seconds

```

List of Example Management Protocols used as per (NMAP and OEM Documentation):

- SSH
- RDP
- And so on...

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

6. Precondition:

- The manufacturer shall supply the list of system functions which include network services, local access via a management console, local usage of operating system and applications.
- The manufacturer shall supply the list of access entries for system functions.

7. Test Objective: To ensure that system functions shall not be used without successful authentication and authorization.

8. Test Plan:

8.1. Number of Test Scenarios:

8.1.1. Test Scenario for System Functions

- This test scenario is access of system functions with at least 2 authentication attributes.

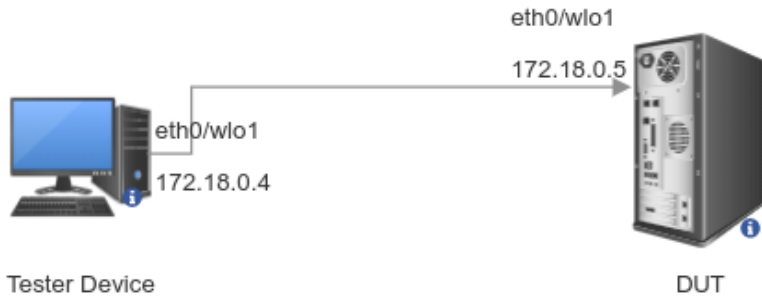
8.1.2. Test Scenario for Local Access

- This test scenario is regarding local access of user accounts

8.1.3. Test Scenario for Machine Accounts

- This test scenario is regarding machine accounts

8.2. Test Setup Diagram



8.3. Tools Required :- NULL

8.4. Test Execution Steps

- Use the Tester device to access the system functions in DUT using right credentials. (Case 1)
- Use the Tester device to access the system functions in DUT using wrong credentials. (Case 2)
- Tester tries to access system functions in DUT locally using terminal access with correct credentials (Case 3)
- Tester tries to access system functions in DUT locally using terminal access with incorrect credentials (Case 4)
- Tester tries to access system functions in DUT locally with machine accounts with correct credentials (Case 5)
- Tester tries to access system functions in DUT locally with machine accounts with incorrect credentials (Case 6)

9. Expected Results for Pass:

- **Case 1:** DUT is able to authenticate tester successfully, so Tester is allowed to access system functions in DUT.
- **Case 2:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access system functions in DUT.
- **Case 3:** DUT is able to authenticate tester successfully, so Tester is allowed to access system functions in DUT.
- **Case 4:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access system functions in DUT.
- **Case 5:** DUT is able to authenticate tester successfully, so Tester is allowed to access system functions in DUT.

- **Case 6:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access system functions in DUT.

10. **Expected Format of Evidence:-** Success message of login as per provided documentation

11. **Test Execution:**

- **Test Case Number: 1**

- a. **Test Case Name:** TC_SYS_FUN_USAGE_1
- b. **Test Case Description:** Tester should be able to access system functions only after getting authenticated
- c. **Execution Steps:**
 - The tester follows the instructions in OEM document to access the provided system functions using correct credentials (Case 1)
 - The tester follows the instructions in OEM document to access the provided system functions using incorrect credentials (Case 2)
- d. **Test Observations:**
 - **Case 1:** DUT is able to authenticate tester successfully, so Tester is allowed to access system functions in DUT.
 - **Case 2:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access system functions in DUT.
- e. **Evidence Provided:** - Success message of login as per provided documentation

- **Test Case Number: 2**

- a. **Test Case Name:** TC_SYS_FUN_USAGE_2
- b. **Test Case Description:** Tester should provide at least one authentication attribute for local access
- c. **Execution Steps:**
 - The tester follows the instructions in OEM document to access the provided system functions using correct credentials (Case 3)
 - The tester follows the instructions in OEM document to access the provided system functions using incorrect credentials (Case 4)
- d. **Test Observations:**
 - **Case 3:** DUT is able to authenticate tester successfully, so Tester is allowed to access system functions of DUT.
 - **Case 4:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access system functions of DUT.
- e. **Evidence Provided:** - Success message of login as per provided documentation

➤ **Test Case Number:** 3

a. **Test Case Name:** TC_SYS_FUN_USAGE_3

b. **Test Case Description:** Tester should provide at least one authentication attribute for machine accounts

c. **Execution Steps:**

- The tester follows the instructions in OEM document to access the provided system functions using correct credentials (Case 5)
- The tester follows the instructions in OEM document to access the provided system functions using incorrect credentials (Case 6)

d. **Test Observations:**

- **Case 5:** DUT is able to authenticate tester successfully, so Tester is allowed to access system functions of DUT.
- **Case 6:** DUT is not able to authenticate tester successfully, so Tester is not allowed to access system functions of DUT.

e. **Evidence Provided:-** Screenshot of Terminal

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_SYS_FUN_USAGE_1		
2	TC_SYS_FUN_USAGE_2		
3	TC_SYS_FUN_USAGE_3		

(Tester has to execute the test in similar fashion for other databases)

Securing Networks

2.2.2 TSTP Report for Evaluation of Authentication Support – External

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.2.2
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 2: Authentication Attribute Management
2. **<Security Requirement No & Name >** 2.2.2 Authentication Support – External
3. **<Requirement Description: >** If the SMF supports external authentication mechanism such as AAA server (for authentication, authorization and accounting services, then the communication between SMF and the external authentication entity shall be protected using the authentication and related service protocols built strictly using the Secure cryptographic controls prescribed in Table1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only.

4. **DUT Confirmation Details**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. **DUT Configuration:-** Check OEM documentation to find all the tools used for deploying an external authentication mechanism.

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

a. For SSH with usage of FreeRadius as AAA Server

Command used: **dpkg -s libpam-radius-auth | grep Version** (To get version information)

```
(base) pratik@pratik-Super-Server:~/radsecproxy$ dpkg -s libpam-radius-auth | grep Version
Version: 1.4.0-3
```

Command used: **dpkg -s radsecproxy | grep Version** (To get version information)

```
(base) pratik@pratik-Super-Server:~/radsecproxy$ dpkg -s radsecproxy | grep Version
Version: 1.8.1-1
Config-Version: 1.8.1-1
```

Command used: **cat /etc/pam.d/sshd** (To get configuration information)

Verify that the standard UNIX authentication for sshd is disabled and RADIUS authentication is used



```
# Adding radius auth for TSTP Check
auth sufficient pam_radius_auth.so

# PAM configuration for the Secure Shell service

# Standard Un*x authentication.
#@include common-auth

# Disallow non-root logins when /etc/nologin exists
account required pam_nologin.so

# Uncomment and edit /etc/security/access.conf to
# access limits that are hard to express in sshd
# account required pam_access.so

# Standard Un*x authorization.
@include common-account

# Adding radius auth for TSTP Check
auth sufficient pam_radius_auth.so

# PAM configuration for the Secure Shell service

# Standard Un*x authentication.
#@include common-auth

# Disallow non-root logins when /etc/nologin exists
account required pam_nologin.so

# Uncomment and edit /etc/security/access.conf to
# access limits that are hard to express in sshd
# account required pam_access.so

# Standard Un*x authorization.
@include common-account
```

Command used: **cat /etc/radsecproxy.conf** (To get configuration information)

Verify that the certificates are situated as per the address provided by the configuration file

```
# The simplest configuration you can do is:
tls default {
    # You must specify at least one of CACertificateFile or CACertificatePath
    # for TLS to work. We always verify peer certificate (client and server)
    CACertificateFile /home/pratik/certs_freeradius/ca.pem
    # CACertificatePath /home/pratik/certs_freeradius

    # You must specify the below for TLS, we always present our certificate
    CertificateFile /home/pratik/certs_freeradius/client.pem
    CertificateKeyFile /home/pratik/certs_freeradius/client.key
    # Optionally specify password if key is encrypted (not very secure)
    CertificateKeyPassword "whatever"
    CipherList "AES256+SHA384"
    #TLSVersion TLS1_2:
    #
    # Optionally enable CRL checking
    # CRLCheck on
    # Optionally specify how long CAs and CRLs are cached, default forever
    # CacheExpiry 3600

    # Optionally require that peer certs have one of the specified policyOIDs
    # policyoid 1.2.3 # this option can be used multiple times
    # policyoid 1.3.4
}
```

Also note the CipherList parameter in the configuration. If no such option exists, then consider the parameter as "DEFAULT"

Command used: **openssl ciphers -v <cipherlist_parameter>**

Verify that the cipher suits listed are as per the latest document "Cryptographic Controls For Indian Telecom Security Assurance Requirements (ITSAR)" only

```
pratik@pratik-HP-Laptop-15g-br0xx:~/radsecproxy$ openssl ciphers -v "AES256+SHA384"
TLS_AES_256_GCM_SHA384 TLSv1.3 Kx=any Au=any Enc=AESGCM(256) Mac=AEAD
TLS_CHACHA20_POLY1305_SHA256 TLSv1.3 Kx=any Au=any Enc=CHACHA20/POLY1305(256) Mac=AEAD
TLS_AES_128_GCM_SHA256 TLSv1.3 Kx=any Au=any Enc=AESGCM(128) Mac=AEAD
ECDHE-ECDSA-AES256-SHA384 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AES(256) Mac=SHA384
ECDHE-RSA-AES256-SHA384 TLSv1.2 Kx=ECDH Au=RSA Enc=AES(256) Mac=SHA384
ECDHE-PSK-AES256-CBC-SHA384 TLSv1 Kx=ECDHEPSK Au=PSK Enc=AES(256) Mac=SHA384
RSA-PSK-AES256-CBC-SHA384 TLSv1 Kx=RSAPSK Au=RSA Enc=AES(256) Mac=SHA384
DHE-PSK-AES256-CBC-SHA384 TLSv1 Kx=DHEPSK Au=PSK Enc=AES(256) Mac=SHA384
PSK-AES256-CBC-SHA384 TLSv1 Kx=PSK Au=PSK Enc=AES(256) Mac=SHA384
```

b. Similarly, and so on for other tools/protocols used.

6. Preconditions

- Documentation that lists the tools used for deploying the AAA server along with the method used to protect the data.
- List of user credentials that can be used to login into the DUT using AAA authentication
- IP address of the DUT should be provided, either automatically detected and assigned by OEM, or assigned manually by Tester
- DUT contains valid certificates that are required for authentication with AAA server

7. **Test Objective:** - To verify that the DUT communicates with AAA server using the authentication and related service protocols

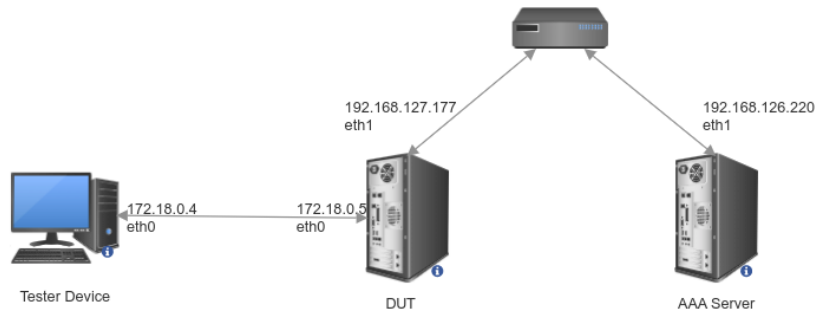
8. Test Plan:

8.1. Number of Test Case Scenarios

8.1.1. Test Scenario for SSH with usage of Free Radius as AAA Server

- This test scenario is when tester tries to login into the DUT using ssh and Free radius tool is used for having external authentication mechanism (Additional Test scenarios based on the OEM document)

8.2. Test Setup Diagram



8.3. Tools Required: Wireshark, Open SS

8.4. Test Execution Steps

- Launch the Wireshark app on the bridge interface.
- Try to login into the DUT device.
- Capture the packets flowing between DUT and AAA Server

9. **Expected Results for Pass:-** DUT communicates with AAA server for the authentication of the tester using the authentication and related service protocols

10. **Expected Format of Evidence:** Screenshots of Wireshark and pcap file

11. Test Execution

➤ Test Case Number: 01

a. **Test Case Name:** TC1_EXTERNAL_AUTH_SSH_FREERADIUS

b. **Test Case Description:** Upon receiving authentication request from tester, DUT should communicate with AAA server using the authentication and related service protocols

c. **Execution Steps:**

- Tester shall launch wireshark on the bridge interface to capture the packets between DUT and AAA server with appropriate filters
- Tester shall then try to login into the DUT using following command
 - **ssh <hostname_DUT>@<DUT IP address>**
- Tester shall then stop capturing packets and observe the packets exchanged

9751 39.851882565	192.168.127.177	192.168.126.220	TLsv1.3	299 Client Hello
9753 39.852909988	192.168.126.220	192.168.127.177	TLsv1.3	165 Hello Retry Request, Change Cipher Spec
9755 39.853144694	192.168.127.177	192.168.126.220	TLsv1.3	338 Change Cipher Spec, Client Hello
9766 39.857423052	192.168.126.220	192.168.127.177	TLsv1.3	1514 Server Hello, Application Data, Application Data
9767 39.857679396	192.168.126.220	192.168.127.177	TLsv1.3	1770 Application Data, Application Data, Application Data
9769 39.859635003	192.168.127.177	192.168.126.220	TLsv1.3	2774 Application Data, Application Data, Application Data
11103 48.193401098	192.168.127.177	192.168.126.220	TLsv1.3	177 Application Data
11104 48.194864207	192.168.126.220	192.168.127.177	TLsv1.3	120 Application Data

d. Test Observations:

- DUT communicates with AAA server using the authentication and related service protocols

e. Evidence Provided

- Screenshot of Wireshark capture and pcap file

12. Test Case Result

SL No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_EXTERNAL_AUTH_SSH_FREERADIUS		

(Tester has to execute the test in similar fashion for other protocols and tools)



2.2.3 TSTP for Protection against brute force and dictionary attacks

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.2.3
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 2: Authentication Attribute Management
2. **<Security Requirement No & Name >** 2.2.3 Protection against brute force and dictionary attacks
3. **<Requirement Description: >** Protection against brute force and dictionary attacks that hinder authentication attribute guessing shall be implemented in SMF. Brute force and dictionary attacks aim to use automated guessing to ascertain authentication attribute for user and machine accounts. Various measures or a combination of the following measures can be taken to prevent this:
 - i. Using the timer delay (this delay could be the same or increased depending on the operator's policy for each attempt) for each newly entered password input following an incorrect entry ("tar pit").
 - ii. Blocking an account following a specified number of incorrect attempts. However, it has to be taken into account that this solution needs a process for unlocking and an attacker can force this to deactivate accounts and make them unusable.
 - iii. Using an authentication attribute blacklist to prevent vulnerable passwords.
 - iv. Using CAPTCHA to prevent automated attempts (often used for Web applications).
 - v. Achieve higher security, two or more of the measures indicated above shall be mandatorily supported by SMF. An exception to this requirement is machine accounts.

[Reference: TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.3.3]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details

- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./system.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal
```

5. **DUT Configuration:** DUT should have a method to configure the timer delay following an incorrect password, blocking the account after specified number of incorrect attempts, authentication attribute blacklist, and captcha (for web applications). Here Linux is used for testing. All Linux distributions have Pluggable Authentication Module (PAM) for authentication. Tester needs to check the DUT configuration as per the method used for configurations of timer delay, blocking accounts, blacklisting and CAPTCHA.

Command used: **microstack.openstack --version** (To find version information of microstack)

```
microstack@microstack:~$ microstack.openstack --version
openstack 5.2.0
```

To get the hash of OS image is using Virtual Machine

Command used: **sha256sum <path to OS image>** (To get hash/digest of OS Image)

```
root@os-controller $sha256sum ubuntu-22.04.1-amd64.iso
10f19c5b2b8d6db711582e0e27f5116296c34fe4b313ba45f9b201a5007056cb  ubuntu-22.04.1-amd64.iso
```

To get the hash of OS if using Docker/Kubernetes

Command used: **docker images <image name> --digests** (To get hash/digest of Container Image)

```
root@os-controller $docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 14be0685b768 sha256:c9820a44b950956a790c354700c1166a7ec648bc0d215fa438d3a339812f1d01
ubuntu focal 14be0685b768 sha256:c9820a44b950956a790c354700c1166a7ec648bc0d215fa438d3a339812f1d01
ubuntu 18.04 5d2df19066ac sha256:a3765b4d74747b5e9bdd03205b3fbc4fa19a02781c185f97f24c8f4f84ed7bbf
ubuntu bionic 5d2df19066ac sha256:a3765b4d74747b5e9bdd03205b3fbc4fa19a02781c185f97f24c8f4f84ed7bbf
ubuntu latest 58db3edaf2be sha256:9a0bdde4188b896a372804be2384015e90e3f84906b750c1a53539b585fbbe7f
ubuntu <none> a8780b506fa4 sha256:4b1d0c4a2d2aaf63b3711f34eb9fa89fa1bf53dd6e4ca954d47caebca4005c2
```

The following commands are with respect to PAM module:

To check if PAM is present: **dpkg -l | grep libpam-modules**

```
openstack@openstack:~$ dpkg -l | grep libpam-modules
ii  libpam-modules:amd64 1.3.1-5ubuntu4.6 amd64 Pluggable Authentication Modules for PAM
ii  libpam-modules-bin 1.3.1-5ubuntu4.6 amd64 Pluggable Authentication Modules for PAM - helper binaries
```

To check the PAM version: **ls -l /lib/*/libpam.so***

```
openstack@openstack:~$ ls -l /lib/*/libpam.so*
lrwxrwxrwx 1 root root 16 Jul 24 22:15 /lib/x86_64-linux-gnu/libpam.so.0 -> libpam.so.0.84.2
```

Check installation of libpam-cracklib : **dpkg -l | grep libpam-cracklib**

```
openstack@openstack:~$ dpkg -l | grep libpam-cracklib
ii  libpam-cracklib:amd64 1.3.1-5ubuntu4.6 amd64 PAM module to enable cracklib support
```

In the following DUT, programs SU, SSH, GDM, PASSWD are used as login and password changing mechanism respectively. However, the tester must check PAM-awareness of all the programs used as a login mechanism and for changing the password.

To check if programs SSH, SU and PASSWD are PAM-aware:

- **sudo ldd /usr/sbin/sshd | grep libpam.so**
- **sudo ldd /usr/bin/su | grep libpam.so**
- **sudo ldd /usr/bin/passwd | grep libpam.so**

```
openstack@openstack:~$ sudo ldd /usr/sbin/sshd | grep libpam.so
libpam.so.0 => /lib/x86_64-linux-gnu/libpam.so.0 (0x00007f40c8530000)
openstack@openstack:~$ sudo ldd /usr/bin/su | grep libpam.so
libpam.so.0 => /lib/x86_64-linux-gnu/libpam.so.0 (0x00007f7bed323000)
openstack@openstack:~$ sudo ldd /usr/bin/passwd | grep libpam.so
libpam.so.0 => /lib/x86_64-linux-gnu/libpam.so.0 (0x00007ff99c6c9000)
```

Note: Method may change based on operating system.

PAM configurations according to different cases:

- **Case 1)** DUT Configuration for testing timer delay. To check pam configuration files:
cd /etc/pam.d/ls

```

openstack@openstack:~$ cd /etc/pam.d
openstack@openstack:/etc/pam.d$ ls
chfn          common-session-noninteractive  login      runuser-l
chpasswd      cron                          newusers   sshd
chsh          cups                          other      su
common-account  gnome-screensaver           passwd     sudo
common-auth    lightdm                     polkit-1   systemd-user
common-password lightdm-autologin           ppp        unity
common-session lightdm-greeter             runuser

```

Check the following configuration line in the common-auth file:

auth requisite pam_faildelay.so delay = 9000000

Note: Here the DUT is set to provide a delay of 9 seconds on an incorrect password entry done by programs SU, SSH, GDM.

Use the command: **cat /etc/pam.d/common-auth** to verify the configuration information.

```

openstack@openstack:~$ cat /etc/pam.d/common-auth
#auth required pam_tally2.so onerr=fail deny=3 unlock_time=100
#auth required pam_unix.so authfail delay=2000000
auth requisite pam_faildelay.so delay=9000000

#
# /etc/pam.d/common-auth - authentication settings common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of the authentication modules that define
# the central authentication scheme for use on the system
# (e.g., /etc/shadow, LDAP, Kerberos, etc.). The default is to use the
# traditional Unix authentication mechanisms.
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules. See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
auth      [success=1 default=ignore]      pam_unix.so nullok_secure
# here's the fallback if no module succeeds
auth      requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
auth      required                       pam_permit.so
# and here are more per-package modules (the "Additional" block)
# end of pam-auth-update config

#my own line
#auth requisite                       pam_faildelay.so delay=9000000
openstack@openstack:~$

```

- **Case 2) DUT Configuration for testing locking of account on specified number of incorrect attempts:**

To check pam configuration files: **cd /etc/pam.d/ls**

```

openstack@openstack:~$ cd /etc/pam.d
openstack@openstack:/etc/pam.d$ ls
chfn          common-session-noninteractive  login      runuser-l
chpasswd      cron                          newusers   sshd
chsh          cups                          other      su
common-account  gnome-screensaver           passwd     sudo
common-auth    lightdm                     polkit-1   systemd-user
common-password lightdm-autologin           ppp        unity
common-session lightdm-greeter             runuser

```

Check the following configuration line in the common-auth file:

auth required pam_tally2.so onerr=fail deny=3

Note: Here the DUT is set to lock the account on specified number of incorrect attempts. This configuration is for same delay. Similarly, the configuration for increased delay should be checked as per the operator's policy which should be provided in the OEM documentation. Use the command: **nano /etc/pam.d/common-auth** to verify the configuration information.

```
GNU nano 2.5.3 File: common-auth
auth required pam_tally2.so onerr=fail deny=3
#auth required pam_unix.so authfail delay=2000000
#
# /etc/pam.d/common-auth - authentication settings common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of the authentication modules that define
# the central authentication scheme for use on the system
# (e.g., /etc/shadow, LDAP, Kerberos, etc.). The default is to use the
# traditional Unix authentication mechanisms.
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules. See
# pam-auth-update(8) for details.
#
# here are the per-package modules (the "Primary" block)
auth [success=1 default=ignore] pam_unix.so nullok_secure
# here's the fallback if no module succeeds
auth requisite pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
auth required pam_permit.so
# and here are more per-package modules (the "Additional" block)
# end of pam-auth-update config
```

➤ **Case 3) DUT Configuration for testing authentication attribute blacklist support:**

Go to the pam directory: **cd /etc/pam.d/**

```
openstack@openstack:~$ cd /etc/pam.d
openstack@openstack:/etc/pam.d$ ls
chfn          common-session-noninteractive  login          runuser-l
chpasswd      cron                          newusers      sshd
chsh          cups                          other          su
common-account  gnome-screensaver            passwd        sudo
common-auth    lightdm                      polkit-1      systemd-user
common-password lightdm-autologin             ppp           unity
common-session lightdm-greeter               runuser
```

Check the following configuration line in the common-password file:

password required pam_cracklib.so enforce_for_root
dictpath=/var/cache/cracklib/cracklib_dict

Use the command: **cat /etc/pam.d/common-password** to verify the configuration information.


```

openstack@openstack:~$ cat /etc/pam.d/common-password
#
# /etc/pam.d/common-password - password-related modules common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of modules that define the services to be
# used to change user passwords. The default is pam_unix.
#
# Explanation of pam_unix options:
#
# The "sha512" option enables salted SHA512 passwords. Without this option,
# the default is Unix crypt. Prior releases used the option "md5".
#
# The "obscure" option replaces the old 'OBSOLETE_CHECKS_ENAB' option in
# login.defs.
#
# See the pam_unix manpage for other options.
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules. See
# pam-auth-update(8) for details.
#
# here are the per-package modules (the "Primary" block)
password      requisite      pam_cracklib.so retry=3 minlen=8 difok=3 dictpath=/usr/share/dict/cracklib-small
password      required       pam_cracklib.so enforce_for_root dictpath=/var/cache/cracklib/cracklib_dict
password      [success=1 default=ignore] pam_unix.so obscure use_authtok try_first_pass sha512
#
# here's the fallback if no module succeeds
password      requisite      pam_deny.so
#
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
password      required       pam_permit.so
#
# and here are more per-package modules (the "Additional" block)
password      optional       pam_gnome_keyring.so
#
# end of pam-auth-update config

```

Verify the last words present in the cracklib_dict using the command:
tail /var/cache/cracklib/cracklib_dict

```

openstack@openstack:~$ tail /var/cache/cracklib/cracklib_dict
zorn
zoroaster
zoroastrian
zounds
z's
zucchini
zurich
zxcvbn
zxcvbnm
zygote

```

➤ Case 4) For Apache httpd:

Command used: **httpd -v** (To get version information)

```

amf@localhost $ httpd -v
Server version: Apache/2.4.57 (Unix)
Server built:   May 11 2023 19:37:07

```

6. Preconditions

- At least one user account has been created as per manufacturer's instructions and the tester has valid credentials as an authorized user. In addition, all the user accounts to be tested, all the login mechanisms on all available interfaces shall be stated in the documentation.
- Directions of how to configure timer delay expected after each newly entered password input following an incorrect entry and the default value of this timer delay are identified in the documentation accompanying the Network Product. It should also be stated in the document whether the delay is expected to be the same or increased after each incorrect attempt.
- Directions of how to configure the blocking of an account following a specified number of incorrect attempts are identified in the documentation accompanying the Network Product. Also, a solution for unlocking the account must be stated as well.

- Directions of how to configure authentication attribute blacklist to prevent vulnerable passwords and at least one blacklisted as well as non-blacklisted password are identified in the documentation accompanying the Network Product.
- Directions of how to configure the web interface with CAPTCHA feature to prevent the automated attempts is identified in the documentation accompanying the Network Product. CAPTCHA feature is optional, and test is done only if implemented.

Note: Password management and blacklist configuration may be done in a separate node that is different to the node under test, e.g. a SSO server or any other central credential manager.

7. **Test Objective:-** To ensure that the system uses two or more of the measures indicated below shall be mandatorily supported by SMF, to ensure that the system uses a mechanism with adequate protection against brute force and dictionary attacks to check whether system follows commonly used preventive measures which are mentioned below.:

1. Using the timer delay after each incorrect password input.
2. Blocking an account following a specified number of incorrect attempts. However, administrator has to keep in account that this solution needs a process for unlocking and an attacker can utilize this process to deactivate the accounts and make them unusable.
3. Using a password blacklist to prevent vulnerable passwords.
4. Using CAPTCHA to prevent automated attempts (often used for Web interface).

8. **Test Plan**

8.1. **Number of Test Scenarios:**

8.1.1 Test Scenario for checking timer delay on each newly entered password following an incorrect entry. In this test case, we take an example of SSH, SU and GDM.

Note: The tester needs to perform this test for all types of login mechanisms and accounts mentioned in the OEM document.

8.1.2 Test Scenario for checking of blocking of an account following specified incorrect attempts and a solution for unlocking the locked account. In this test case, we take an example of SU.

Note: The tester needs to perform this test for all types of login mechanisms and accounts mentioned in the OEM document.

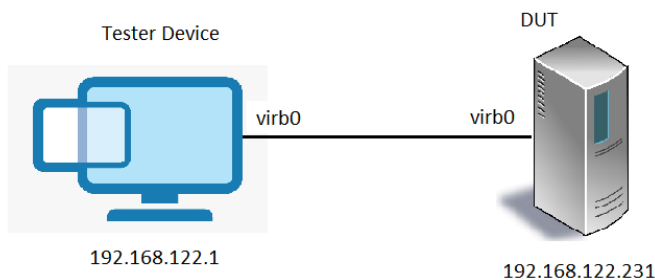
8.1.3 Test Scenario for checking of the blacklisting of password. In this test case, we take an example of PASSWD.

Note: The tester needs to perform this test for all types of passwords changing mechanisms and accounts mentioned in the OEM document.

8.1.4 Test Scenario for checking of CAPTCHA.

Note: The tester needs to perform this test for all types of web application login interfaces and accounts mentioned in the OEM document.

8.2. Test Bed Diagram



8.3. Tools Required Linux CLI

8.4. Test Execution Steps

- Use the tester device to login into a user account with correct and incorrect password for each of the cases.
- Ensure that DUT provides a timer delay before entering new password when an incorrect password is entered by the tester. (Case 1). The delay could be same or increased for each attempt depending on operators' policy.
- Ensure that the DUT blocks the account on specified number of incorrect attempts and a solution to unlock the locked account. (Case 2)
- Ensure that the DUT blacklists the password entered by the tester which is belonging to DUT's blacklist dictionary when the tester tries to change it. (Case 3)
- Ensure that the DUT enforces Captcha verification. (Case 4)

Note: Repeat the above steps for all the login mechanisms and accounts as per the OEM documentation.

9. Expected Results for Pass:

- **Case 1:** Tester should be able to login into DUT with the correct password. With an incorrect password, the tester must get a delay from the DUT before entering a new password.
- **Case 2:** Tester should be able to login into DUT with the correct password. With an incorrect password, the account must be locked out by the DUT on specified number of incorrect attempts.
- **Case 3:** Tester should be able to change the password of its account using a non-blacklisted password. With an blacklisted password, the tester must not be able to change the password of its account by the DUT.

- **Case 4:** Tester should be able to login with the correct CAPTCHA. On null captcha, the tester must not be allowed to login.

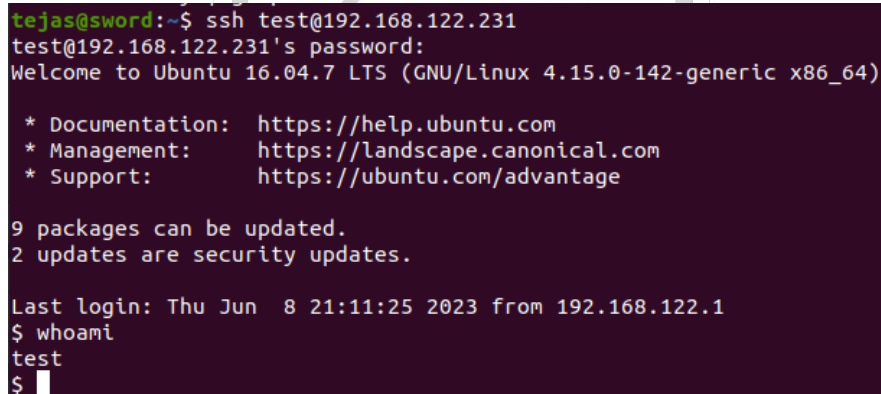
10. **Expected Format of Evidence:** Screenshots of Terminal

11. **Test Execution:**

➤ **Test Case Number:** 01

- Test Case Name:** TC1_PROTECT_AGAINST_BRUTE_FORCE_AND_DICTIONARY_ATTACKS
- Test Case Description:** DUT should provide timer delay to the tester device for each newly entered password input following an incorrect entry.
- Execution Steps:**
 - Check the default value of delay mentioned in precondition 2.
 - **Case A)** The tester tries to login to the user “test” using the following login commands and enters correct password. To test the timer delay not being provided by the DUT in case of correct passwords, enter the correct password.

- For SSH: **ssh test@192.168.122.231**



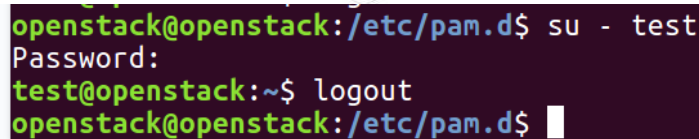
```
tejas@sword:~$ ssh test@192.168.122.231
test@192.168.122.231's password:
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 4.15.0-142-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

9 packages can be updated.
2 updates are security updates.

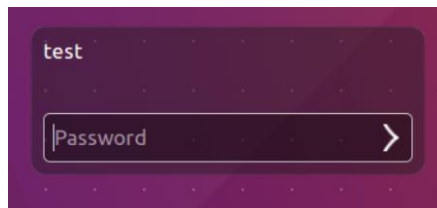
Last login: Thu Jun  8 21:11:25 2023 from 192.168.122.1
$ whoami
test
$
```

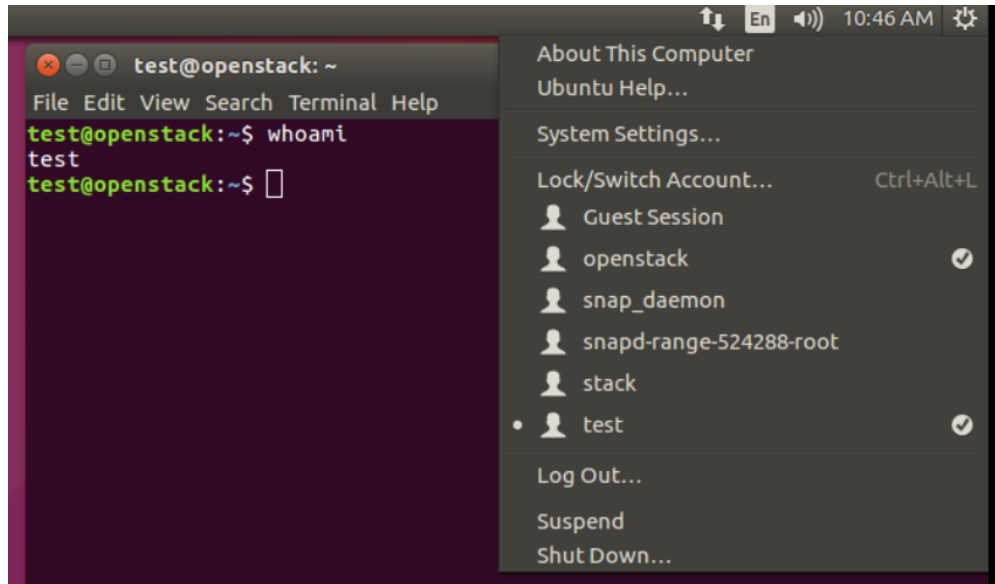
- For SU: **su - test**



```
openstack@openstack:/etc/pam.d$ su - test
Password:
test@openstack:~$ logout
openstack@openstack:/etc/pam.d$
```

- For GDM: Using the username “test”





- **Case B)** The tester tries to login to the user “test” using the following login commands and enters incorrect password. To test the timer delay of DUT, the tester must input with incorrect password. Note: This test should be carried out for the appropriate timer delay to be expected (same or increasing).
- For SSH: **ssh test@192.168.122.231**

```
tejas@sword:~$ ssh test@192.168.122.231
test@192.168.122.231's password:
```

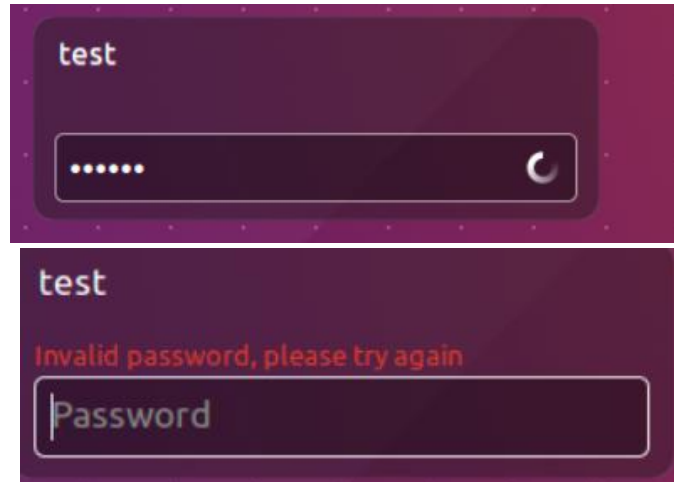
```
tejas@sword:~$ ssh test@192.168.122.231
test@192.168.122.231's password:
Permission denied, please try again.
test@192.168.122.231's password:
Permission denied, please try again.
test@192.168.122.231's password:
test@192.168.122.231: Permission denied (publickey,password).
tejas@sword:~$
```

- For SU:

```
openstack@openstack: ~
openstack@openstack:~$ su - test
Password:
```

```
openstack@openstack: ~  
openstack@openstack:~$ su - test  
Password:  
su: Authentication failure  
openstack@openstack:~$
```

- For GDM:



d. **Test Observations:**

- **Case A:** Tester should not get a delay from DUT in case of correctly entered password in case of SSH, SU and GDM.
- **Case B:** Tester should get a delay of 9 seconds from DUT to enter another password, following an incorrect password.

e. **Evidence Provided:-** Screenshot of Terminal

Securing Networks

➤ **Test Case Number:** 02

- a. **Test Case Name:** TC2_PROTECT_AGAINST_BRUTE_FORCE_AND_DICTIONARY_ATTACKS
- b. **Test Case Description:** DUT should block the account if the tester device is exceeding incorrect login attempts. There should also be a mechanism for unlocking the locked account.
- c. **Execution Steps:**
 - Check the value of incorrect login attempts mentioned in precondition 3.
 - **Case A)** The tester tries to login to the user “test”. To test the account locking not done by the DUT, enter a correct password. Check the log file of pam_tally2 to get the number of failed attempts for user “test” using the following command:
sudo pam_tally2 -u test

- For SSH: **ssh test@192.168.122.231**

```
tejas@sword:~$ ssh test@192.168.122.231
test@192.168.122.231's password:
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 4.15.0-142-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

9 packages can be updated.
2 updates are security updates.

Last login: Thu Jun  8 21:11:25 2023 from 192.168.122.1
$ whoami
test
$
```

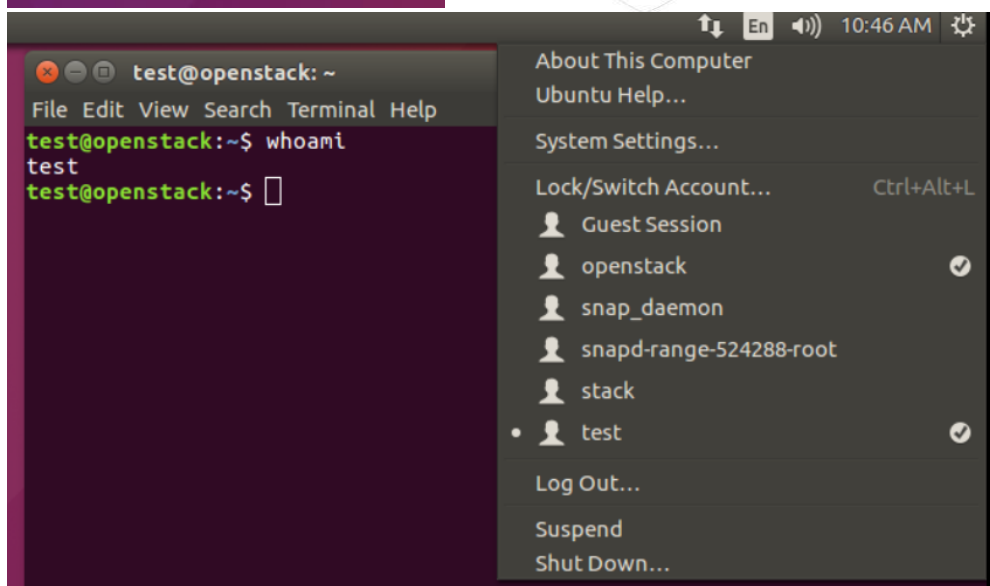
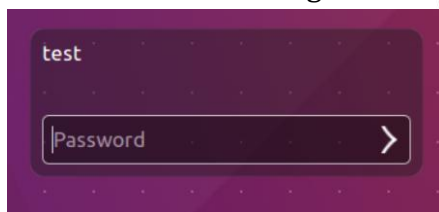
```
openstack@openstack:/etc/pam.d$ sudo pam_tally2 -u test
Login      Failures Latest failure    From
test              0
```

- For SU: **su - test**

```
openstack@openstack:/etc/pam.d$ su - test
Password:
test@openstack:~$ logout
openstack@openstack:/etc/pam.d$
```

```
openstack@openstack:/etc/pam.d$ sudo pam_tally2 -u test
Login      Failures Latest failure    From
test              0
```

- For GDM: Using the username "test"



```
openstack@openstack:/etc/pam.d$ sudo pam_tally2 -u test
Login      Failures Latest failure    From
test              0
```

- **Case B)** The tester tries to login to the user “test”. To test the account locking done by the DUT, the tester must input with incorrect password. Check the log file of pam_tally2 to get the number of failed attempts for user “test” using the following command:

sudo pam_tally2 -u test

- **The** DUT should be able to clear the failed attempts to unlock the user using the following command:

sudo pam_tally2 -u test -r

- For SSH: **ssh -l test@192.168.122.231**

```
tejas@sword:~$ ssh test@192.168.122.231
test@192.168.122.231's password:
Permission denied, please try again.
test@192.168.122.231's password:
Permission denied, please try again.
test@192.168.122.231's password:
test@192.168.122.231: Permission denied (publickey,password).
tejas@sword:~$ ssh test@192.168.122.231
^C
tejas@sword:~$
```

```
openstack@openstack:~$ sudo pam_tally2 -u test
Login      Failures Latest failure    From
test              3    06/13/23 10:41:02  192.168.122.1
```

Unlocking the “test” account as root user:

```
openstack@openstack:~$ sudo pam_tally2 -u test -r
Login      Failures Latest failure    From
test              3    06/13/23 10:41:02  192.168.122.1
openstack@openstack:~$ sudo pam_tally2 -u test
Login      Failures Latest failure    From
test              0
```

- For SU: **su - test**

```
openstack@openstack:/etc/pam.d$ su - test
Password:
su: Authentication failure
openstack@openstack:/etc/pam.d$ su - test
Password:
su: Authentication failure
openstack@openstack:/etc/pam.d$ su - test
Password:
su: Authentication failure
openstack@openstack:/etc/pam.d$ su - test
Account locked due to 4 failed logins
Password:
su: Authentication failure
```



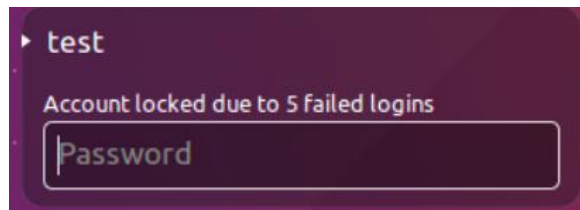
```
openstack@openstack:/etc/pam.d$ sudo pam_tally2 -u test
Login      Failures Latest failure    From
test       4      06/13/23 10:21:56    /dev/pts/6
```

Note: The number of times the account is tried to login with incorrect password (without the account being locked) and the correct/incorrect attempts (with account being locked) updates the lock count.

Unlocking the "test" account as root user:

```
openstack@openstack:/etc/pam.d$ sudo pam_tally2 -u test -r
Login      Failures Latest failure    From
test       4      06/13/23 10:21:56    /dev/pts/6
openstack@openstack:/etc/pam.d$ sudo pam_tally2 -u test
Login      Failures Latest failure    From
test       0
```

- For GDM: Using the username "test"



```
openstack@openstack:~$ sudo pam_tally2 -u test
Login      Failures Latest failure    From
test       5      06/13/23 10:49:02    :2
openstack@openstack:~$ sudo pam_tally2 -u test -r
Login      Failures Latest failure    From
test       5      06/13/23 10:49:02    :2
openstack@openstack:~$ sudo pam_tally2 -u test
Login      Failures Latest failure    From
test       0
```

d. Test Observations:

- Case A: Tester should be able to login with the correct password and the account must not be locked by the DUT.
- Case B: Tester should be locked out after specified number of incorrect attempts by the DUT.

e. Evidence Provided:- Screenshot of Terminal

- **Test Case Number:** 03

- Test Case Name:** TC3_PROTECT_AGAINST_BRUTE_FORCE_AND_DICTIONARY_ATTACKS
- Test Case Description:** DUT should use an authentication attribute blacklist to prevent vulnerable passwords entered by the tester.
- Execution Steps:**
 - The tester tries uses the following command to change the password for user "test":
sudo passwd test

- **Case A)** In order to test that a password not mentioned in this cracklib_dict is allowed by the DUT, enter a password which is not in this dictionary: “Password@123”. Note: The tester needs to use the non-blacklisted password given in OEM documentation.

```
openstack@openstack:~$ sudo passwd test
New password:
Retype new password:
passwd: password updated successfully
```

- **Case B)** In order to test that a password mentioned in this cracklib_dict is blacklisted by the DUT, enter a password which is in this dictionary: “zorn”. Note: The tester needs to use the blacklisted password given in OEM documentation.

```
openstack@openstack:~$ sudo passwd test
New password:
BAD PASSWORD: it is based on a dictionary word
```

d. Test Observations:

- Case A: Tester should be able to change the password of the user “test” successfully using the password “Password@123” which is not mentioned in the dictionary cracklib_dict. The tester should be able to change the password to a non-blacklisted password.
- Case B: Tester should not be able to change the password of the user “test” successfully using the password “zorn” which is mentioned in the dictionary cracklib_dict. It should be blacklisted by the DUT with an appropriate error message. The tester should not be able to change the password to a blacklisted password.

e. Evidence Provided:- Screenshot of Terminal

Securing Networks

➤ **Test Case Number:** 04

- Test Case Name:** TC3_PROTECT_AGAINST_BRUTE_FORCE_AND_DICTIONARY_ATTACKS
- Test Case Description:** DUT should use a CAPTCHA verification in case of Web Applications.
- Execution Steps:**
 - **Case A)** User after entering valid credentials should also solve a CAPTCHA challenge which in this case is an image text CAPTCHA

SIGN IN NOW

User

C4QT1T

⌂ SIGN IN

Welcome

- **Case B)** Users should not be allowed to login their accounts regardless of their credentials if CAPTCHA is not entered correctly or not entered at all.

SIGN IN NOW

Invalid CAPTCHA

User ID

Password

zXJaZr

⌂ SIGN IN

d. **Test Observations:**

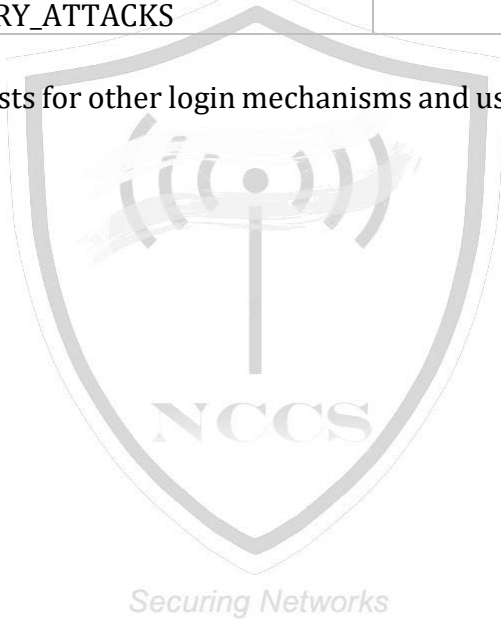
- Case A: After the correct credentials are provided along with valid CAPTCHA, user should be able to log in successfully
- Case B: Valid captcha is enforced on filling in null captcha.

e. **Evidence Provided:-** Screenshot of Terminal

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_PROTECT_AGAINST_BRUTE_FORCE_AND_DICTIONARY_ATTACKS		
2	TC2_PROTECT_AGAINST_BRUTE_FORCE_AND_DICTIONARY_ATTACKS		
3	TC3_PROTECT_AGAINST_BRUTE_FORCE_AND_DICTIONARY_ATTACKS		
4	TC4_PROTECT_AGAINST_BRUTE_FORCE_AND_DICTIONARY_ATTACKS		

(Tester has to execute the tests for other login mechanisms and user accounts mentioned in the OEM document)



2.2.4 TSTP Report for Evaluation for Enforcement of Strong Password under 5G ITSAR

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.2.4
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 2: Authentication Attribute Management

2. **<Security Requirement No & Name >** 2.2.4 Enforce Strong Password

3. **<Requirement Description: >**

- a) The configuration setting shall be such that SMF shall only accept passwords that comply with the following complexity criteria:
 - i. Absolute minimum length of 8 characters (shorter lengths shall be rejected by the SMF). It shall not be possible setting this absolute minimum length to a lower value by configuration.
 - ii. Password shall mandatorily comprise all the following four categories of characters: - at least 1 uppercase character (A-Z) - at least 1 lowercase character (a-z) - at least 1 digit (0-9) - at least 1 special character (e.g. @;!\$.)
- b) The minimum length of characters in the passwords and the set of allowable special characters shall be configurable by the operator. The special characters may be categorized in sets according to their Unicode category.
- c) If a central system is used for user authentication password policy, then additional assurance shall be provided that the central system enforces the same password complexity rules as laid down for the local system in this sub-clause.
- d) If a central system is not used for user authentication, the assurance on password complexity rules shall be performed on the SMF.
- e) When a user is changing a password or entering a new password, SMF /central system checks and ensures that it meets the password requirements. Above requirements shall be applicable for all passwords used (e.g. application-level, OS-level, etc.). Password shall not be stored in clear text in the system; passwords shall be salted and hashed.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.3.1]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal
```

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2dea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

DUT should have a method to configure the password settings. Here Linux is used for testing.

All Linux distributions have Pluggable Authentication Module (PAM) for authentication.

Check the configuration at /etc/security/pwquality.conf

```
open  pwquality.conf
/etc/security

# Configuration for systemwide password quality limits
# Defaults:
#
# Number of characters in the new password that must not be present in the
# old password.
# difok = 1
#
# Minimum acceptable size for the new password (plus one if
# credits are not disabled which is the default). (See pam_cracklib manual.)
# Cannot be set to lower value than 6.
minlen = 8
#
# The maximum credit for having digits in the new password. If less than 0
# it is the minimum number of digits in the new password.
dcredit = 1
#
# The maximum credit for having uppercase characters in the new password.
# If less than 0 it is the minimum number of uppercase characters in the new
# password.
ucredit = 1
#
# The maximum credit for having lowercase characters in the new password.
# If less than 0 it is the minimum number of lowercase characters in the new
# password.
lcredit = 1
```

The special characters may be configured in sets according to their Unicode category in the same file.

```
..
# The maximum credit for having other characters in the new password.
# If less than 0 it is the minimum number of other characters in the new
# password.
ocredit = 1
#
```

Check that password shall not be stored in clear text in the system; passwords shall be salted and hashed.

```

osboxes@osboxes:~$ cat /etc/pam.d/common-password
#
# /etc/pam.d/common-password - password-related modules common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of modules that define the services to be
# used to change user passwords. The default is pam_unix.
#
# Explanation of pam_unix options:
#
# The "sha512" option enables salted SHA512 passwords. Without this option,
# the default is Unix crypt. Prior releases used the option "md5".

```

6. **Preconditions:**

- Tester has rights to create user account.
- Tester has Sudo rights.

7. **Test Objective:** To verify that password structure adheres to the password complexity criteria. To verify that password structure is configurable as per the complexity criteria.

8. **Test Plan:**

8.1 **Number of Test Case Scenarios:** 1

8.1.1 **Test scenario for PAM.**

This test scenario is when the DUT uses Pluggable Authentication Module (PAM) to manage authentication and access control.

8.1.2 **Test scenarios for other cases.**

Any other authentication method based on the OS being used.

8.2 **Tools Used:** Command line

8.3 **Test Execution Steps:**

➤ **Test Case 1**

1. The tester logs into Network Product application using admin account.
2. The tester creates user A following the password complexity criteria.
3. The tester logs in as user A and attempts to change their password which contains characters from all four categories mentioned in the password complexity criteria.

➤ **Test Case 2**

1. The tester logs in with privileged account.
2. The tester modifies password structure policy on the network product by strengthening the policy (e.g., changing the minimum password length to 8+x, changing the minimum number of character Unicode categories to 4).
3. The tester logs in as user A and attempts to change their password to a password with a strength of less than that permitted by the policy strengthened in step 2 above.

9. **Expected Results for Pass:** Tester can change password only if new password fulfils the password complexity criteria

10. **Expected Format of Evidence:** Evidence suitable for the interface, e.g. screenshot containing the operation result or report in text form.

11. Test Execution:

a. **Test Case Number:** 01

b. **Test Case Name:** TC_ENFORCE_STRONG_PASSWORD

c. **Test Case Description:** Test needs to be performed to check the complexity of the password.

d. Execution Steps:

- **Case 1** - The tester logs in as user A and attempts to change their password which contains characters from all four categories mentioned in the password complexity criteria.

```
osboxes@osboxes:~$ sudo passwd tester
New password:
Retype new password:
passwd: password updated successfully
```

- **Case 2** - Changing the password policy using privileged account & entering a password that does not satisfy the newly updated password policy:

```
(current) UNIX password:
New password:
BAD PASSWORD: The password is the same as the old one
New password:
BAD PASSWORD: The password contains less than 1 digits
New password:
BAD PASSWORD: The password contains less than 1 uppercase letters
```

e. Test Observations:

Securing Networks

Testers can change passwords only if the new password fulfils the password complexity criteria.

12. Test Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_ENFORCE_STRONG_PASSWORD		

2.2.5 TSTP For Inactive Session timeout under 5G ITSAR

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.2.5
-----------------------------------	--------------------------	-----------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 2 – Authentication Attribute Management

2. **<Security Requirement No & Name >** 2.2.5 Inactive Session Timeout

3. **<Requirement Description: >**

An OAM user interactive session shall be terminated automatically after a specified period of inactivity. It shall be possible to configure an inactivity time-out period. SMF shall monitor inactive sessions of administrative login users and initiate session locking mechanism based on user configurable timers. Unlocking the session shall be permissible only by authentication. If the inactivity period further continues for a defined period, Session /user ID time out must occur after this inactivity. The timer values can be admin configurable as per requirement, normally set between 2 to 5 minutes.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.5.2]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

In Linux, For checking Inactivity Time Period Configuration:

Command used: **gsettings get org.gnome.desktop.session idle-delay**

```
amf@localhost $ gsettings get org.gnome.desktop.session idle-delay
uint32 300
```

Here, the command gives 300, which session will be auto logged out after the user remains inactive for 300 seconds.

To set the inactive time use below command

gsettings set org.gnome.desktop.session idle-delay <time in seconds>

For setting Inactivity Timeout Period for Remote Session,

Locate the configuration file for the session management. This may vary depending on the Linux distribution.

Common files include **/etc/profile**, **~/.bashrc**, or **/etc/bash.bashrc**.

Open the configuration file using a text editor such as **nano** or **vi**.

Check whether the desired timeout duration is set.

In the below screenshot, the timeout is set to 100 seconds. This makes the user logout automatically after an inactivity of 100 seconds.

```
# Add an "alert" alias for long running commands. Use like so:
# sleep 10; alert
alias alert='notify-send --urgency=low -i "${?} = 0" && echo terminal || echo error)' "${history|tail -n 50}"

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi

TMOUT=100
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line M-E Redo

For SSH:

command used: **ssh -V** (To get version information)


```
amf@localhost $ ssh -V  
OpenSSH_8.9p1 Ubuntu-3ubuntu0.1, OpenSSL 3.0.2 15 Mar 2022
```

Similarly for other remote connection protocols.

6. **Preconditions:**

- The tester has privileges to create an OAM user interactive session.
- The tester has privileges to configure the inactivity time-out period for user interactive sessions.

7. **Test Objective/ Purpose:** - To ensure an OAM user interactive session shall be terminated at inactivity timeout.

8. **Test Plan:**

8.1 **Number of Test Scenarios:**

8.1.1 **Test Scenario for Local Login:**

This test scenario is regarding local login. Check for session termination for console access.

8.1.2 **Test Scenario for Remote Login:**

This test scenario is regarding remote login. Check for remote session termination.

8.2 **TestBed Diagram:**



8.3 **Tools required:-** Command Line Interface of the DUT, ssh, snmp or other remote access protocols.

8.4 **Test Execution Step:**

1. The tester creates an OAM user A interaction session.
2. The tester configures the inactivity time-out period for user A to x minute, for example 1 minute.
3. The tester does not make any actions on the network production in x minutes. After that, the tester checks whether OAM user A interaction session has been terminated automatically

9. **Expected Results:** OAM user A interaction session has been terminated automatically after x minute.

10. **Expected Format of Evidence:** Evidence suitable for the interface, e.g. screenshot containing the operational results.

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC1_INACTIVE_SESSION_TIMEOUT

b. **Test Case Description:** Test Needs to be conducted to ensure an OAM user interactive session shall be terminated at inactivity timeout.

c. **Execution Steps:**

- Login via console using the privilege user credentials.
- Wait for the Inactivity time-out period configured.
- Check whether the user is auto logged out after being inactive more than configured time.

d. **Test Observation:**

➤ **Case 1:** Session Terminated automatically after configured time (**Positive Testcase**)

If the user remains inactive for more than configured time, it will be auto logged out.

➤ **Test Case Number: 02**

a. **Test Case Name:** TC2_INACTIVE_SESSION_TIMEOUT

b. **Test Case Description:** Test Needs to be conducted to ensure an OAM remote interactive session shall be terminated at inactivity timeout.

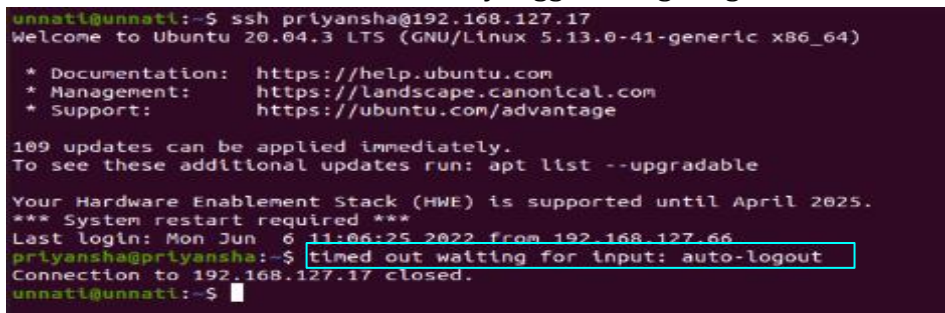
c. **Execution Steps:**

- Login remotely via console using the privilege user credentials.
- Assign an Inactivity period for remote sessions.
- Wait for Inactivity time-out period configured for remote session.
- Check for remote session termination.

d. **Test Observation:**

➤ **Case 1:** Remote Session Terminated automatically after configured time (**Positive Testcase**)

The below screenshot shows that if a remote user remains inactive for configured time duration, then it will be automatically logged out giving the time out message.



```
unnati@unnati:~$ ssh priyansha@192.168.127.17
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.13.0-41-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

109 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
*** System restart required ***
Last login: Mon Jun  6 11:06:25 2022 from 192.168.127.66
priyansha@priyansha:~$ timed out waiting for input: auto-logout
Connection to 192.168.127.17 closed.
unnati@unnati:~$
```

e. **Evidence Provided:**

A testing report which will consist of the following information:

- Screenshot of the output of the terminal of the PC through which DUT logins.

12. **Test Case Result:**

SL.No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_INACTIVE_SESSION_TIMEOUT TC2_INACTIVE_SESSION_TIMEOUT		



2.2.6 TSTP for Evaluation of Password Changes

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.1.6
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 2: Authentication Attribute Management
2. **<Security Requirement No & Name >** 2.2.6 Password Changes
3. **<Requirement Description: >** If a password is used as an authentication attribute, then the system shall offer a function that enables a user to change his password at any time. When an external centralized system for user authentication is used, it should be possible to implement this function on this system. Password change shall be enforced after initial login. SMF shall enforce password change based on password management policy. In particular, the system shall enforce password expiry. SMF shall support a configurable period for expiry of passwords. Previously used passwords shall not be allowed up to a certain number (Password History).

The number of disallowed previously used passwords shall be:

- Configurable;
- Greater than 0;
- And its minimum value shall be 3. This means that the SMF shall store at least the three previously set passwords. The maximum number of passwords that the SMF can store for each user is up to the manufacturer. When a password is about to expire, a password expiry notification shall be provided to the user.

Above requirements shall be applicable for all passwords used (e.g. application level, OS level, etc.). An exception to this requirement is machine accounts. SMF to have in-built mechanism to support this requirement. If a central system is used for user authentication password policy, then additional assurance shall be provided that the central system enforces the same password change policies as laid down for the local system in this subclause. And if a central

system is not used for user authentication, the assurance on password changes rules shall be performed on the SMF.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.3.2]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./system.out** (Used in IITH testbed to get SYSTEM version. Check with OEM manufacturer document for command specific to your SYSTEM)

Here we are assuming DUT to be SYSTEM, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SYSTEM_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

a) For Linux and PAM module

command used: **sudo nano /etc/shadow** (To get shadow file)

```
vivek:$1$fnfffc$pGteyHdicpGOffXX4ow#5:13064:0:99999:7:::
```

↓
1

↓
2

↓
3

↓
4

↓
5

↓
6

Here 6 will provide the number of days before password is to expire that user is warned that his/her password must be changed

command used: **sudo nano /etc/pam.d/common-password** (To get configuration file)

```
# here are the per-package modules (the "Primary" block)
password [success=1 default=ignore] pam_unix.so obscure sha512 remember=3
# here's the fallback if no module succeeds
password requisite pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
```

Verify that the number of passwords to remember are configurable

b) Similarly for other tools used

6. Preconditions:

- Tester has account with username and password in the network product.

- Network product vendor will provide documentation for password management policy which should include details on how to change the password, configure password expiry rule and disallowing specified number of previously used passwords.
- The network product vendor shall supply information on how many passwords the network product can store for each user in the password history.
- The tester has privilege to modify the number of disallowed previously used password.

7. **Test Objective:**

- To check whether the network product is provisioned with the functionality that enables its user to change the password at any time.
- The network product enforces password change after initial login. -
- To verify whether it has password expiry rule.
- The network product is configured to disallow specified number of previously used passwords (Password History).
- To verify the new password adheres to the password management policy (no password from password policy shall be set)

8. **Test Plan:**

8.1. **Number of Test Scenarios**

8.1.1. **Test Case for Initial Login**

- Test Scenario to test if a new user is prompted to change the password during initial login

8.1.2. **Test Case for Password Change**

- Test Scenario to test if user can change their password anytime

8.1.3. **Test Case for Password Expiry**

- Test Scenario to test if it is possible to prompt user with a warning before their password gets expired

8.1.4. **Test Case for Disallowed Passwords**

- Test Scenario to test if it possible to configure the number of disallowed passwords and verify if user adheres to it. (Additionally perform the above tests for external centralized system if external authentication mechanism is provided)

8.2. **Testbed Diagram**



8.3. Tools Required: *NULL*

8.4. Test Execution Steps:

- Tester creates a new user and then tries to login as the new user (Case 1)
- Tester changes the password of the account again (Case 2)
- Tester logs into the DUT and verifies that a warning notification is visible (Case 3)
- Tester logs into the DUT and tries to change the password to some old password (Case 4)

9. Expected Results for Pass:

- **Case 1:** User is enforced to change their password
- **Case 2:** User is able to change their password
- **Case 3:** User receives a warning notification as soon as they log into the DUT
- **Case 4:** User is not able to change their password to previously used passwords

10. Expected Format of Evidence:- Screenshots of Terminal

11. Test Execution:

➤ **Test Case Number:** 01

a. **Test Case Name:** TC_PASSWORD_CHANGES_1

b. **Test Case Description:** Test Scenario to test if a new user is prompted to change the password during initial login

c. **Execution Steps:**

- Tester creates a new user according to the documentation provided by the OEM
- Tester then shall login into the newly created account

d. **Test Observation:**

(Case 1) User is enforced to change their password

e. **Evidence Provided:-** Screenshot of Terminal

➤ **Test Case Number:** 02

a. **Test Case Name:** TC_PASSWORD_CHANGES_2

b. **Test Case Description:** Tester changes the password of the account again to check if user can change their password anytime

c. **Execution Steps:**

- Tester logs into the DUT and then tries to change their password using following command.
 - `passwd <account_name>`

```
Changing password for iith.  
Current password:  
New password:  
Retype new password:  
passwd: password updated successfully
```

d. **Test Observation:** (Case 2) User is able to change their password

e. **Evidence Provided:-** Screenshot of Terminal

➤ **Test Case Number:** 03

a. **Test Case Name:** TC_PASSWORD_CHANGES_3

b. **Test Case Description:** Test Scenario to test if it is possible to prompt user with a warning before their password gets expired

c. **Execution Steps:**

- Tester logs in as a privileged user and then adds password expiration date within coming n days (say 10 days) and set warning to be notified to the user for atleast final n+1 days (say 11 days) before expiration

- *sudo chage -M <days> -W <days> <account_name> (eg. sudo chage -M 10 -W 11 iith)*

- The tester logs out and logs into the DUT as the user whose expiration notification was generated

- The tester verifies that a warning notification is visible

```
Warning: your password will expire in 10 days
```

d. **Test Observations:** (Case 3): User receives a warning notification as soon as they log into the DUT

e. **Evidence Provided:-** Screenshot of Terminal

➤ **Test Case Number:** 04

a. **Test Case Name:** TC_PASSWORD_CHANGES_4

b. **Test Case Description:** Test Scenario to test if it possible to configure the number of disallowed passwords and verify if user adheres to it.

c. **Execution Steps:**

- Tester logs into the DUT and tries to change the password to some new password using following command. We will call this password as pass1

- *passwd <account_name>*

```
Changing password for iith.  
Current password:  
New password:  
Retype new password:  
passwd: password updated successfully
```

- Tester changes their password again twice to some pass2 and pass3 using the above command

- Now try to change the password again to pass1

```
Changing password for iith.
Current password:
New password:
Retype new password:
Password has been already used. Choose another.
New password:
```

d. **Test Observations:**

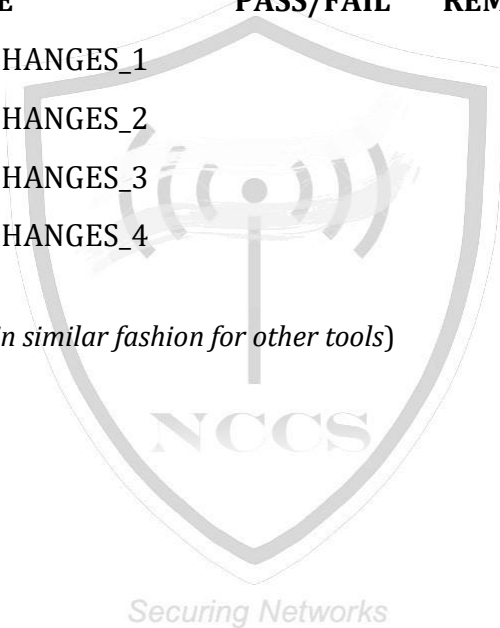
(Case 4): User is not able to change their password to previously used passwords

e. **Evidence Provided:-** Screenshot of Terminal

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	REMARKS
1	TC_PASSWORD_CHANGES_1		
2	TC_PASSWORD_CHANGES_2		
3	TC_PASSWORD_CHANGES_3		
4	TC_PASSWORD_CHANGES_4		

(Tester has to execute the test in similar fashion for other tools)



2.2.7 TSTP Report for Evaluation of Protected Authentication feedback (2.2.7 of CSR)

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.2.7
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. <ITSAR Section No & Name>2: Authentication Attribute Management

2. <Security Requirement No & Name >2.2.7 Protected Authentication feedback

3. <Requirement Description: >

The Authentication attribute shall not be displayed in such a way that it could be seen and misused by a casual local observer. Typically, the individual characters of the password are replaced by a character such as "*".

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.3.4]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Command used: **cat /etc/os-release** (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5. **DUT Configuration:** Check the Operating system and its version for the DUT to know the commands for setting access permissions for data and application execution.

- **To get the hash of configuration file if the file is a ASCII text file.**

- **Command - sha256sum DUT_config.conf**

- **Digest Hash of Tested Configuration:**

- DUT_config.conf:

- 19be54b42975f7f414c53a280373aa9466398781ecd77a2e9d090dbc6dbfdd8

- **To get the hash of OS if using docker**

- **Command - docker images --digests**

- **Digest Hash of OS:**

- DUT_IMAGE:

fd363f0e0d146c869d60649bc4f42e7008829d9d67a5dfdaf3b38ad24af7d53a

1) **visudo** file in DUT can be configured as follows to see '*' while typing password in linux.

```

Defaults    env_reset,pwfeedback

```


Using this, while writing **sudo** command '*' will be displayed. For **su** command, linux does not support password visibility to be turned on.

2) **For the GNOME** - display manager

Run the following command to disable the password reveal option:

gsettings set org.gnome.desktop.interface password-reveal-enabled false

6. **Pre-Conditions:** -

- Tester already has account or has right to create account with username and password in the network product.
- List of all interfaces that requires user login. Check for all the interfaces that requires login using password.

7. **Test Objective:** - To verify that the given password is not visible to the casual local observer.

8. **Test Plan:**

8.1 **Number of Test Scenarios:** 1

8.2 **Tools required:** Login interface of DUT

8.3 **Test Execution Steps**

1. The network product will display the login screen.
2. The tester enters the username.
3. The tester enters the password.

9. **Expected Results for Pass:**

Case 1: Tester is not able to view password while typing.

10. **Expected Format of Evidence:** Screenshots of the terminal.

11. **Test Execution:**

a. **Test Case Number:** 1

b. **Test Case Name:** TC_PROTECTED_AUTHENTICATION_FEEDBACK

c. **Test Case Description:** Tester tries to login into a user account. While entering the password, it should not be displayed in such a way that it could be seen and misused by a casual local observer.

d. **Tools Used:** login interface of DUT

e. **Execution Steps:**

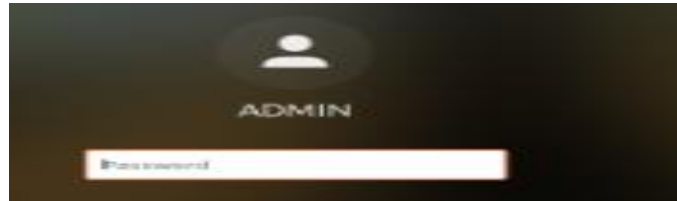
- Check that all pre-conditions are met.
- Login with your username.
- Give a password.

f. **Test Observations:** The individual characters of the password are replaced by a character such as "*" or empty space.

g. Evidence Provided:

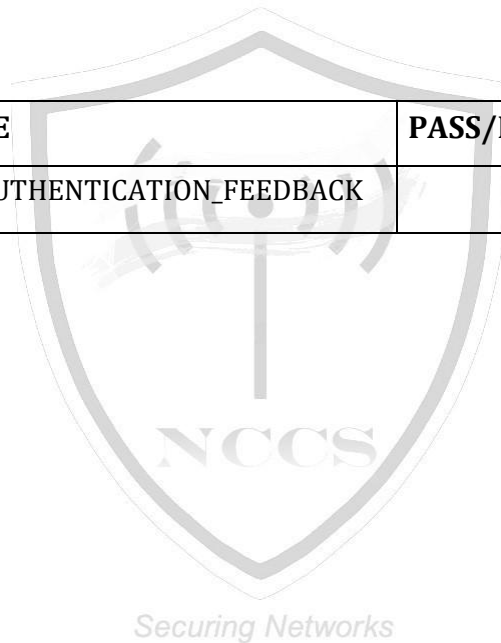
```
osboxes@osboxes:~$ su - tester
Password:
(base) supriya@supriya-Super-Server:~$ sudo ls
[sudo] password for supriya: *****
```

So, the test case is passed as we can see that password is being displayed as “*” or empty space. For Display manager, eye option is no longer present to view the password.



12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_PROTECTED_AUTHENTICATION_FEEDBACK		



2.2.8 TSTP for Evaluation of Removal of predefined or default authentication attributes

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.2.8
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list:

1. **ITSAR Section No & Name:** Section 2: Authentication Attribute Management
2. **Security Requirement No & Name:** 2.2.8 Removal of predefined or default authentication attributes
3. **Requirement Description:** Predefined or default authentication attributes shall be deleted or disabled. Normally, authentication attributes such as password or cryptographic keys will be preconfigured from producer, OEM or developer of a system. Such authentication attributes shall be changed by automatically forcing a user to change it on 1st time login to the system or the OEM provides instructions on how to manually change it.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.2.3]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

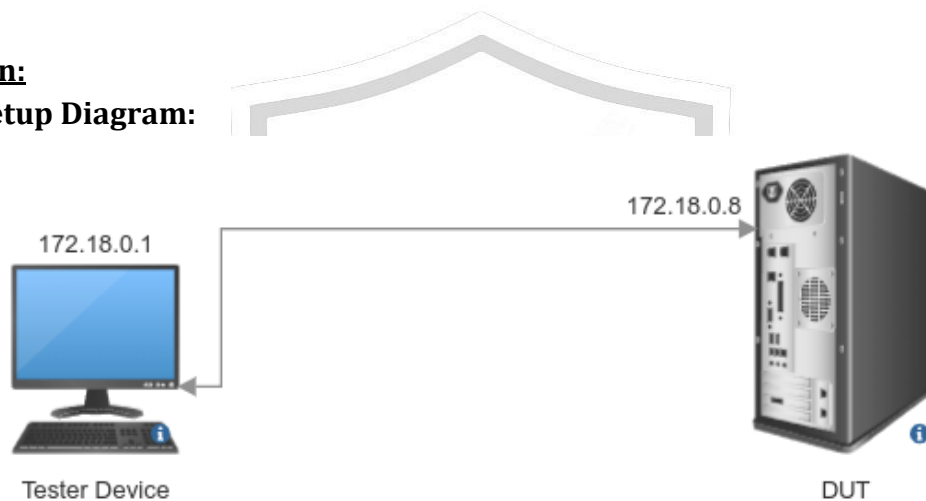
6. **Preconditions:**

- Instructions of how administrator user can view all existing accounts in the database are provided in the documentation accompanying the Network Product.
- All predefined accounts and their respective predefined or default passwords are identified in the documentation accompanying the Network Product.
- The location of keys will be provided in the documentation.
- Also, the commands for changing password and keys will be provided by the OEM.

7. **Test objective:** To ensure that predefined or default authentication attributes are deleted or disabled as defined

8. **Test Plan:**

8.1 **Test Setup Diagram:**



8.2 **Tools Used:** Command line

8.3 **Test Execution Steps**

- Check in documentation of the existence of any documented predefined account and what is the login password or if any cryptographic key for such accounts is preinstalled.
- Check for the location of the predefined password and keys from the OEM documentation.
- Follow the procedures given in the OEM document to remove the password and keys.

9. **Expected Results for Pass:**

- When login is attempted to any predefined account the user is automatically forced to change login password at first time login to the system.
- If there is no automatic password change enforced, then recommendation and clear instructions of how to manually change the password or how to create and reinstall a new cryptographic key exist in the documentation.

10. **Expected Format of Evidence:**

Evidence can be presented in the form of screenshot/screen-capture on how the network product prompts for password change at first login. Also extracts from product documentation with clear instructions of how to change any default password or cryptographic key.

11. **Test Execution:**

a. **Test Case Number:** 01

b. **Test Case Name:**

TC1_REMOVAL_OF_PREDEFINED_OR_DEFAULT_AUTHENTICATION_ATTRIBUTES

c. **Test Case Description:**

Default attributes like password and keys should be deleted for first access of the user.

d. **Execution Steps:**

The tester needs to check the documentation provided by the OEM and execute the procedure given.

Note: - the test execution steps will be different for different implementations. It should be provided by the OEM, what are the steps to change the password and where are the keys stored in the system.

e. **Test Observations:**

- **Case 1 (Success case):** when the user logs in for the first time, if prompt is shown to change the password and/or reinstall the key.
- **Case 2 (Failure case):** when this functionality is not there, there needs to be documentation providing to manually change the password and/or to reinstall the key by OEM.

f. **Evidence Provided:** Screenshot on how the network product prompts for password and cryptographic key change at first login

12. **Test Case Result:**

SL. No	OUTCOME OF RUNNING THE SCRIPT	PASS/FAIL	Remarks
1	TC1_REMOVAL_OF_PREDEFINED_OR_DEFAULT_AUTHENTICATION_ATTRIBUTES		

2.2.9 TSTP for Evaluation of Logout function

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.2.9
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. ITSAR Section No & Name: Section 2: Authentication Attribute Management

2. Security Requirement No & Name: 2.2.9 Logout function

3. Requirement Description:

The system shall have a function that allows a signed-in user to logout at any time. All processes under the logged-in user ID shall be terminated on logout. The network product shall be able to continue to operate without interactive sessions. Only for debugging purposes, processes under a logged-in user ID may be allowed to continue to run after detaching the interactive session.

[Reference: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.5.1]

4. DUT Confirmation Details:

- Use the command line interface to get details of the machine on which test is conducted.
- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

We have used **screen** utility for debugging purpose. To check if it is there in DUT, check with the command: - **screen -v**

```
vm2@vm2:~$ screen -v
Screen version 4.09.00 (GNU) 30-Jan-22
vm2@vm2:~$
```

Note: - For debugging purpose vendor should provide what is the utility they have implemented (Tester may obtain information from the vendor implementation provided in the documentation)

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2dea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

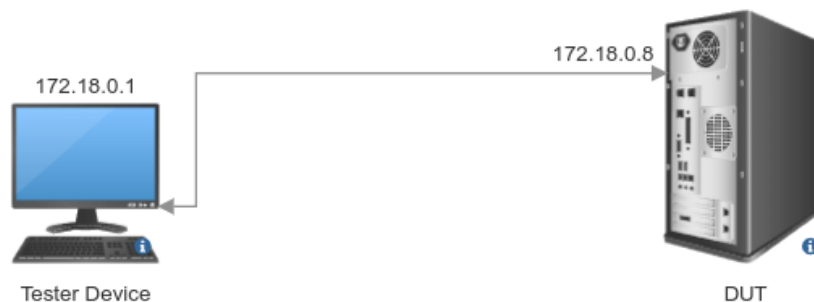
6. Preconditions:

- The manufacturer shall declare that it has a function that allows a signed in user to logout at any time.
- The tester has privileges to create a new account or use an existing account.
- The tester should have root access of DUT.

7. Test Objective: To ensure a signed in user can logout at any time

8. Test Plan:

8.1 Test Setup Diagram:



8.2 Tools Used:

8.3 Test Execution Steps:

- All the preconditions should meet.
- The tester should login to the network product.
- The tester creates a new account and logs in.
- Check the processes running for this user using the command: -
 - **ps -u <username> | wc -l**
- Any time after login the tester logout of the DUT, all the running processes should be deleted.
- After logout again check what is the process count using the same command.
- **For debugging case: -**

- We can use screen for this.
- Start a screen using the command: - **screen**
- Now login in this screen, start some process and logout of the account not screen.
- Check for the process count.

9. **Expected Results for Pass:** The tester can use a new account or an existing account to log into network product and logout network product after x minutes.

10. **Expected Format of Evidence:** Screenshots for the configurations used and for the results if the testcase passed or not.

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC1_PROTECTING_SESSION_LOGOUT

b. **Test Case Description:** To ensure a signed in user can logout at any time

c. **Execution steps:**

Note: - Please note that the actual steps and methods for checking these functionalities may vary depending on the specific system, software, or network product you are working with. It is essential to refer to the system documentation or consult with the development team for guidance on accessing and testing these features accurately.

- Check all the preconditions are met.
- The tester creates a new account.
- The tester tries to create a new account to login to SMF or uses existing account.
- After x minutes the tester tries to logout network product. (This x can be set by the tester.)

d. **Test Observations:**

Checking for the number of processes for the logged in user, using the command – **ps -u iith | wc -l**

```
vm2@vm2:/$ ps -u iith | wc -l
5
```

```
vm2@vm2:~$ ps -U iith u
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
iith      1916  0.0  0.1  19644  5272 pts/1    S   13:24   0:00 -bash
iith      1941  0.0  0.0  18972  1268 pts/1    S+  13:25   0:00 ping 8.8.8.8
iith      1995  0.0  0.1  19644  5240 pts/3    S   13:25   0:00 -bash
iith      2002  0.0  0.0  18972  1272 pts/3    S+  13:26   0:00 ping 8.8.8.8
vm2@vm2:~$
```

Checking for the number of processes after logout, using the command – **ps -u iith | wc -l**.

```
vm2@vm2:~$ ps -U iith u
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
vm2@vm2:~$
```

For debugging purpose, logging with screen

```
vn2@vm2:/$ ps -u iith | wc -l
3
vn2@vm2:/$
```

Even after logout the process is still there

```
vn2@vm2:/$ ps -U iith u
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
iith      167581 0.0  0.0   2888    980 pts/4    S    15:58   0:00 -sh
iith      167586 0.0  0.0  18972   1284 pts/4    S+   15:59   0:00 ping 8.8.8.8
vn2@vm2:/$ ps -u iith | wc -l
```

12. Test Case Result:

SL. No	OUTCOME OF RUNNING THE SCRIPT (CASE 1/CASE2)	PASS/FAIL	Remarks
--------	---	-----------	---------

1	TC1_PROTECTING_SESSION_LOGOUT		
---	-------------------------------	--	--



2.2.10 TSTP Report for Evaluation of Policy regarding consecutive failed login attempts

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.2.10
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. <ITSAR Section No & Name> Section 2: Authentication Attribute Management

2. <Security Requirement No & Name > 2.2.10 Policy regarding consecutive failed login attempts

3. <Requirement Description: >

- a) The maximum permissible number of consecutive failed user account login attempts should be configurable by the operator. The definition of the default value set at manufacturing time for the maximum number of failed user account login attempts shall be less than or equal to 8, typically 5. After the maximum permissible number of consecutive failed user account login attempts is exceeded by a user, there shall be a block delay in allowing the user to attempt login again. This block delay and the capability to set the period of the block delay, e.g., double the delay, or 5 minutes delay, or 10 minutes delay, after each login failure should be configurable by the operator. The default value set at manufacturing time for this delay shall be greater than or equal to 5 sec.
- b) If supported, infinite (permanent) locking of an account that has exceeded the maximum permissible number of consecutive failed user account login attempts should also be possible via configuration, with the exception of administrative accounts, which shall get only temporarily locked.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.4.5]

4. DUT Confirmation Details:

- Use the command line interface to get details of the machine on which test is conducted.
- Use command to get IP and Interfaces details
- Use command to get Application No/Version

- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal
```

5. DUT Configuration:

Check the OEM documentation to find all the login mechanisms.

Check the Operating system and its version for the DUT to know the commands for setting access permissions for data and application execution.

- **To get the hash of configuration file if the file is a ASCII text file.**
- Command - **sha256sum DUT_config.conf**
- **Digest Hash of Tested Configuration:**

- DUT_config.conf:-
19be54b42975f7f414c53a280373aa9466398781ecd77a2e9d090dbc6dbfdd8
- To get the hash of OS if using docker
- Command - **docker images --digests**
- **Digest Hash of OS:**
- DUT_IMAGE:
fd363f0e0d146c869d60649bcaf42e7008829d9d67a5dfdaf3b38ad24af7d53a

Most modern Unix-like operating systems include PAM or have similar authentication mechanisms that provide similar capabilities. In this TSTP PAM has been used, if there is any other authentication mechanisms present in DUT, that should be used.

DUT should have a method to configure the maximum permissible login attempts. All Linux distributions have Pluggable Authentication Module (PAM) for authentication.

If PAM is present, add and verify that the user “test” is added.

```
openstack@openstack:/etc/pam.d$ sudo useradd -m test
openstack@openstack:/etc/pam.d$ echo 'test:test' | sudo chpasswd
openstack@openstack:/etc/pam.d$ su - test
Password:
su: Authentication failure
```

Method may change based on the operating system.

6. Preconditions

- At least one user account has been created as per manufacturer's instructions.
- Directions of how to configure the maximum permissible number of consecutive failed user account login attempts and the default value of this number are identified in the documentation accompanying the Network Product. Default value shall be stated as well.
- Directions on how to configure the block delay in allowing a user to attempt to login again when the number of failed login attempts has exceeded the maximum number are identified in the documentation accompanying the Network Product. Default value of the delay shall be stated as well.
- Directions of how to optionally configure permanent locking for non-administrative accounts shall be stated as well.

7. **Test Objective:** To ensure that the policy regarding failed login attempts is adhered to.

8. Test Plan:

8.1 Number of Test Scenarios: 1

8.1.1 Test scenario for PAM.

This test scenario is when the DUT uses Pluggable Authentication Module (PAM) to manage authentication and access control.

8.2 Tools required: Linux CLI

8.3 **Execution steps:** As per the test case.

9. **Expected Results for pass:**

- User is blocked for a given amount of time, if they enter incorrect password for maximum number of times(configurable).
- If supported, non-admin users can also be permanently blocked after given number of incorrect attempts.

10. **Expected form of evidence:** Screenshots of Terminal

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC1_FAILED_LOGIN_ATTEMPTS

b. **Test Case Description:**

The tester checks default values, performs consecutive failed login attempts, and attempts an extra login beyond the maximum attempts. Subsequently, they test successful logins within and beyond the specified delay duration. The objective is to ensure proper account lockout behaviour and access control in the Linux environment.

c. **Test Execution:** The accredited evaluator's test lab is required to execute the following steps:

- 1) Check default values from precondition 2 and 3.
- 2) Perform consecutive failed login attempts for the user account until the default maximum number of precondition 2 is reached.
- 3) Attempt again one extra login, which fails again.
- 4) Attempt one extra login in less time than the default for the delay of precondition 3, using the correct credentials.
- 5) Attempt one extra login in more time than the default for the delay of precondition 3, using the correct credentials.

d. **Test Observations:**

1. Default values from preconditions 2 and 3 are in accordance with the requirement.

```
osboxes@osboxes:~$ cd /etc/pam.d/
osboxes@osboxes:/etc/pam.d$ ls
chfn          common-session-noninteractive  newusers      su
chpasswd      cron                          other          sudo
chsh          cups                          passwd         su-l
cinnamon-screensaver lightdm                       polkit-1       system-auth
common-account lightdm-autologin            ppp            systemd-user
common-auth    lightdm-greeter              runuser        vmtoolsd
common-password login                          runuser-l
common-session login_admin                  samba
```

```
osboxes@osboxes:/etc/pam.d$ cat lightdm
#%PAM-1.0
auth    requisite      pam_nologin.so
auth    sufficient      pam_succeed_if.so user ingroup nopasswdlogin
@include common-auth
auth    optional        pam_gnome_keyring.so
auth    optional        pam_kwallet.so
auth    optional        pam_kwallet5.so
@include common-account
session [success=ok ignore=ignore module_unknown=ignore default=bad] pam_selinux.so close
#session required      pam_loginuid.so
session required      pam_limits.so
@include common-session
session [success=ok ignore=ignore module_unknown=ignore default=bad] pam_selinux.so open
session optional      pam_gnome_keyring.so auto_start
session optional      pam_kwallet.so auto_start
session optional      pam_kwallet5.so auto_start
session required      pam_env.so readenv=1
session required      pam_env.so readenv=1 user_readenv=1 envfile=/etc/default/locale
@include common-password
auth    required        pam_faillock.so preauth
auth    [success=1 default=bad] pam_unix.so
auth    [default=die] pam_faillock.so authfail deny=3 unlock_time=60
```

Add the 'auth [default=die] pam_faillock.so authsucc deny=3 unlock_time=1 fail_interval=900' in lightdm file.

2. In execution step 4, the login attempt shall be rejected in all cases.

Securing Networks

```
osboxes@osboxes:/etc/pam.d$ su tester
osboxes@osboxes:/etc/pam.d$ su tester
The account is locked due to 3 failed logins.
(10 minutes left to unlock)
Password:
su: Authentication failure
osboxes@osboxes:/etc/pam.d$ 
su: Authentication failure
osboxes@osboxes:/etc/pam.d$ su tester
The account is locked due to 3 failed logins.
(10 minutes left to unlock)
Password: 
```

3. In execution step 5, the login attempt shall be accepted, it is verified that the user can login only at the last login attempt.

```
osboxes@osboxes:/etc/pam.d$ su tester
Password:
tester@osboxes:/etc/pam.d$
```

➤ **Test Case Number:**_02

a. **Test Case Name:** TC2_FAILED_LOGIN_ATTEMPTS

b. **Test Case Description:** An extra login attempt, resulting in failure, is made after reaching the maximum attempts. Subsequently, a login attempt is made after a longer delay than the default delay. The process is repeated for both normal users and users with administrative access, if supported, with permanent locking enabled for accounts exceeding the maximum failed attempts. The goal is to ensure proper account security measures for various user types in the system.

c. **Test Execution:** The accredited evaluator's test lab is required to execute the following steps:

- 1) Check default values from precondition 2 and 3.
- 2) Perform consecutive failed login attempts for the user account until the default maximum number of precondition 2 is reached.
- 3) Attempt again one extra login, which fails again.
- 4) Attempt one extra login in less time than the default for the delay of precondition 3, using the correct credentials.

5)

- If supported enable permanent locking of accounts exceeding the maximum permissible number of consecutive failed user account login attempts and repeat steps 1-4 for a normal user.
- If supported enable permanent locking of accounts exceeding the maximum permissible number of consecutive failed user account login attempts and repeat steps 1-4 for a user with administrative access rights.

d. **Test Observations:**

- 1) In execution step 5a it is verified that the user cannot login at any execution step.

Add the following line in /etc/pam.d/common-auth file.

```
auth [default=die] pam_faillock.so authfail
auth [success=1 default=bad] pam_unix.so
auth [default=die] pam_faillock.so authsucc deny=3 unlock_time=0
#For root
auth [success=1 default=bad] pam_unix.so
auth [default=die] pam_faillock.so authsucc deny=3 unlock_time=600
```

2) In execution step 5b it is verified that an administrator user can successfully login only at execution step 5b.

An account with root privileges is blocked temporarily and can be unlocked.

```
osboxes@osboxes:/etc/pam.d$ su admin
Password:
su: Authentication failure
osboxes@osboxes:/etc/pam.d$ su admin
Password:
su: Authentication failure
osboxes@osboxes:/etc/pam.d$ su admin
Password:
su: Authentication failure
osboxes@osboxes:/etc/pam.d$ su admin
The account is locked due to 3 failed logins.
(10 minutes left to unlock)
Password: 
```

Under test plan - Configuration will vary slightly depending on the type of login, for eg- ssh, or using su. So, verification should be done accordingly.

If same user can login via multiple interfaces, there should be test case for each of them.

12. Test Case Result:

To ensure that the policy regarding failed login attempts is adhered to. A testing report which will consist of the following information:

- Screenshot of the output of the terminal of the DUT.

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_FAILED_LOGIN_ATTEMPTS		
2	TC2_FAILED_LOGIN_ATTEMPTS		

2.3.1 TSTP Report for Evaluation of Secure Update

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.3.1
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 3: Software Security

2. **<Security Requirement No & Name >** 2.3.1 Secure Update

3. **<Requirement Description: >**

- a) Software package integrity shall be validated during software update stage.
- b) SYSTEM shall support software package integrity validation via cryptographic means, e.g. digital signature using Secure cryptographic controls prescribed in Table 1 of the latest document "Cryptographic Controls For Indian Telecom Security Assurance Requirements (ITSAR)" only. To this end, the network product has a list of public keys or certificates of authorised software sources, and uses the keys to verify that the software update is originated from only these sources.
- c) Tampered software shall not be executed or installed if integrity check fails.
- d) A security mechanism is required to guarantee that only authorized individuals can initiate and deploy a software update, and modify the list mentioned in bullet b.

Note: Code signing (valid and not time expired) is also allowed as an option in bullet b.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.3.5]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./system.out** (Used in IITH testbed to get SYSTEM version. Check with OEM manufacturer document for command specific to your SYSTEM)

Here we are assuming DUT to be SYSTEM, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SYSTEM_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

a. For .dsc files with dpkg

Verify that gpg is privileged command (To ensure no one but admin can import keys)

Command used: **ls -lrt /bin/gpg**

```
-rwxr--r-- 1 root root 1066992 Jul  4 2022 /bin/gpg
```

Command used: **gpg --version** (To get version information)

```
gpg (GnuPG) 2.2.19
libgcrypt 1.8.5
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: /home/pratik/.gnupg
Supported algorithms:
Pubkey: RSA, ELG, DSA, ECDH, ECDSA, EDDSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
        CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compression: Uncompressed, ZIP, ZLIB, BZIP2
```

Command used: **dpkg --version** (To get version information)

```
Debian 'dpkg' package management program version 1.19.7 (amd64).
This is free software; see the GNU General Public License version 2 or
later for copying conditions. There is NO warranty.
```

Verify that the public key used for signing the .dsc file is present in the system Command used:

gpg --verify <package_name.dsc> (Verify the software package)

```
gpg: Signature made Tuesday 26 October 2021 03:16:52 AM IST
gpg:         using RSA key 8F8D83B5270649A080648255EAF23F4A0BD34BF0
gpg:         issuer "mfo@canonical.com"
gpg: Good signature from "Mauricio Faria de Oliveira <mfo@canonical.com>" [unknown]
gpg:         aka "Mauricio Faria de Oliveira <mauricio.oliveira@canonical.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:         There is no indication that the signature belongs to the owner.
Primary key fingerprint: 8F8D 83B5 2706 49A0 8064  8255 EAF2 3F4A 0BD3 4BF0
```

Verify that the hashing and code signing algorithm used for the package are in accordance with “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only

Command used: **cat <package_name.dsc>** (Display .dsc file)

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512

Format: 3.0 (quilt)
Source: ufw
Binary: ufw
Architecture: all
Version: 0.36-6ubuntu1
Maintainer: Jamie Strandboge <jamie@ubuntu.co
```

Command used: **gpg --list-keys | grep -B 1 <finger_print>** (Display the signing algorithm)

```
pub   rsa4096 2018-07-16 [SC]
      8F8D83B5270649A080648255EAF23F4A0BD34BF0
```

b. Similarly for other package types and package managers

6. Precondition:

- A network product document containing information regarding software package integrity checks, including details of how the integrity check is carried out, where public keys or certificates of sources authorized to sign software packages are stored on the network

product and who these sources are, and what evidence is created to prove that the integrity check has been executed and what the result of the check was. Documentation which describes the installation procedure including how a user is authorized and authenticated to perform the installation process.

- A valid network product software load/package and one that is not-valid (or could be deemed to have been tampered with) are available.

7. **Test Objective:** To verify that the network product performs software integrity checks before the software is updated.

8. **Test Plan:**

8.1 Number of Test Case Scenarios

8.1.1. Test Scenario for .dsc files with dpkg:

- This test scenario verifies that integrity checks are carried out before software package is updated (Additional Test scenarios based on the OEM document)

8.2 Test Setup Diagram

8.3 Tools Used: NULL

8.4 Test Execution Steps

- Try to install updates from the valid software package (Case 1)
- Try to install updates from the invalid software package (Case 2)



9. **Expected Results for Pass:**

- (Case 1) Software is updated using the provided package
- (Case 2) Software is not updated using the provided package

10. **Expected Format of Evidence:** Screenshots of terminal

11. Test Execution:

- **Test Case Number:** 01
- a. **Test Case Name:** TC1_DSC_DPKG

b. **Test Case Description:** Tester tries to verify that integrity checks are carried out on the update package before any updates are carried out

c. **Execution Steps:**

- Given a genuine .dsc file for the software update, tester uses following command to source the .dsc file which will be used to create .deb file for applying updates (Case 1)

- `dpkg-source -x package_name.dsc`

```
dpkg-source: info: extracting ufw in ufw-0.36
dpkg-source: info: unpacking ufw_0.36.orig.tar.gz
dpkg-source: info: unpacking ufw_0.36-6ubuntu1.debian.tar.xz
dpkg-source: info: using patch list from debian/patches/series
dpkg-source: info: applying 0001-optimize-boot.patch
dpkg-source: info: applying 0002-fix-check-requirements.patch
dpkg-source: info: applying 0003-lp1838764.patch
dpkg-source: info: applying 0004-make-root-tests-chain-order-agnostic.patch
dpkg-source: info: applying 0005-use-only-python3.patch
dpkg-source: info: applying 0006-bug921680.patch
dpkg-source: info: applying 0007-bug921680-pt2.patch
dpkg-source: info: applying 0008-fix-check-requirements-again.patch
dpkg-source: info: applying 0009-empty-non-functioning-iptables-modules.patch
dpkg-source: info: applying 0010-add-other-firewall-checks.patch
dpkg-source: info: applying 0011-suppress-legacy-warnings-in-tests.patch
dpkg-source: info: applying 0012-set-default-policy-after-load.patch
dpkg-source: info: applying 0013-unconditionally-reload-with-delete.patch
```

- Given a malformed .dsc file for the software update, tester uses following command to source the .dsc file which will be used to create .deb file for applying updates (Case 2)

- `dpkg-source -x package_name.dsc`

```
gpgv: can't allocate lock for '/home/pratik/.gnupg/trustedkeys.gpg'
gpgv: Signature made Tuesday 26 October 2021 03:16:52 AM IST
gpgv: using RSA key 8F8D83B5270649A080648255EAF23F4A0BD34BF0
gpgv: issuer "mfo@canonical.com"
gpgv: BAD signature from "Mauricio Faria de Oliveira <mfo@canonical.com>"
dpkg-source: error: failed to verify signature on ./ufw_0.36-6ubuntu1.dsc
```

d. **Test Observation:**

- **(Case 1)** Tester verifies that the package has generated a source code using the .dsc file
- **(Case 2)** Tester verifies that integrity check fails and no source code has been generated

e. **Evidence Provided:** Screenshots of terminal

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_DSC_DPKG		

(Tester has to execute the test in similar fashion for other package types and package managers)

2.3.2 TSTP Report for Evaluation of Secure Upgrade

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.3.2
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 3: Software Security

2. **<Security Requirement No & Name >** 2.3.2 Secure Upgrade

3. **<Requirement Description: >**

- a) Software package integrity shall be validated during software upgrade stage.
- b) SMF shall support software package integrity validation via cryptographic means, e.g. digital signature using Secure cryptographic controls prescribed in Table 1 of the latest document "Cryptographic Controls For Indian Telecom Security Assurance Requirements (ITSAR)" only. To this end, the network product has a list of public keys or certificates of authorised software sources, and uses the keys to verify that the software update is originated from only these sources.
- c) Tampered software shall not be executed or installed if integrity check fails.
- d) A security mechanism is required to guarantee that only authorized individuals can initiate and deploy a software upgrade, and modify the list mentioned in bullet b.

Note: Code signing (valid and not time expired) is also allowed as an option in bullet b.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.3.5]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

Use command to get IP and Interfaces details

- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

a. For .dsc files with dpkg

Verify that gpg is privileged command (To ensure no one but admin can import keys)

Command used: **ls -lrt /bin/gpg**

```
-rwxr--r-- 1 root root 1066992 Jul  4 2022 /bin/gpg
```

Command used: **gpg --version** (To get version information)

```
gpg (GnuPG) 2.2.19
libgcrypt 1.8.5
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: /home/pratik/.gnupg
Supported algorithms:
Pubkey: RSA, ELG, DSA, ECDH, ECDSA, EDDSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
        CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compression: Uncompressed, ZIP, ZLIB, BZIP2
```

Command used: **dpkg --version** (To get version information)

```
Debian 'dpkg' package management program version 1.19.7 (amd64).
This is free software; see the GNU General Public License version 2 or
later for copying conditions. There is NO warranty.
```

Verify that the public key used for signing the .dsc file is present in the system

Command used: **gpg --verify <package_name.dsc>** (Verify the software package)

```
gpg: Signature made Tuesday 26 October 2021 03:16:52 AM IST
gpg:         using RSA key 8F8D83B5270649A080648255EAF23F4A0BD34BF0
gpg:         issuer "mfo@canonical.com"
gpg: Good signature from "Mauricio Faria de Oliveira <mfo@canonical.com>" [unknown]
gpg:         aka "Mauricio Faria de Oliveira <mauricio.oliveira@canonical.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:         There is no indication that the signature belongs to the owner.
Primary key fingerprint: 8F8D 83B5 2706 49A0 8064  8255 EAF2 3F4A 0BD3 4BF0
```

Verify that the hashing and code signing algorithm used for the package are in accordance with "Cryptographic Controls For Indian Telecom Security Assurance Requirements (ITSAR)" only

Command used: **cat <package_name.dsc>** (Display .dsc file)

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512

Format: 3.0 (quilt)
Source: ufw
Binary: ufw
Architecture: all
Version: 0.36-6ubuntu1
Maintainer: Jamie Strandboge <jamie@ubuntu.cc>
```

Command used: **gpg --list-keys | grep -B 1 <finger_print>** (Display the signing algorithm)

```
pub   rsa4096 2018-07-16 [SC]
      8F8D83B5270649A080648255EAF23F4A0BD34BF0
```

b. Similarly for other package types and package managers

6. Precondition:

- A network product document containing information regarding software package integrity checks, including details of how the integrity check is carried out, where public keys or certificates of sources authorized to sign software packages are stored on the network

product and who these sources are, and what evidence is created to prove that the integrity check has been executed and what the result of the check was. Documentation which describes the installation procedure including how a user is authorized and authenticated to perform the installation process.

- A valid network product software load/package and one that is not-valid (or could be deemed to have been tampered with) are available.

7. **Test Objective:** To verify that the network product performs software integrity checks before the software is updated

8. **Test Plan:**

8.1 Number of Test Case Scenarios

8.1.1. Test Scenario for .dsc files with dpkg:

- This test scenario verifies that integrity checks are carried out before software package is upgraded. (Additional Test scenarios based on the OEM document)

8.1.2. Test Setup Diagram

8.2 Tools Used: NULL

8.3 Test Execution Steps

- Try to install upgrades from the valid software package (Case 1)
- Try to install upgrades from the invalid software package (Case 2)



9. **Expected Results for Pass:**

- (Case 1) Software is upgraded using the provided package
- (Case 2) Software is not upgraded using the provided package

10. **Expected Format of Evidence:** Screenshots of terminal

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC1_DSC_DPKG

b. **Test Case Description:** Tester tries to verify that integrity checks are carried out on the upgrade package before any upgrades are carried out

c. **Execution Steps:**

- Given a genuine .dsc file for the software upgrade, tester uses following command to source the .dsc file which will be used to create .deb file for applying upgrades (Case 1)

○ `dpkg-source -x package_name.dsc`

```
dpkg-source: info: extracting ufw in ufw-0.36
dpkg-source: info: unpacking ufw_0.36.orig.tar.gz
dpkg-source: info: unpacking ufw_0.36-6ubuntu1.debian.tar.xz
dpkg-source: info: using patch list from debian/patches/series
dpkg-source: info: applying 0001-optimize-boot.patch
dpkg-source: info: applying 0002-fix-check-requirements.patch
dpkg-source: info: applying 0003-lp1838764.patch
dpkg-source: info: applying 0004-make-root-tests-chain-order-agnostic.patch
dpkg-source: info: applying 0005-use-only-python3.patch
dpkg-source: info: applying 0006-bug921680.patch
dpkg-source: info: applying 0007-bug921680-pt2.patch
dpkg-source: info: applying 0008-fix-check-requirements-again.patch
dpkg-source: info: applying 0009-empty-non-functioning-iptables-modules.patch
dpkg-source: info: applying 0010-add-other-firewall-checks.patch
dpkg-source: info: applying 0011-suppress-legacy-warnings-in-tests.patch
dpkg-source: info: applying 0012-set-default-policy-after-load.patch
dpkg-source: info: applying 0013-unconditionally-reload-with-delete.patch
```

- Given a malformed .dsc file for the software upgrade, tester uses following command to source the .dsc file which will be used to create .deb file for applying upgrades (Case 2)

○ `dpkg-source -x package_name.dsc`

```
gpgv: can't allocate lock for '/home/pratik/.gnupg/trustedkeys.gpg'
gpgv: Signature made Tuesday 26 October 2021 03:16:52 AM IST
gpgv: using RSA key 8F8D83B5270649A080648255EAF23F4A0BD34BF0
gpgv: issuer "mfo@canonical.com"
gpgv: BAD signature from "Mauricio Faria de Oliveira <mfo@canonical.com>"
dpkg-source: error: failed to verify signature on ./ufw_0.36-6ubuntu1.dsc
```

d. Test Observation:

- (Case 1) Tester verifies that the package has generated a source code using the .dsc file
- (Case 2) Tester verifies that integrity check fails and no source code has been generated

e. Evidence Provided: Screenshots of terminal

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_DSC_DPKG		

(Tester has to execute the test in similar fashion for other package types and package managers)

2.3.3 TSTP for evaluation of Source Code Security Assurance

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.3.3
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 3: Software Security
2. **<Security Requirement No & Name >** 2.3.3 Source Code Security Assurance
3. **<Requirement Description: >**

- a) OEM shall follow best security practices including secure coding for software development. Source code shall be made available either at TSTP premises or at the mutually agreed location for source code review by the designated TSTP. It may be supported by furnishing the Software Test Document (STD).
- b) Also, OEM shall submit the undertaking as below:
 - i. Industry standard best practices of secure coding have been followed during the entire software development life cycle of the SMF Software which includes OEM developed code, third party software and opensource code libraries used/embedded in the SMF.
 - ii. SMF software shall be free from CWE top 25, OWASP top 10 security vulnerabilities and OWASP top 10 API Security vulnerabilities as on the date of latest release of product or three months prior to the date of offer of product for testing, whichever is latest. For security weaknesses, vulnerabilities identified or discovered during the interim period, OEM shall give mitigation plan.
 - iii. The binaries for SMF and upgrades/updates thereafter generated from the source code are free from all known security vulnerabilities stated in bullet (ii) above.

4. DUT Confirmation Details:

- Use the command line interface to get details of the machine on which test is conducted.
- Use command to get IP and Interfaces details
- Use command to get Application No/Version

- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. **DUT Configuration:** Load the source code into tester's machine for analysis.

6. **Preconditions:**

- The source code must be available at the TSTP lab or mutually agreed location for testing. It shall be supported by Software Test Document (STD) from the vendor.
- Undertaking to be submitted by the vendor for above mentioned points in the requirement clause.

7. Test Objective: To Verify that:

- Best security practices including secure coding are followed for software development.
- software is free from CWE top 25 and OWASP top10 security weaknesses on the date of offer of product to designated TSTP for testing.
- The binaries for SMF and upgrades/updates thereafter generated from the source code are free from all known security vulnerabilities.

8. Test Plan:

a. Tools Used:

A suitable source code analysis tool having support for the programming language of the source code & having capabilities to detect vulnerabilities as per the requirement clause.

b. Test Execution Steps:

➤ **Case 1: Vendor provides source code to the TSTP lab:**

1. The tester should perform appropriate source code analysis practices (SAST) on the source code.
2. The tester must analyse the report generated by the source code analysis tool for identifying the vulnerabilities if present and highlight the same.

➤ **Case 2: verification of software test document submitted by vendor**

1. The tester must verify the software test documentation submitted by the vendor for any documented Security vulnerabilities.

9. Expected Result to Pass: The source code security analysis reports are as per the standard security specifications OR the Undertaking provided by the Vendor is in compliance with up-to-date secure software development practices.

10. Expected Format of Evidence: output of reports from source code analysis, and comments by tester.

11. Test Execution

Number of Test Cases: 2

➤ **Test Case Number: 1**

a. **Test Case Name:** TC_SOURCE_CODE_SEC_CHECK

b. **Test Case Description:**

Verify that:

1. Best security practices including secure coding are followed for software development.
2. Software is free from CWE top 25 and OWASP top10 security weaknesses on the date of offer of product to designated TSTP for testing.
3. The binaries for SMF and upgrades/updates thereafter generated from the source code are free from all known security vulnerabilities.

c. **Execution Steps:**

➤ **Case 1: Vendor provides source code to the TSTP lab:**

1. The tester should perform appropriate source code analysis practices (SAST) on the source code.
2. The tester must analyse the report generated by the source code analysis tool for identifying the vulnerabilities if present and highlight the same.

d. **Test Observations:**

The Reports generated by source code analysis tool do not report any vulnerability, especially present in CWE top 25 and OWASP top10 security weaknesses.

➤ **Test Case Number: 2**

e. **Test Case Name:** TC_VERIF_SOFT_TEST_DOC

f. **Test Case Description:**

To Verify the software test document provided by Vendor.

g. **Execution Steps:**

➤ **Case 2: verification of software test document submitted by vendor**

1. The tester must verify the software test documentation submitted by the vendor.

h. **Test Observations:**

The Software Test Documentation properly explains each test case performed on the software, and its results.

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_SOURCE_CODE_SEC_CHECK		
2.	TC_VERIF_SOFT_TEST_DOC		

2.3.4 TSTP Report for Known Malware and backdoor Check

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.3.4
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **ITSAR Section No & Name:** Section 3: Software Security
2. **Security Requirement No & Name:** 2.3.4 Known Malware and backdoor Check
3. **Requirement Description:** OEM shall submit an undertaking stating that SMF is free from all known malware and backdoors as on the date of offer of SMF to designated TSTP for testing and shall submit their internal Malware Test Document (MTD) of the SMF to the designated TSTP.

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal
```

5. **DUT Configuration:** Check the Operating system and its version for the DUT to know the commands for setting access permissions for data and application execution.

- **To get the hash of configuration file if the file is a ASCII text file.**

- Command - **sha256sum DUT_config.conf**

- **Digest Hash of Tested Configuration:**

- DUT_config.conf:-

19be54b42975f7f414c53a280373aa9466398781ecd77a2e9d090dbc6dbfdd8

- **To get the hash of OS if using docker**

- Command - **docker images --digests**

- **Digest Hash of OS:**

- DUT_IMAGE:

fd363f0e0d146c869d60649bc4f42e7008829d9d67a5dfdaf3b38ad24af7d53a

A known Malware and backdoor tool that has capabilities to support the given testing should be present. Following TSTP is developed using **clam AV** and **Rkhunter**. Both of them are open-source tools. Any of them can be used or any other tool can also be used, but documentation should be provided that the tool meets the need.

- Install Clam AV using: `sudo apt-get install clamav clamav-daemon`
- Install Rkhunter using: `sudo apt install rkhunter`

6. **Preconditions:**

- Database used should be updated. For e.g. in this TSTp, virus database for Clam AV should be updated. Steps -
 - First, stop the related processes to allow for the update to proceed.
 - `sudo systemctl stop clamav-freshclam`
 - run the updater application: `sudo freshclam`
- Start and enable the services, as below:
 - `sudo systemctl start clamav-freshclam`
 - `sudo systemctl enable clamav-freshclam`
- OEM shall submit document stating the abilities of the tool used.
- OEM shall submit the folders for which check was done.

7. **Test Objective:** Verify the OEM's claim of a malware-free and backdoor-free Authentication Management Function (SMF) by performing testing and reviewing the submitted undertaking and internal Malware Test Document (MTD).

8. **Test Plan:**

8.1 **Number of test scenarios/test cases: 1**

- 8.1.1 Test to check that OEM's claim of a malware-free and backdoor-free Authentication Management Function (SMF) by performing testing and reviewing the submitted undertaking and internal Malware Test Document (MTD).

8.2 **Test bed Diagram:**



8.3 **Tools used:** Command line of network function.

8.4 **Test case Execution:** The accredited evaluator's test lab is required to execute the following steps:

- The tester checks all the folders that needs to be scanned and checks the path.
- Based on the tool checking tool used, give path of the required folder else complete system.

- Generator test report showing the present risks.
- Compare the generated report with the report provided by vendor.

9. **Expected Results for Pass:** There should not be any known malware or backdoors present in SMF.

10. **Expected Format of Evidence:** Screenshot of the output terminal after running the tool or a report file (if given) by the tool.

11. **Test Execution:**

➤ **Test Case Number:** 1

a. **Test Case Name:** TC_MALWARE_BACKDOOR_CHECK

b. **Test Case Description:** The tester performs the malware and backdoor check and generates a report.

c. **Test Execution:** The accredited evaluator's test lab is required to execute the following steps:

Test scan the required folder that contains all the files, as below:

❏ `sudo clamscan -r <path>`

d. **Test Observations:**

➤ **Case 1:** Tester scans the required folder and checks the scan summary. There should not be any infected file.

```
supriya@supriya-Super-Server:~$ sudo clamscan -r Performance-Benchmarking/5G/src
/home/supriya/Performance-Benchmarking/5G/src/In5gtUpfServer.h: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtTelecom.o: OK
/home/supriya/Performance-Benchmarking/5G/src/.In5gtRan.cpp.swp: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtNssf.h: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtPcfCfg.txt: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtTun.cpp: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtRanUeEmulator.cpp: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtBsf.h: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtAusfServer.h: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtAmfServerNew.o: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtAmfCfg.cpp: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtUeKdf.cpp: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtGtpu.o: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtUpfCfg.txt: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtUdm.h: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtKdf.h: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtLogUtils.o: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtAusfCfg.cpp: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtPcf.cpp: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtSctpServerOld.cpp: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtAmfServer.cpp: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtSmfServer.h: OK
/home/supriya/Performance-Benchmarking/5G/src/In5gtSeaf.h: OK
```



```

----- SCAN SUMMARY -----
Known viruses: 92
Engine version: 0.103.6
Scanned directories: 7
Scanned files: 205
Infected files: 0
Data scanned: 66.15 MB
Data read: 99.89 MB (ratio 0.66:1)
Time: 2.262 sec (0 m 2 s)
Start Date: 2022:06:13 12:20:09
End Date: 2022:06:13 12:20:12
supriya@supriya-Super-Server:~$

```

Rkhunter (Rootkit Hunter): is an open-source Unix/Linux based scanner tool for Linux systems released under GPL that scans backdoors, rootkits, and local exploits on your systems.

```

root@supriya-Super-Server:/tmp/rkhunter-1.4.6# rkhunter --check --sk
[ Rootkit Hunter version 1.4.6 ]
Checking system commands...

Performing 'strings' command checks
  Checking 'strings' command [ OK ]

Performing 'shared libraries' checks
  Checking for preloading variables [ None found ]
  Checking for preloaded libraries [ None found ]
  Checking LD_LIBRARY_PATH variable [ Not found ]

Performing file properties checks
  Checking for prerequisites [ OK ]
  /usr/local/bin/rkhunter [ OK ]
  /usr/sbin/adduser [ Warning ]
  /usr/sbin/chroot [ OK ]
  /usr/sbin/cron [ OK ]
  /usr/sbin/depmod [ OK ]
  /usr/sbin/fsck [ OK ]
  /usr/sbin/groupadd [ OK ]
  /usr/sbin/groupdel [ OK ]
  /usr/sbin/groupmod [ OK ]
  /usr/sbin/grpck [ OK ]
  /usr/sbin/ifconfig [ OK ]
  /usr/sbin/init [ OK ]
  /usr/sbin/insmod [ OK ]
  /usr/sbin/ip [ OK ]
  /usr/sbin/lsmmod [ OK ]
  /usr/sbin/modinfo [ OK ]
  /usr/sbin/modprobe [ OK ]
  /usr/sbin/nologin [ OK ]
  /usr/sbin/pwck [ OK ]
  /usr/sbin/rmmod [ OK ]
  /usr/sbin/route [ OK ]
  /usr/sbin/rsyslogd [ OK ]
  /usr/sbin/runlevel [ OK ]
  /usr/sbin/sulogin [ OK ]
  /usr/sbin/sysctl [ OK ]
  /usr/sbin/useradd [ OK ]
  /usr/sbin/userdel [ OK ]
  /usr/sbin/usermod [ OK ]

```

```

Performing filesystem checks
  Checking /dev for suspicious file types [ Warning ]
  Checking for hidden files and directories [ Warning ]

```

```

System checks summary
=====

File properties checks...
  Files checked: 138
  Suspect files: 6

Rootkit checks...
  Rootkits checked : 478
  Possible rootkits: 3

Applications checks...
  All checks skipped

The system checks took: 1 minute and 45 seconds

All results have been written to the log file: /var/log/rkhunter.log

One or more warnings have been found while checking the system.
Please check the log file (/var/log/rkhunter.log)

```

- **Case 2:** Tester checks that the internal Malware Test Document (MTD) of the SMF submitted by vendor is same as the one generated. It shall contain all the malwares that have been found.

12. Test Case Result:

SL. No	OUTCOME OF RUNNING THE SCRIPT (CASE 1/CASE2)	PASS/FAIL	REMARKS
1	TC_MALWARE_BACKDOOR_CHECK		

Securing Networks

2.3.5 TSTP For No Unused Software

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.3.5
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 3 - Software Security
2. **<Security Requirement No & Name >**2.3.5 No unused software
3. **<Requirement Description: >** Software components or parts of software which are not needed for operation or functionality of the SMF shall not be present. Orphaned software components /packages shall not be present in SMF. OEM shall provide the list of software that are necessary for SMF's operation. In addition, OEM shall furnish an undertaking as "SMF does not contain Software that is not used in the functionality of SMF"

[Reference: TSDSI STD T1.3GPP 33.117 -16.7.0 V.1.0.0. Section 4.3.2.3]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5. **DUT Configuration:** Different Linux distributions use different package managers.

Here are some commonly used package managers for popular Linux distributions:

- Debian, Ubuntu, and related distributions: dpkg (Debian Package)
- Red Hat Enterprise Linux (RHEL), CentOS, Fedora: yum (Yellowdog Updater Modified) or dnf (Dandified YUM)
- Arch Linux and derivatives: pacman (Package Manager)
- SUSE Linux Enterprise, openSUSE: zypper
- Gentoo Linux: emerge
- Alpine Linux: apk

- Slackware Linux: pkgtool

For Linux,

To determine the package manager available use **which** command with the appropriate package manager:

Command used: `which dpkg`

```
priyansha@priyansha:~$ which dpkg
/usr/bin/dpkg
```

If the command returns a path, it means the package manager is installed. If there is no output, it means the package manager is not available.

Similarly, and so on for other package managers.

Note: We have made TSTP for dpkg package manager. The tester must check for other package managers too based on OEM documentation.

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:** A list of all available software and libraries and associated components containing at least the following information shall be included in the documentation accompanying the Network Product:

- name of the software / library;
- version of the software / library installed;
- list of dependencies and versions;
- any add-ons and functions;
- any special hardware/debugging ports;
- software support type;
- licensing information;
- brief description of their purpose.

For running the python script, the Vendors should fill in the csv file that contains the details of Software/Package Name, Software/Package version and Description.

Name of Software/Package/Library	Version of Software/Package/Library	Description
acl	2.2.53-6	
acpi-support	0.143	
acpid	1:2.0.32-1ubuntu1	
adduser	3.118ubuntu2	
adwaita-icon-theme	3.36.1-2ubuntu0.20.04.2	
aisleriot	1:3.22.9-1	
alsa-base	1.0.25+dfsg-0ubuntu5	
alsa-topology-conf	1.2.2-1	
alsa-ucm-conf	1.2.2-1ubuntu0.9	
alsa-utils	1.2.2-1ubuntu2.1	
amd64-microcode	3.20191218.1ubuntu1	
anacron	2.3-29	
apg	2.2.3.dfsg.1-5	
app-install-data-partner	19.04	
apparmor	2.13.3-7ubuntu5.1	
apport	2.20.11-0ubuntu27.21	
apport-gtk	2.20.11-0ubuntu27.21	
apport-symptoms	0.23	
appstream	0.12.10-2	
apt	2.0.6	

Figure: Sample CSV file to be filled by the vendors containing list of Software's

7. **Test Objective / Purpose:** To ensure that there is no unused software or associated components that might be installed in the network product which are not required for its operation or functionality.

8. **Test Plan:**

8.1 **Number of Test case scenarios: 1**

Test to check that there is no unused software or associated components that might be installed in the network product which are not required for its operation or functionality.

8.2 **Testbed Diagram:**



8.3 **Tools required:** Command Line Interface of the DUT, python.

8.4 **Test Execution Step:** The accredited evaluator's test lab is required to execute the following steps:

1. Verification of the compliance to the prerequisites:
 - a. Verification that the list of software / libraries and components is available in the documentation of the Network Product.
 - b. Validation that all entries in the list of software / libraries and components are meaningful and reasonably necessary for the operation of the Network Product class.
2. Identification of the software / libraries or components which are installed in the system using any suitable command line tools or any other suitable means of determination.

3. Validation that there are no entries in the list of software / libraries installed in the system apart from the ones that have been mentioned and deemed necessary for the operation of the network product in the attached documentation.
4. Based on his/her experience, the tester will check for known default example files for software installed on the system.

9. **Expected Results:**

- The report will contain the names and versions of the tool(s) used for finding out what software /libraries are installed in the system. The detailed report will contain the name and version information of all the software / libraries installed in the system generated by the tool.
- The list of all available software / libraries which has been deemed necessary for the operation of the network product by the vendor shall also be included as the test result. Any software / library not in the list of allowed software / libraries will be highlighted and brought out as a part of the report.
- There should be no unnecessary software / library installed in the network product except for the ones which are deemed necessary for its operation.
- There should be no more default example files for the installed software on the system.

10. **Expected Format of Evidence:** A testing report provided by the testing agency which will consist of the following information:

- The used tool(s) name and version information,
- Settings and configurations used
- the output pertaining to the test case performed and,
- the test results i.e. list of allowed and disallowed software

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC1_NO_UNUSED_SOFTWARE

b. **Test Case Description:** Test Needs to be conducted to check that there exists no unused software.

c. **Execution Steps:**

At OS-Level:

- Check if the documentation accompanying the Network Product includes a list of all available software and libraries and associated components.
- Use package management tools like *dpkg* to list installed packages.
- Compare the list of installed software and libraries obtained in the previous step with the list mentioned in the documentation accompanying the Network Product.
- Ensure that all the entries in the installed software / libraries match the ones deemed necessary for the operation of the network product.

- The comparison between installed softwares in DUT and the list of software provided by the vendor can be done manually.

To perform above steps using python script run below steps:

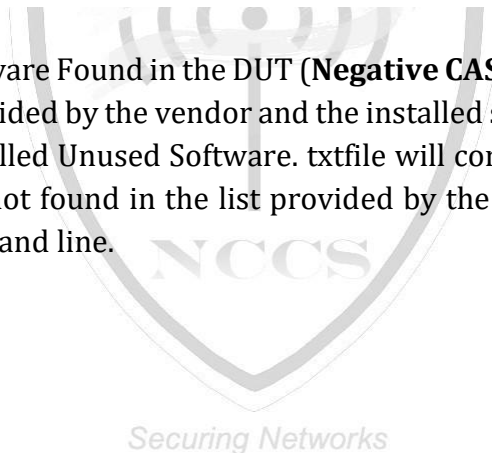
- Run the below command on DUT to save the result of the command in a text file.
dpkg -l > allsoftwares.txt
- Copy the file allsoftwares.txt to the tester device.
- Keep the vendor's list, copied file from DUT (which contains list of software installed in DUT) and python script in the same folder.
- Open the terminal where python script (no_unused_sw.py), list of software present in DUT (allsoftwares.txt) and list provided by vendor (Vendor_List.csv) is kept.
- Run the python script by using command: **python3 no_unused_sw.py**

At Application Level:

- Use appropriate Source Code Analyzer tool to verify that there is no unused software component present in the network product.
- Run the SCA tool against the source code to perform a comprehensive code analysis.
- The SCA tool will scan the codebase for potential issues, including unused software components.

d. Test Observation:

- **Case 1: Unused Software Found in the DUT (Negative CASE)** In case there is a mismatch between the list provided by the vendor and the installed software in DUT then a .txt file will be created. Installed Unused Software. txtfile will contain the name and version of Software/Packages not found in the list provided by the vendor. This list will also get printed on the command line.



```

This is the list of software not found in Vendor's List
anydesk:6.2.0
apt-config-icons:0.12.10-2
apt-config-icons-hidpi:0.12.10-2
apt-transport-https:2.0.9
apt-utils:2.0.9
aptdaemon:1.1.1+bzr982-0ubuntu32.3
aptdaemon-data:1.1.1+bzr982-0ubuntu32.3
apturl:0.5.2ubuntu19
apturl-common:0.5.2ubuntu19
arping:2.20-1
aspell:0.60.8-1ubuntu0.1
aspell-en:2018.04.16-0-1
at-spi2-core:2.36.0-2
avahi-autoipd:0.7-4ubuntu7.2
avahi-daemon:0.7-4ubuntu7.2
avahi-utils:0.7-4ubuntu7.2
baobab:3.34.0-1
base-files:11ubuntu5.7
base-passwd:3.5.47
bash:5.0-6ubuntu1.2
bash-completion:1:2.10-1ubuntu1
bc:1.07.1-2build1
bind9-dnsutils:1:9.16.1-0ubuntu2.12
bind9-host:1:9.16.1-0ubuntu2.12
bind9-libs:amd64:1:9.16.1-0ubuntu2.12
binutils:2.34-6ubuntu1.5
binutils-common:amd64:2.34-6ubuntu1.5
binutils-x86-64-linux-gnu:2.34-6ubuntu1.5
bluez:5.53-0ubuntu3.6
bluez-cups:5.53-0ubuntu3.6
bluez-obexd:5.53-0ubuntu3.6
bolt:0.9.1-2-ubuntu20.04.1
branding-ubuntu:0.10

```

Figure: installedUnusedSoftware.txt file is created in the same directory.

This list will contain unused software present in DUT that are required to be removed.

➤ **Case 2: No unused Software Found in the DUT (Positive CASE)**

When there is no mismatch between the list provided by the vendor and the installed software in the DUT list, then that means there is no unused software present that is required to be removed. In this case a statement in the command line interface will display output that indicates that there exists no unused software in the DUT. Please see the screenshot below for the reference.

All software and their version are correct

Figure: The command line will display the test above in case Test is passed

e. Evidence Provided:

A testing report which will consist of the following information:

- **Vendor_List.csv** file provided by the vendor which has a list of allowed software's.
- **installedUnusedSoftware.txt** file produced by script which has a list of unused software's.
- Screenshot of Settings and configurations used.
- Screenshot of the output of the terminal by running the python script.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_NO_UNUSED_SOFTWARE		

2.3.6 TSTP for Evaluation of Unnecessary Services Removal

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.3.6
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 3 - Software Security

2. **<Security Requirement No & Name >** 2.3.6 Unnecessary Services removal

3. **<Requirement Description: >**

SMF shall only run protocol handlers and services which are needed for its operation and which do not have any known security vulnerabilities. By default, all other ports and services will be permanently disabled. SMF Shall not support following services:

- FTP
- TFTP
- Telnet
- rlogin, RCP, RSH
- HTTP
- SNMPv1 and v2
- SSHv1
- TCP/UDP Small Servers (Echo, Charge, Discard and Daytime)
- Finger
- BOOTP server
- Discovery protocols (CDP, LLDP)
- IP Identification Service (Identd)
- PAD
- MOP

Any other protocols, services that are vulnerable are also to be permanently disabled. Full documentation of required protocols and services (communication matrix) of the SMF and their purpose needs to be provided by the OEM as prerequisite for the test case.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.3.2.1]

4. DUT Confirmation Details:

- Use the command line interface to get details of the machine on which test is conducted.
- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

The DUT must be reachable from the tester machine. It can be verified by sending ping (ICMP) requests from the tester machine to the DUT, using the following command.

`ping <DUT_IP_ADDRESS> -c 10`

```
siddhesh@stark99:~$ ping 192.168.127.177 -c 10
PING 192.168.127.177 (192.168.127.177) 56(84) bytes of data.
64 bytes from 192.168.127.177: icmp_seq=1 ttl=64 time=0.369 ms
64 bytes from 192.168.127.177: icmp_seq=2 ttl=64 time=0.331 ms
64 bytes from 192.168.127.177: icmp_seq=3 ttl=64 time=0.373 ms
64 bytes from 192.168.127.177: icmp_seq=4 ttl=64 time=0.351 ms
64 bytes from 192.168.127.177: icmp_seq=5 ttl=64 time=0.402 ms
64 bytes from 192.168.127.177: icmp_seq=6 ttl=64 time=0.419 ms
64 bytes from 192.168.127.177: icmp_seq=7 ttl=64 time=0.346 ms
64 bytes from 192.168.127.177: icmp_seq=8 ttl=64 time=0.356 ms
64 bytes from 192.168.127.177: icmp_seq=9 ttl=64 time=0.359 ms
64 bytes from 192.168.127.177: icmp_seq=10 ttl=64 time=0.373 ms

--- 192.168.127.177 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9218ms
rtt min/avg/max/mdev = 0.331/0.367/0.419/0.024 ms
siddhesh@stark99:~$
```

Verify that DUT firewall is disabled to allow network requests/traffic from the tester machine.

a. **For ufw (Uncomplicated Firewall):**

ufw can be disabled using the following command: `ufw status`

```
siddhesh@stark99:~$ sudo ufw status
Status: inactive
siddhesh@stark99:~$
```

Use `sudo ufw status` if disabling the firewall requires user authentication

NOTE: If the DUT supports a firewall service other than the one mentioned above (ufw), then the tester must follow steps required to check status of the respective firewall.

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. Preconditions:

A list of all required network protocols and services containing at least the following information shall be included in the documentation accompanying the Network Product, provided by the Vendor:

- protocol handlers and services needed for the operation of network product;
- their open ports and associated services;
- and a description of their purposes.

The tester machine must be equipped with a port scanning tool, for scanning the ports of the DUT to get information required the status of the ports and the protocols/services running on them.

The firewall on DUT must be disabled for allowing network traffic/requests from the tester machine.

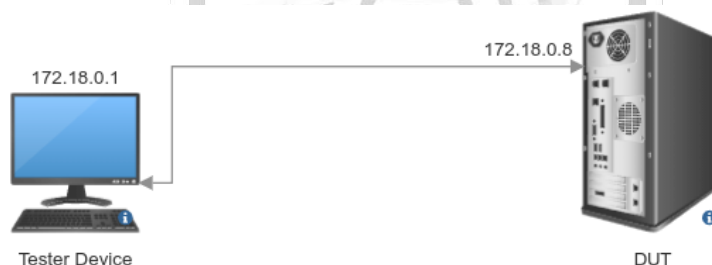
7. Test Objective: To ensure that on all network interfaces, there are no unnecessary and unsecure services or protocols that might be running.

8. Test Plan:

8.1 Number of Test Scenarios: 1

Test to check that there are no unnecessary and unsecure services or protocols running on all interfaces

8.2 Test Setup Diagram



8.3 Tools Used: Nmap (a Port scanning Tool)

8.4 Test Execution Steps:

- The tester must verify for the compliance to the pre-requisites:
 - a. Verification that the list of available network services and protocol handlers is available in the documentation of the Network Product.
 - b. Validation that the entries in the list do not contain the prohibited protocols/services from the list mentioned above or any other protocol that is known to have security vulnerabilities, and it should be reasonably necessary for the operation of the Network Product class.
- The tester must identify the protocols and services running on the DUT using an appropriate port scanning tool.
- The tester must validate that there are no entries in the list of network services and handlers apart from the ones that have been mentioned and deemed necessary for the operation of the Network Product in the attached documentation.
- The tester shall reboot the network product and re-execute execution steps 2 and 3 without further configuration.

9. Expected Result for Pass:

The Network product must only run protocol/services which are needed for its operations and which do not have any security vulnerabilities.

10. Expected Format of Evidence:

file/report containing the information regarding the status of the ports and the protocol/services running on them, along with the comments on which protocols/services among them are necessary and which are not.

11. Test Plan:

Test Execution:

➤ **Test Case Number:** 01

a. Test Case Name: TC_NO_UNNECESSARY_SERVICE

b. Test Case Description: Ensuring that on all network interfaces, there are no unnecessary and unsecure services or protocols that might be running.

c. Execution Steps:

- The tester must verify for the compliance to the pre-requisites:
 - a. Verification that the list of available network services and protocol handlers is available in the documentation of the Network Product.
 - b. Validation that the entries in the list do not contain the prohibited protocols/services from the list mentioned above or any other protocol that is known to have security vulnerabilities, and it should be reasonably necessary for the operation of the Network Product class.
- The tester must identify the protocols and services running on the DUT using an appropriate port scanning tool.
- We used the nmap port scanning tool for identifying the above information using the following command:
Securing Networks
`nmap -p <port_range> <Address_of_DUT> -v`
The -v flag is used to print logs while execution

```
siddhesh@stark99:~$ nmap -p 1-1024 127.0.0.1 -v
Starting Nmap 7.80 ( https://nmap.org ) at 2023-06-13 19:15 IST
Initiating Ping Scan at 19:15
Scanning 127.0.0.1 [2 ports]
Completed Ping Scan at 19:15, 0.00s elapsed (1 total hosts)
Initiating Connect Scan at 19:15
Scanning localhost (127.0.0.1) [1024 ports]
Discovered open port 80/tcp on 127.0.0.1
Discovered open port 22/tcp on 127.0.0.1
Discovered open port 631/tcp on 127.0.0.1
Completed Connect Scan at 19:15, 0.02s elapsed (1024 total ports)
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000083s latency).
Not shown: 1021 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
631/tcp   open  ipp

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.05 seconds
```

Also, the command `nmap -p- <Address_of_DUT>`

Can be used to scan all ports on the DUT, without the need to specify the entire range (1-65535)

```
siddhesh@stark99:~$ nmap -p- 127.0.0.1 -v
Starting Nmap 7.80 ( https://nmap.org ) at 2023-06-13 19:23 IST
Initiating Ping Scan at 19:23
Scanning 127.0.0.1 [2 ports]
Completed Ping Scan at 19:23, 0.00s elapsed (1 total hosts)
Initiating Connect Scan at 19:23
Scanning localhost (127.0.0.1) [65535 ports]
Discovered open port 80/tcp on 127.0.0.1
Discovered open port 22/tcp on 127.0.0.1
Discovered open port 12001/tcp on 127.0.0.1
Discovered open port 20628/tcp on 127.0.0.1
Discovered open port 17600/tcp on 127.0.0.1
Discovered open port 17500/tcp on 127.0.0.1
Discovered open port 17603/tcp on 127.0.0.1
Discovered open port 631/tcp on 127.0.0.1
Discovered open port 6655/tcp on 127.0.0.1
Discovered open port 7001/tcp on 127.0.0.1
Discovered open port 4000/tcp on 127.0.0.1
Discovered open port 25001/tcp on 127.0.0.1
Discovered open port 7071/tcp on 127.0.0.1
Discovered open port 6657/tcp on 127.0.0.1
Discovered open port 6654/tcp on 127.0.0.1
Discovered open port 6656/tcp on 127.0.0.1
Discovered open port 40205/tcp on 127.0.0.1
Completed Connect Scan at 19:23, 1.23s elapsed (65535 total ports)
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00010s latency).
Not shown: 65518 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
631/tcp   open  ipp
4000/tcp  open  remoteanything
6654/tcp  open  unknown
6655/tcp  open  pcs-sf-ui-man
6656/tcp  open  engmsg
6657/tcp  open  palcom-disc
7001/tcp  open  afs3-callback
7071/tcp  open  lwg1
12001/tcp open  entextnetwk
17500/tcp open  db-lsp
17600/tcp open  unknown
17603/tcp open  unknown
20628/tcp open  unknown
25001/tcp open  icl-twobase2
40205/tcp open  unknown

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 1.28 seconds
```

NOTE: the tester can use any other port scanning tool to scan all the ports on the DUT to get the list and status of ports and the protocols/services running on them.

- The tester must validate that there are no entries in the list of network services and handlers apart from the ones that have been mentioned and deemed necessary for the operation of the Network Product in the attached documentation.
- The tester shall reboot the network product and re-execute execution steps 2 and 3 without further configuration.

12. **Test Observations:**

The network product does not run any unnecessary or unsecure protocols or services.

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_UNNECESSARY_SERVICE		



2.3.7 TSTP for Evaluation of Restricting System Boot Source

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.3.7
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 3: Software Security
2. **<Security Requirement No & Name >** 2.3.7 Restricting System Boot Source.
3. **<Requirement Description: >** SMF can boot only from the memory devices intended for this purpose.
4. **DUT Confirmation Details:**
 - Use the command line interface to get details of the machine on which test is conducted.
 - Use command to get Application No/Version
 - Use command to get OS Version/No

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Pre-Conditions:** An OEM/Vendor document which contains information regarding the firmware (UEFI/BIOS) access mechanism supported by the product, the memory devices from which the network product can boot and the authentication credentials for accessing the firmware.

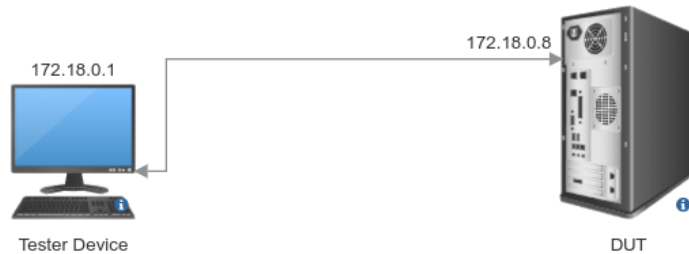
7. **Test Objective:** To verify that the Network Product can only boot from the memory devices that are mentioned in the documentation provided by the OEM, and access to the firmware is only possible with the correct authentication.

8. Test Plan:

8.1 Number of Test case scenarios: 1

Test to check that Network Product can only boot from the memory devices that are mentioned in the documentation provided by the OEM, and access to the firmware is only possible with the correct authentication.

8.2 Test Setup Diagram



8.3 **Tools Used:-** firmware of the DUT (generally BIOS/UEFI).

8.4 **Test Execution Steps:**

1. The Tester accesses the firmware (BIOS/UEFI) of the DUT, by referring to the instructions provided in the OEM Documentation.
2. The tester verifies that the access to the firmware is possible only through correct authentication (for e.g. Password based authentication).
3. The tester checks the firmware (BIOS/UEFI) boot configuration and verifies that the network product is configured to boot from memory devices declared in the network product document only.

9. **Expected Results for Pass:** The Network Product firmware (BIOS/UEFI) is accessible only after successful authentication and the Network Product is allowed to boot only from the allowed memory devices (as specified in the OEM Document).

10. **Expected Format of Evidence:**

- Screenshots of successful/unsuccessful firmware (UEFI/BIOS) authentication.
- Screenshots of firmware (UEFI/BIOS) boot configuration specifying the allowed memory devices for boot and their order.

11. **Test Execution**

➤ **Test Case Number:** 01

a) Test Case Name: TC_BOOT_INT_MEM

b) Test Case Description: To verify that the Network product can boot only from the memory devices intended for this purpose. (e.g., not from external memory like a USB key).

c) Tools Used: firmware of the DUT (BIOS/UEFI)

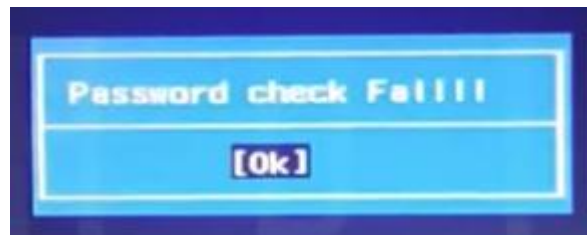
d) Execution Steps:

1. The tester should verify that UEFI Boot Mode is enabled for the VM. (In case of a containerised environment, the test to be performed directly on the Host Machine).
2. The Tester accesses the firmware (BIOS/UEFI) of the DUT, by referring to the instructions provided in the OEM Documentation.

3. The tester verifies that the access to the firmware is possible only through correct authentication (for e.g. Password based authentication).
For e.g. Password based authentication screen for accessing the firmware (UEFI/BIOS).



Unsuccessful authentication:



Successful Authentication (redirected to firmware (UEFI/BIOS) main page):



4. The tester checks the firmware (BIOS/UEFI) boot configuration and verifies that the network product is configured to boot from memory devices declared in the network product document only.
Here, we consider the scenario wherein booting the Network product is only allowed via Hard disk.
- **Positive case:** The firmware (UEFI/BIOS) allows booting the network product only from the Hard disk.



- **Negative case:** The firmware (UEFI/BIOS) allows booting the network product from memory devices other than the Hard disk (USB key).



Note: The tester must check that no other memory device than the ones mentioned in the OEM/Vendor document are present in the boot order configuration.

e) Test Observations:

- It should be ensured that the access to firmware (UEFI/BIOS) is protected via authentication.
- It should be ensured that the boot configuration in the firmware (UEFI/BIOS) should only allow booting the network product from memory devices intended for the purpose (as specified in the OEM document).

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_BOOT_INT_MEM		

2.3.8 TSTP for Evaluation of Secure Time Synchronizations

Session Management Function ITSAR

ITSAR No: ITSAR111092401

Clause no: 2.3.8

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<TSTP Document ID:>

<Applicant Name:> Ex: XYZ

<Application Number>

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 3: Software Security

2. **<Security Requirement No & Name >** 2.3.8 Secure Time Synchronization

3. **<Requirement Description: >**

- i. SMF shall establish secure communication channel with the NTP/PTP server.
- ii. SMF shall establish secure communication channel strictly using Secure cryptographic controls prescribed in Table1 of the latest document "Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)" with NTP/PTP server.
- iii. SMF shall generate audit logs for all changes to time settings.

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

The DUT **must** support secure versions of NTP/PTP for secure time synchronization. For secure communication over NTP, Check if the NTPsec is installed on the DUT, along with its version (1.1.8+).

```
siddhesh@siddhesh:/$ /usr/local/sbin/ntpd -V
ntpd ntpsec-1.2.2+69-gc9c7f716e
siddhesh@siddhesh:/$
```

NOTE: For Security over PTP, there are currently no open-source implementations available that provide a secure PTP communication. The TSTP can check if any secure implementation of PTP which is proprietary is available, and if the DUT supports it, and if not then suggest the

Vendor to provide support for the same in the DUT. If no such secure implementation is available, then it is advised not to use PTP for time synchronization.

5. **DUT Configuration:** For security over NTP, the NTPSec client must be configured on the DUT to send and receive NTP message packets in a secure manner.

Sample NTPsec config:

```
root@siddhesh:/etc# cat ntp.conf
# Example 1: unsecured NTP
#server ntp1.glypnod.com iburst minpoll 3 maxpoll 6

# Example 2: NTS-secured NTP (default NTS-KE port (123); using certificate pool of the #operating system)
server ntp1.glypnod.com iburst minpoll 3 maxpoll 6 nts

# Example 3: NTS-secured NTP (custom certificate and NTS-KE port)
#server nts3-e.ostfalia.de:443 iburst minpoll 3 maxpoll 6 nts ca /var/lib/ntp/certs/rootCaBundle.pem

# Example 4: NTS-secured NTP (skip DNS check)
#server nts3-e.ostfalia.de:443 iburst minpoll 3 maxpoll 6 nts ca /var/lib/ntp/certs/rootCaBundle.pem noval

# optional: allows a fast frequency error correction on startup
driftfile /var/lib/ntp/ntp.drift

# optional: collect statistics
statsdir /var/log/ntpstats
statistics loopstats peerstats clockstats rawstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable
filegen rawstats file rawstats type day enable

# optional: logging
logfile /var/log/ntp.log
logconfig =syncall +clockall +peerall +sysall
root@siddhesh:/etc#
```

- To get the hash of configuration file if the file is a ASCII text file sha256sum DUT_config.conf
- Digest Hash of Tested Configuration:
- DUT_config.conf:-
19be54b42975f7f414c53a280373aa9466398781ecd77a2e9d090dbc6dbfdd8
- To get the hash of OS if using docker
- docker images --digests
- Digest Hash of OS:
- DUT_IMAGE:
fd363f0e0d146c869d60649bcaf42e7008829d9d67a5dfdaf3b38ad24af7d53a

6. **Preconditions:**

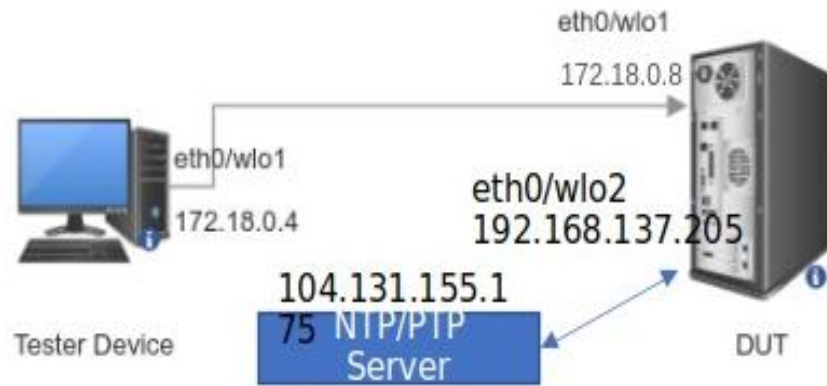
- The Vendor must specify the path to and configuration details of the NTP(NTPsec)/PTP client.
- The tester must be given privileges to run the the NTS(NTPsec)/PTP client with the provided configurations.
- The Vendor must specify the path at which the log files are stored for the respective protocol.

7. **Test Objective:** To verify that the NTP/PTP is used for time synchronization and the NTP/PTP messages are exchanged over a secure communication channel, and the changes in the time settings are logged appropriately.

8. **Test Plan:**

8.1 **number of test scenarios/test cases**

Test scenario for Secure Time Synchronization.



8.2 **TEST BED DIAGRAM:**

8.3 **Tools Required:** Wireshark, terminal of the DUT.

8.4 **Test Execution Steps**

- Start the NTS (NTPsec)/PTP service on the DUT with the appropriate configuration.
- Start the Wireshark on the DUT Interface through which NTS (NTPsec)/PTP request/response are being sent/received.
- Capture the NTS(NTPsec)/PTP packets and check for appropriate fields in the packets that ensure a secure communication channel. The cryptographic algorithms used must be in compliance with Table1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)”
- To make changes in the time settings, the tester should make changes in the configuration file of NTS(NTPsec)/PTP by trying different configuration parameter combinations.
- Access the logs from the appropriate location to check if the changes in the time settings are logged by the DUT.

9. **Expected Results for Pass:** The DUT uses NTP/PTP for time synchronization, and it is able to establish secure communication channel for NTP/PTP based time synchronization and proper logs are maintained for changes in time settings

10. **Expected Format of Evidence:**

- Screenshots of Wireshark and Pcap files capturing the NTS(NTPsec)/PTP packets with appropriate fields that provide security.
- screenshots of audit logs for NTS(NTPsec)/PTP on change in the time settings.

11. **Test Execution**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC_SEC_TIME_SYNC

b. **Test Case Description:** To verify that the NTP/PTP messages are sent over a secure communication channel.

To verify that audit logs are generated for change in time settings.

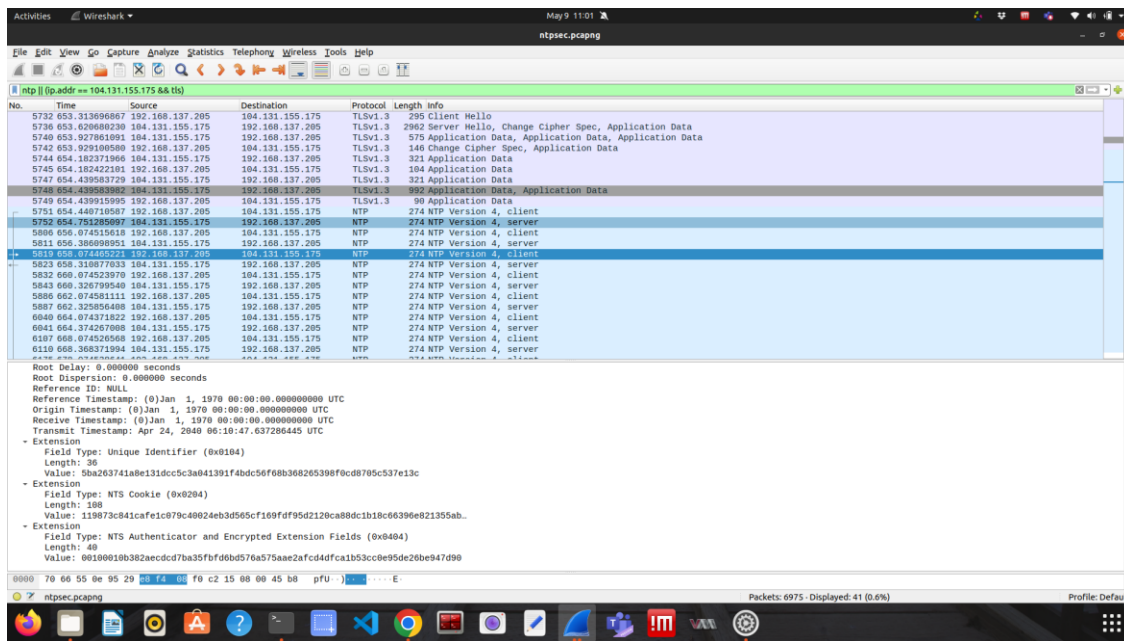
c. **Execution Steps:**

NOTE: The screenshots & test commands for terminal are provided assuming the DUT and/or tester machine (with Ubuntu 20.04 OS installed) used for performing the test. They might differ for other OS, and tester must consider the ones for the respective OS installed in the DUT and/or tester machine

- Start the NTS (NTPsec)/PTP service on the DUT with the appropriate configuration.
- Start the Wireshark on the DUT Interface through which NTS (NTPsec)/PTP request/response are being sent/captures.
- Capture the NTP/PTP packets and check for appropriate fields in the packets that ensure a secure communication channel. The cryptographic algorithms used must be in compliance with Table1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)”

For NTS (NTPsec), the Wireshark trace must show the NTS-KE followed by NTPv4 Packets with the following extensions:

1. Unique Identifier
2. NTS Cookie
3. NTS authenticator and encrypted extension fields.



- Make changes in the configuration file of NTS(NTPsec)/PTP for making changes in the time settings.
- Check if the changes are logged by the DUT in a log file, by accessing log files at the location as provided by the vendor.

In our case, For NTS(NTPsec), the changes are logged in the /var/log/ntp.log & /var/log/syslog, the below screenshot shows initialization of ntpsec after starting the ntpd service.

```
2023-05-08T17:04:39 ntpd[49331]: INIT: ntpd ntpsec-1.2.2+69-gc9c7f716e: Starting
2023-05-08T17:04:39 ntpd[49331]: INIT: Command line: /usr/local/sbin/ntpd -g -N -u ntp:ntp
2023-05-08T17:04:39 ntpd[49331]: INIT: Using SO_TIMESTAMPNS(ns)
2023-05-08T17:04:39 ntpd[49331]: IO: Listen and drop on 0 v6wildcard [::]:123
2023-05-08T17:04:39 ntpd[49331]: IO: Listen and drop on 1 v4wildcard 0.0.0.0:123
2023-05-08T17:04:39 ntpd[49331]: IO: Listen normally on 2 lo 127.0.0.1:123
2023-05-08T17:04:39 ntpd[49331]: IO: Listen normally on 3 wlo1 192.168.137.205:123
2023-05-08T17:04:39 ntpd[49331]: IO: Listen normally on 4 lo [::1]:123
2023-05-08T17:04:39 ntpd[49331]: IO: Listen normally on 5 wlo1 [fe80::85ab:1e98:81c9:7547%2]:123
2023-05-08T17:04:39 ntpd[49331]: IO: Listening on routing socket on fd #22 for interface updates
2023-05-08T17:04:39 ntpd[49331]: PROTO: ntp1.glypnod.com c011 81 mobilize assoc 17767
2023-05-08T17:04:39 ntpd[49331]: INIT: MRU 10922 entries, 13 hash bits, 65536 bytes
2023-05-08T17:04:39 ntpd[49331]: PROTO: 0.0.0.0 c01d 0d kern kernel time sync enabled
2023-05-08T17:04:39 ntpd[49331]: PROTO: 0.0.0.0 c012 02 freq_set kernel 26.255000 PPM
2023-05-08T17:04:39 ntpd[49331]: PROTO: 0.0.0.0 c016 06 restart
2023-05-08T17:04:39 ntpd[49331]: INIT: OpenSSL 1.1.1f 31 Mar 2020, 1010106f
2023-05-08T17:04:39 ntpd[49331]: NTSc: Using system default root certificates.
2023-05-08T17:04:40 ntpd[49331]: DNS: dns_probe: ntp1.glypnod.com, cast_flags:1, flags:21901
2023-05-08T17:04:40 ntpd[49331]: NTSc: DNS lookup of ntp1.glypnod.com took 0.002 sec
2023-05-08T17:04:40 ntpd[49331]: NTSc: connecting to ntp1.glypnod.com:4460 => 104.131.155.175:4460
2023-05-08T17:04:41 ntpd[49331]: NTSc: set cert host: ntp1.glypnod.com
2023-05-08T17:04:41 ntpd[49331]: NTSc: Using TLSv1.3, TLS_AES_256_GCM_SHA384 (256)
2023-05-08T17:04:41 ntpd[49331]: NTSc: certificate subject name: /CN=ntp.glypnod.com
2023-05-08T17:04:41 ntpd[49331]: NTSc: certificate issuer name: /C=US/O=Let's Encrypt/CN=R3
2023-05-08T17:04:41 ntpd[49331]: NTSc: SAN:DNS ntp.glypnod.com, ntp1.glypnod.com
2023-05-08T17:04:41 ntpd[49331]: NTSc: certificate is valid.
2023-05-08T17:04:41 ntpd[49331]: NTSc: Good ALPN from ntp1.glypnod.com
2023-05-08T17:04:42 ntpd[49331]: NTSc: read 880 bytes
2023-05-08T17:04:42 ntpd[49331]: NTSc: Got 8 cookies, length 104, aead=15.
2023-05-08T17:04:42 ntpd[49331]: NTSc: NTS-KE req to ntp1.glypnod.com took 1.362 sec, OK
2023-05-08T17:04:42 ntpd[49331]: DNS: dns_check: processing ntp1.glypnod.com, 1, 21901
2023-05-08T17:04:42 ntpd[49331]: DNS: Server taking: 104.131.155.175
2023-05-08T17:04:42 ntpd[49331]: DNS: dns_take_status: ntp1.glypnod.com=>good, 0
2023-05-08T17:04:42 ntpd[49331]: PROTO: 104.131.155.175 e014 84 reachable
2023-05-08T17:04:48 ntpd[49331]: PROTO: 104.131.155.175 f01a 8a sys_peer
```

The below screenshot shows the logs regarding the operation on/by ntpd service.

```

root@siddhesh:/var/log# less syslog | grep ntp
May 9 00:48:00 siddhesh NetworkManager[1008]: <info> [1683573480.9712] dhcp4 (wlo1): option requested_ntp_servers => '1'
May 9 10:35:15 siddhesh kernel: [ 0.07315] Mou: ntp: cache hash table entries: 16384 (order: 5, 131072 bytes, linear)
May 9 10:35:15 siddhesh kernel: [ 5.186668] audit: type=1400 audit(1683608715.36477): apparmor="STATUS" operation="profile_load" profile="t
May 9 10:35:15 siddhesh ntpd[1266]: 2023-05-09T10:35:15 ntpd[1266]: INIT: ntpd ntpsec-1.2.2+69-gc9c7f716e: Starting
May 9 10:35:15 siddhesh ntpd[1266]: 2023-05-09T10:35:15 ntpd[1266]: INIT: Command line: /usr/local/sbin/ntpd -g -N -u ntp:ntp
May 9 10:35:15 siddhesh ntpd[1266]: INIT: ntpd ntpsec-1.2.2+69-gc9c7f716e: Starting
May 9 10:35:15 siddhesh ntpd[1266]: INIT: Command line: /usr/local/sbin/ntpd -g -N -u ntp:ntp
May 9 10:35:15 siddhesh ntpd[1272]: INIT: precision = 0.082 usec (-23)
May 9 10:35:15 siddhesh ntpd[1272]: INIT: successfully locked into RAM
May 9 10:35:15 siddhesh ntpd[1272]: CONFIG: readconf: parsing file: /etc/ntp.conf
May 9 10:35:15 siddhesh ntpd[1272]: LOG: switching logging to file /var/log/ntp.log
May 9 10:35:16 siddhesh systemd[1]: tmp-syscheck[x2dmou: ntpoint(x2d579703519.mount: Succeeded.
May 9 10:35:16 siddhesh systemd[1508]: tmp-syscheck[x2dmou: ntpoint(x2d579703519.mount: Succeeded.
May 9 10:35:19 siddhesh NetworkManager[976]: <info> [1683608719.8790] dhcp4 (wlo1): option requested_ntp_servers => '1'
May 9 10:39:02 siddhesh NetworkManager[976]: <info> [1683608942.7252] dhcp4 (wlo1): option requested_ntp_servers => '1'
May 9 11:56:02 siddhesh ntpd[275571]: INIT: ntpd ntpsec-1.2.2+69-gc9c7f716e: Starting
May 9 11:56:02 siddhesh ntpd[275571]: INIT: Command line: /usr/local/sbin/ntpd
May 9 11:56:02 siddhesh ntpd[27558]: INIT: precision = 0.145 usec (-23)
May 9 11:56:02 siddhesh ntpd[27558]: INIT: successfully locked into RAM
May 9 11:56:02 siddhesh ntpd[27558]: CONFIG: readconf: parsing file: /etc/ntp.conf
May 9 11:57:48 siddhesh ntpd[27558]: LOG: switching logging to file /var/log/ntp.log
May 9 11:57:48 siddhesh systemd[1]: ntpd.service: Succeeded.
May 9 11:57:48 siddhesh ntpd[27825]: INIT: ntpd ntpsec-1.2.2+69-gc9c7f716e: Starting
May 9 11:57:48 siddhesh ntpd[27825]: 2023-05-09T11:57:48 ntpd[27825]: INIT: ntpd ntpsec-1.2.2+69-gc9c7f716e: Starting
May 9 11:57:48 siddhesh ntpd[27825]: INIT: Command line: /usr/local/sbin/ntpd -g -N -u ntp:ntp
May 9 11:57:48 siddhesh ntpd[27825]: INIT: precision = 0.145 usec (-23)
May 9 11:57:48 siddhesh ntpd[27826]: INIT: successfully locked into RAM
May 9 11:57:48 siddhesh ntpd[27826]: CONFIG: readconf: parsing file: /etc/ntp.conf
May 9 11:57:48 siddhesh ntpd[27826]: LOG: switching logging to file /var/log/ntp.log
May 9 11:58:19 siddhesh ntpd[27880]: INIT: ntpd ntpsec-1.2.2+69-gc9c7f716e: Starting
May 9 11:58:19 siddhesh ntpd[27880]: INIT: Command line: /usr/local/sbin/ntpd -n -d
May 9 11:58:19 siddhesh ntpd[27880]: INIT: precision = 0.044 usec (-24)
May 9 11:58:19 siddhesh ntpd[27880]: INIT: successfully locked into RAM
May 9 11:58:19 siddhesh ntpd[27880]: CONFIG: readconf: parsing file: /etc/ntp.conf
May 9 11:58:19 siddhesh ntpd[27880]: LOG: switching logging to file /var/log/ntp.log
May 9 11:59:11 siddhesh systemd[1]: ntpd.service: Succeeded.

```

The “ls -lrt <config-file>” command in Ubuntu can be used to check the last modification timestamp on the configuration files.

```

root@siddhesh:/var/log# ls -lrt /etc/ntp.conf
-rw-r--r-- 1 root root 1065 May 9 12:14 /etc/ntp.conf

```

d. Test Observation:

- If DUT uses NTP/PTP for time synchronization and is properly supporting the secure versions of NTP/PTP, and the logs are properly maintained for changes in the time settings, then the time synchronization is secure.
- If DUT is not properly supporting the secure versions of NTP/PTP OR the logs are not being properly maintained for changes in the time settings, then the time synchronization is not secure.

e. Evidence Provided:

Screenshot of pcap file & log files should be shared.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_SEC_TIME_SYNC		

2.3.9 TSTP Report for Evaluation of Restricted reachability of services

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.3.9
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 3: Software Security
2. **<Security Requirement No & Name >** 2.3.9 Restricted reachability of services
3. **<Requirement Description: >** The SYSTEM shall restrict the reachability of services such that they can be reached only on interfaces meant for the purpose. On interfaces where services are active, the reachability should be limited to legitimate communication peers. Administrative services (e.g. SSH, HTTPS, RDP) shall be restricted to interfaces in the management plane for separation of management traffic from user traffic.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.3.2.2]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)


```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.110.73 netmask 255.255.0.0 broadcast 192.168.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: **./system.out** (Used in IITH testbed to get SYSTEM version. Check with OEM manufacturer document for command specific to your SYSTEM)

Here we are assuming DUT to be SYSTEM, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: **cat /etc/os-release** (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SYSTEM_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)


```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:**

- The vendor shall declare, in the documentation accompanying the network product if the network product supports the capability to restrict services reachability to only the nodes authorized to access them. In this case, the vendor shall detail how this capability can be configured.
- A list of all required network protocols and services containing at least the following information shall be included in the documentation accompanying the Network Product:
 - protocol handlers and services needed for the operation of network product;
 - their open ports and associated services;
 - the configuration options;
 - and a description of their purposes.
- The network product is configured such that the required network protocols and services (as described in the network product documentation) are setup and each service is bound to an IP address of a specific network interface (e.g. IP1 which is the ip address of if1). Configuration may occur automatically during the initialization phase of the network product or manually as defined in the network product administration documentation.
- The network product shall have at least two interfaces enabled, if1 and if2 respectively configured with IP Address IP1 and IP2.
- The tester has administrative privileges.
- A tester machine equipped with a network port scanner tool is available.

7. **Test Objective:-** To verify that it is possible to bind the services only to the interfaces from which they are expected to be reachable.

8. **Test Plan**

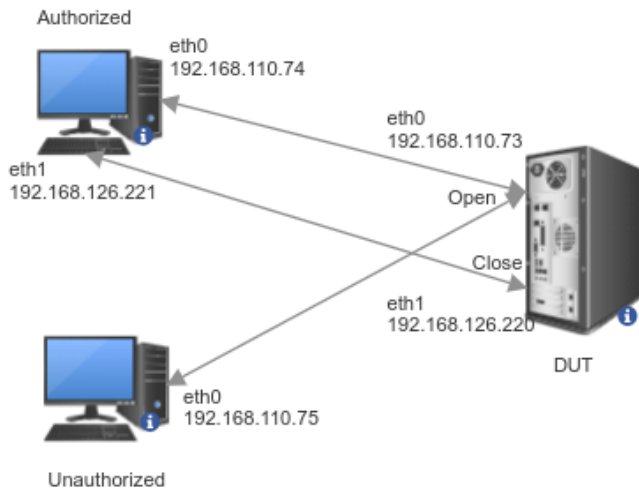
8.1. **Number of Test Scenarios:**

Securing Networks

8.1.1. **Test Scenarios for SSH**

- This test scenario tests the presence of ssh service on a network product (Additional Test scenarios based on the OEM document)

8.2. **Test Bed Diagram**



8.3. Tools Required nmap etc

8.4. Test Execution Steps:

- Tester scans the network product interface on which the service is open for the tester (say if1) (Case 1)
- Tester scans the network product interface on which the service is not open (say if2) (Case 2)
- Tester tries to scan if1 for the open service with another device which is not authorised by the DUT (Case 3)
- Perform the test for all configured services

9. Expected Results for Pass:

- **Case 1:** Tester verifies that only and only documented services for given interfaces are open in DUT.
- **Case 2:** Tester verifies that the documented services are not reachable from rogue devices

10. Expected Format of Evidence: Screenshots of Terminal

11. Test Execution

➤ **Test Case Number:** 01

a. **Test Case Name:** TC1_SSH_RS

b. **Test Case Description:** Tester analyses all the ports of the DUT on various interfaces using a port scanning tool

c. **Execution Steps:**

- The tester tries to contact DUT interface if1 with authorized tester device using following command
 - `sudo nmap -p- <ip_address of if1>`

```
Starting Nmap 7.80 ( https://nmap.org ) at 2023-06-11 21:22 IST
Nmap scan report for 192.168.110.73
Host is up (0.00055s latency).

PORT      STATE SERVICE
22/tcp    open  ssh

Nmap done: 1 IP address (1 host up) scanned in 0.23 seconds
```

- The tester tries to contact DUT interface if2 with authorized tester device using following command

- *sudo nmap -p- <ip_address of if2>*

```
Starting Nmap 7.80 ( https://nmap.org ) at 2023-07-04 14:47 IST
Nmap scan report for 192.168.126.220
Host is up (0.11s latency).

PORT      STATE SERVICE
22/tcp    closed ssh

Nmap done: 1 IP address (1 host up) scanned in 0.46 seconds
```

- The tester tries to contact DUT interface if1 with unauthorized tester device using following command

- *sudo nmap -p- <ip_address of if1>*

```
Starting Nmap 7.80 ( https://nmap.org ) at 2023-07-04 15:00 IST
Nmap scan report for 192.168.110.73
Host is up (0.11s latency).

PORT      STATE SERVICE
22/tcp    closed ssh

Nmap done: 1 IP address (1 host up) scanned in 0.48 seconds
```

d. Test Observations:

- **Case 1 (Positive Test Case):** Tester is able to reach only and only the documented services (int this case: ssh service) for a given interface of DUT
- **Case 2 (Negative Test Case):** Tester is not able to reach the documented services from rohue device

e. Evidence Provided:- Screenshot of Terminal

12. Test Case Result:

SL. No	Test case name	PASS/FAIL	Remarks
1	TC1_SSH_RS		

2.3.10 TSTP for Evaluation of Self Testing

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.3.10
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 3: Software Security
2. **<Security Requirement No & Name >** 2.3.10 Self Testing
3. **<Requirement Description: >** The SMF's cryptographic module shall perform power-up self-tests and conditional self-tests to ensure that the module is functioning properly. Power-up self-tests shall be performed when the cryptographic module is powered up. Conditional self-tests shall be performed when an applicable security function or operation is invoked (i.e., security functions for which self-tests are required). If a cryptographic module fails a self-test, the module shall enter an error state and output an error indicator via the status output interface. The cryptographic module shall not perform any cryptographic operations while in an error state.

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

Note: We have made this TSTP for "*OpenSSL FIPS-140*" the tester must verify for the respective Cryptographic suite as mentioned in the OEM documentation.

Example Cryptographic Software Suite:

- OpenSSL
- PGP
- GnuPG
- Microsoft Cryptography API (CryptoAPI)
- Cryptlib and more

- Command used: **fips-mode-setup -check** (To check if fips is working properly in kernel mode or not)

```
amf@localhost $ fips-mode-setup --check
FIPS mode is enabled.
```

- Command used: **fips-mode-setup --is-enabled** (To check if fips kernel mode is enabled or not)

```
amf@localhost $ fips-mode-setup --is-enabled
FIPS mode is enabled.
```

a. For OpenSSL:

- Command used: **openssl version** (To get version information)

```
amf@localhost $ openssl version
OpenSSL 3.1.0 14 Mar 2023 (Library: OpenSSL 3.0.8 7 Feb 2023)
```

- Command used: **cat <path to fipsmodule.cnf>** (To verify that the fips module is enabled with OpenSSL)

```
amf@localhost $ cat /usr/local/ssl/fipsmodule.cnf
[fips_sect]
activate = 1
install-version = 1
conditional-errors = 1
security-checks = 1
tls1-prf-ems-check = 0
module-mac = 7F:4A:E2:BC:FC:C8:70:57:DA:3C:ED:43:C9:7F:6D:19:77:46:7
1:45:42:05:F7:2F:DD:00:12:C5:D1:27:67:C8
```

- Here we can see that the activate is set to 1 (means activated)

Securing Networks

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Cryptographic Software Suite apart from OpenSSL is used by DUT.

6. Preconditions

- The tester has administrative privileges
- A tester machine is available.

7. **Test Objective:** To verify that the SMF's cryptographic module shall perform power-up self-tests and conditional self-tests to ensure that the module is functioning properly.

8. **Test Plan:**

8.1 **Number of Test Case Scenarios**

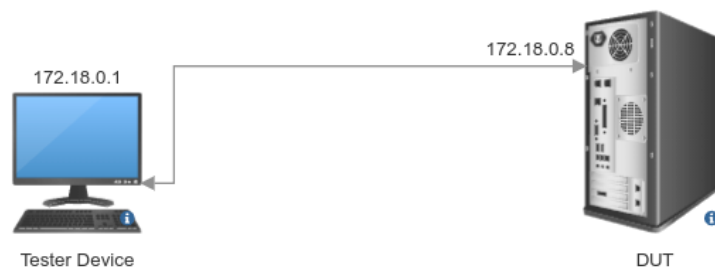
8.1.1 Test Scenario for POWERUP SELFTESTS on OpenSSL:

- The requirement checks for the Powerup self-test based on FIPS-140

8.1.2 Test Scenario for CONDITIONAL SELFTESTS on OpenSSL

- The requirement is checked specifically for OpenSSL
- For other Cryptographic modules check vendor document

8.2 **Test Setup Diagram:**



8.3 **Tools Used:** Default DUT access tool as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 **Test Execution Steps**

- Power up the testbed.
- The tester will comply the 2.6.2 and 2.6.3 of CSR to verify the information regarding the Crypto Module and Crypto Algo available.
- The tester tries to access the DUT command line.
- The tester should verify the power-up self-tests is enabled.
- The tester should verify the conditional self-tests is enabled.

9. **Expected Results for Pass:** The DUT cryptographic module must perform power-up self-tests and conditional self-tests to ensure that the module is functioning properly.

10. **Expected Format of Evidence:-** Log files and screen shots of test executions.

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC_POWERUP_SELFTESTS

b. **Test Case Description:** To verify that SMF's cryptographic module shall perform power-up self-tests to ensure that the module is functioning properly.

c. **Execution Steps:**

1. With proper tool and authorization, the tester shall open the CLI of the DUT and use the following command to verify.

- Command Used: ***reboot force***

```
amf@localhost $ reboot force
.....
Starting sysmand...
-----
This product contains third-party software provided under
one or more open source licenses. Type "show about" after
logging in for full license details.
-----
...

Mocana FIPS Power Up Self Test: Started...
Mocana FIPS Power Up Self Test: Finished

FIPS_RSA_Signature_Verify: PASSED!!!
Starting tSecured...
Starting tAuthd...
Starting tCtcd...
Starting tIked...
Starting tTscfd...
Starting tAppWeb...
Starting tAuditd...
Starting tAuditpusher...
Starting tSnmpd...
Starting snmpd...
Start platform alarm...
Starting tIFMIBd...
Initializing /opt/ Cleaner
Starting tLogCleaner task
Bringing up shell...

*****
*   System is in FIPS 140-2 level-2 compatible mode.   *
*   FIPS: All Power on self test completed successfully. *
*****
password secure mode is enabled
Admin Security is disabled
Starting SSH...
SSH_cli_init: allocated memory for 5 connections

*****
***   System is in FIPS 140-2 level-2 compatible mode.   ***
*****
Password:
```

- Here we can see that the FIPS Self-test is being performed

- d. **Test Observations:-** It is ensured that cryptographic module is enabled to perform power-up self-tests.

➤ **Test Case Number: 02**

- a. **Test Case Name:** TC_CONDITIONAL_SELFTESTS
- b. **Test Case Description:** To verify that SMF's cryptographic module shall perform conditional self-tests to ensure that the module is functioning properly.
- c. **Execution Steps:**
 1. With proper tool and authorization, the tester shall open the CLI of the DUT and use the following command to verify.
- Command Used: ***cat < path to fipsmodule.cnf >***

```
amf@localhost $ cat /usr/local/ssl/fipsmodule.cnf
[fips_sect]
activate = 1
install-version = 1
conditional-errors = 1
security-checks = 1
tls1-prf-ems-check = 0
module-mac = 7F:4A:E2:BC:FC:C8:70:57:DA:3C:ED:43:C9:7F
:6D:19:77:46:71:45:42:05:F7:2F:DD:00:12:C5:D1:27:67:C8
```

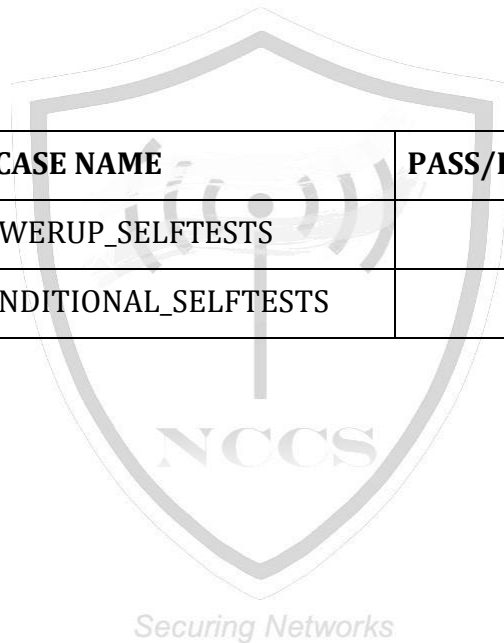
- Here we can see that the *conditional-errors* is set to 1 (Enabled)

d. Test Observations:

–It is ensured that cryptographic module is enabled to perform conditional self-tests.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_POWERUP_SELFTESTS		
2	TC_CONDITIONAL_SELFTESTS		



2.4.1 TSTP For No Unused Functions

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.4.1
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 4 - System Secure Execution Environment
2. **<Security Requirement No & Name >**2.4.1 No Unused Functions
3. **<Requirement Description: >** Unused functions i.e the software and hardware functions which are not needed for operation or functionality of the SMF shall be deactivated in the SMF's software and/or hardware. The list of hardware and software functions installed in the system shall match with the ones that have been mentioned and deemed necessary for the operation of the SMF.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.3.2.4]

Note: The reference to hardware may not be applicable here for GVNP Models of Type 1& 2.

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5. **DUT Configuration:** Different Linux distributions use different package managers.

Here are some commonly used package managers for popular Linux distributions:

- Debian, Ubuntu, and related distributions: dpkg (Debian Package)
- Red Hat Enterprise Linux (RHEL), CentOS, Fedora: yum (Yellowdog Updater Modified) or dnf (Dandified YUM)
- Arch Linux and derivatives: pacman (Package Manager)
- SUSE Linux Enterprise, openSUSE: zypper
- Gentoo Linux: emerge
- Alpine Linux: apk

- Slackware Linux: pkgtool

For Linux,

To determine the package manager available use **which** command with the appropriate package manager: **Command used: which dpkg**

```
amf@localhost $ which dpkg
/usr/bin/dpkg
```

If the command returns a path, it means the package manager is installed. If there is no output, it means the package manager is not available.

Similarly, and so on for other package managers.

Note: We have made TSTP for dpkg package manager. The tester must check for other package managers too based on OEM documentation.

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:** A list of all available software and associated components containing at least the following information shall be included in the documentation accompanying the Network Product:

- name of the software;
- version of the software installed;
- list of dependencies and versions;
- any add-ons and functions;
- any special hardware/debugging ports;
- software support type;
- licensing information;
- requirement during functioning of system;
- brief description of their purpose.

7. **Test Objective / Purpose:** To ensure that there is no unused hardware or software functions that are not deactivated in the network product which are not required for its operation or functionality.

8. **Test Plan:**

8.1 **Number of Test case scenarios: 1**

Test to check that there is no unused hardware or software functions that are not deactivated in the network product which are not required for its operation or functionality.

8.2 **TestBed Diagram:**



8.3 **Tools required:** Command Line Interface of the DUT

8.4 **Test Execution Step:**

The accredited evaluator's test lab is required to execute the following steps:

1. Identification of the hardware and software functions which are installed in the system or might have been disabled using any suitable command line tools or any other suitable means of determination.
2. Validate that there are no entries in the list of hardware and software functions installed in the system apart from the ones that have been mentioned and deemed necessary for the operation of the network product in the attached documentation.

9. **Expected Results:**

- The report will contain the names and versions of the tool(s) used for finding out what software and associated function is installed in the system. The detailed report will contain the name and version information of all the software and components installed in the system generated by the test tool.
- The list of all available software which has been deemed necessary for the operation of the network product by the vendor shall also be included as the test result. Any software not in the list of allowed software will be highlighted and brought out as a part of the report.
- There should be no unused function that is not deactivated in the network product except for the ones which are deemed necessary for its operation.

10. **Expected Format of Evidence:**

A testing report provided by the testing agency which will consist of the following information:

- The used tool(s) name and version information,

- Settings and configurations used
- The list of software and associated functions
- the test results i.e. allowed list of functions

11. Test Execution:

➤ **Test Case Number: 01**

a. Test Case Name: TC1_NO_UNUSED_FUNCTION

b. Test Case Description: Test Needs to be conducted to check that there exists no unused function.

c. Execution Steps:

At OS-Level:

- Check if the documentation accompanying the Network Product includes a list of all available software and libraries and associated components provided by the vendor.
- Use package management tools like ***dpkg -l*** to list installed packages in DUT.

```
snf@localhost $ dpkg -l
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend
|/ Err?=(none)/ReInst-required (Status,Err: uppercase=bad)
++-- Name Version Architecture Description
++--
ii accountsservice 0.6.55-0ubuntu12~20.04.6 amd64 query and manipulate user account information
ii acl 2.2.53-6 amd64 access control list - utilities
ii acpi-support 0.143 amd64 scripts for handling many ACPI events
ii acpid 1:2.0.32-1ubuntu1 amd64 Advanced Configuration and Power Interface event daemon
ii adduser 3.118ubuntu2 all add and remove users and groups
ii adwaita-icon-theme 3.36.1-2ubuntu0.20.04.2 all default icon theme of GNOME (small subset)
ii aislertot 1:3.22.9-1 amd64 GNOME solitaire card game collection
ii alsa-base 1.0.25+dfsg-0ubuntu5 all ALSA driver configuration files
ii alsa-topology-conf 1.2.2-1 all ALSA topology configuration files
ii alsa-ucm-conf 1.2.2-1ubuntu0.13 all ALSA Use Case Manager configuration files
ii alsa-utils 1.2.2-1ubuntu2.1 amd64 Utilities for configuring and using ALSA
ii amd64-microcode 3.20191218.1ubuntu1.1 amd64 Processor microcode firmware for AMD CPUs
ii anacron 2.3-29 amd64 cron-like program that doesn't go by time
ii anydesk 6.2.0 amd64 The fastest remote desktop software on the market.
```

- Compare the list of installed software functions obtained in the previous step with the list mentioned in the documentation accompanying the Network Product provided by the vendor.
- Ensure that all the entries in the installed software / libraries and functions match the ones deemed necessary for the operation of the network product.

At Application Level:

- Use appropriate Source Code Analyzer tool to verify that there is no unused software function present in the network product.
- Run the SCA tool against the source code to perform a comprehensive code analysis.
- The SCA tool will scan the codebase for potential issues, including unused software functions and components.

d. Test Observation:

➤ **Case 1: Unused Software Found in the DUT (Negative CASE)**

In case there is a mismatch between the list provided by the vendor and the installed software functions in DUT, then that means there are unused software functions present that are required to be removed.

➤ **Case 2: No unused Software Found in the DUT (Positive CASE)**

In case there is no mismatch between the list provided by the vendor and the installed software function in DUT, then that means there are no unused software functions present that are required to be removed.

e. Evidence Provided:

- The used tool(s) name and version information
- Settings and configurations used
- The list of software and associated functions
- the test results i.e. allowed list of functions

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_NO_UNUSED_FUNCTION		



2.4.2 TSTP For No Unsupported Components

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.4.2
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 4 - System Secure Execution Environment
2. **<Security Requirement No & Name >**2.4.2 No Unsupported Components
3. **<Requirement Description: >** OEM to ensure that the SMF shall not contain software and hardware components that are no longer supported by them or their 3rd Parties including the opensource communities, such as components that have reached end-of-life or end-of-support. An undertaking in this regard shall be given by OEM.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.3.2.5]

Note: The reference to hardware may not be applicable here for GVNP Models of Type 1& 2.

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. **DUT Configuration:** Different Linux distributions use different package managers.

Here are some commonly used package managers for popular Linux distributions:

- Debian, Ubuntu, and related distributions: dpkg (Debian Package)
- Red Hat Enterprise Linux (RHEL), CentOS, Fedora: yum (Yellowdog Updater Modified) or dnf (Dandified YUM)
- Arch Linux and derivatives: pacman (Package Manager)
- SUSE Linux Enterprise, openSUSE: zypper
- Gentoo Linux: emerge
- Alpine Linux: apk

- Slackware Linux: pkgtool

For Linux,

To determine the package manager available use **which** command with the appropriate package manager:

Command used: *which dpkg*

```
amf@localhost $ which dpkg
/usr/bin/dpkg
```

If the command returns a path, it means the package manager is installed. If there is no output, it means the package manager is not available.

Similarly, and so on for other package managers.

Note: We have made TSTP for dpkg package manager. The tester must check for other package managers too based on OEM documentation.

To get the hash of configuration file if the file is a ASCII text file

Command used: *sha256sum SMF_config.conf* (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: *docker images --digests* (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:** A list of all available software and associated components containing at least the following information shall be included in the documentation accompanying the Network Product:

- name of the software;
- version of the software installed;
- list of dependencies and versions;
- any add-ons and functions;
- any special hardware/debugging ports;
- software support type;
- licensing information;
- requirement during functioning of system;
- brief description of their purpose.

7. **Test Objective / Purpose:** To ensure that there is no unsupported software that is running in the network product which is not supported anymore and has reached its end-of-life or end-of-support.

8. **Test Plan:**

8.1 **Test Bed Diagram:**

8.2 **Tools required:** Command Line Interface of the DUT



8.3 **Test Execution Step:**

The accredited evaluator's test lab is required to execute the following steps:

1. Identification of the hardware and software components, version information and the kind of support available for the software provided by the vendor, the producer, the developer or other contractual partner of the operator using any tool or any other suitable means of determination.
2. Validate that there are no entries in the list of hardware and software installed in the system which are not supported as given by the vendor of network product in the attached documentation.

9. **Expected Results:**

- The report will contain the names and versions of the tool(s) used for finding out what software and hardware components are installed in the system. The detailed report will contain the name and version of the software and hardware used in the system, and the period of support for each of these components.
- The list of all available software and hardware components and their associated support information which has been deemed necessary for the operation of the network product by the vendor shall also be included as the test result. Any software or component which is not supported any longer by the vendor will be highlighted and brought out as a part of the report.
- There should be no software installed in the network product which is unsupported as of the day of testing.

10. **Expected Format of Evidence:** A testing report provided by the testing agency which will consist of the following information:

- The used tool(s) name and version information
- Software and hardware components used in the network product
- the test results i.e. support information of each listing

11. Test Execution:

➤ **Test Case Number: 01**

a. Test Case Name: TC1_NO_UNSUPPORTED_COMPONENTS

b. Test Case Description: Test Needs to be conducted to check that there exist no unsupported components.

c. Execution Steps:

At OS-Level:

- Check if the documentation accompanying the Network Product includes a list of all available software and libraries and associated components provided by the vendor.
- Use package management tools like ***dpkg -l*** to list installed packages in DUT.

```
amf@localhost $ dpkg -l
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-Inst/trig-aWait/Trig-pend
|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
++-----+
||/ Name                               Version                               Architecture Description
++-----+
ii accountsservice                     0.6.55-0ubuntu12-20.04.6             amd64        query and manipulate user account information
ii acl                                 2.2.53-6                             amd64        access control list - utilities
ii acpi-support                        0.143                                amd64        scripts for handling many ACPI events
ii acpid                               1:2.0.32-1ubuntu1                    amd64        Advanced Configuration and Power Interface event daemon
ii adduser                             3.118ubuntu2                         all          add and remove users and groups
ii adwaita-icon-theme                  3.36.1-2ubuntu0.20.04.2              all          default icon theme of GNOME (small subset)
ii aisleriot                           1:3.22.9-1                           amd64        GNOME solitaire card game collection
ii alsa-base                           1.0.25+dfsg-0ubuntu5                 all          ALSA driver configuration files
ii alsa-topology-conf                  1.2.2-1                              all          ALSA topology configuration files
ii alsa-ucm-conf                       1.2.2-1ubuntu0.13                    all          ALSA Use Case Manager configuration files
ii alsa-utils                          1.2.2-1ubuntu2.1                     amd64        Utilities for configuring and using ALSA
ii amd64-microcode                     3.20191218.1ubuntu1.1                amd64        Processor microcode firmware for AMD CPUs
ii anacron                             2.3-29                               amd64        cron-like program that doesn't go by time
ii anydesk                             6.2.0                                amd64        The fastest remote desktop software on the market.
```

- Compare the list of installed software components obtained in the previous step with the list mentioned in the documentation accompanying the Network Product provided by the vendor.
- Ensure that all the entries in the installed software / libraries and components match the ones deemed necessary for the operation of the network product.
- Also check the official documentation of each component and check for their current support status.
- Identify components that have reached end-of-life or end-of-support based on the official sources (e.g., vendor websites, community forums).
- Verify that none of the identified components with end-of-life or end-of-support status are present in DUT.
- For open-source software components, Check the status of support for each open-source component in the respective open-source communities.

At Application Level:

- Use appropriate Source Code Analyzer tool to verify that there is no unsupported software component present in the network product.
- Run the SCA tool against the source code to perform a comprehensive code analysis.
- The SCA tool will scan the codebase for potential issues, including unsupported software functions and components.

d. Test Observation:

➤ **Case 1: Unsupported Software in the DUT (Negative CASE)**

If any unsupported software component is identified in the DUT with end-of-life or end-of-support status. This component must be removed from the DUT.

➤ **Case 2: No Unsupported Software Found in the DUT (Positive CASE)**

If no unsupported software component is identified in the DUT with end-of-life or end-of-support status. And all the open-source software components present in DUT have active community support, then this Test Case Passes.

e. Evidence Provided:

- The used tool(s) name and version information
- Settings and configurations used
- The list of software and associated functions
- the test results i.e. allowed list of functions

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_NO_UNSUPPORTED_COMPONENT		



2.4.3 TSTP Report for Evaluation of Avoidance of Unspecified mode of Access

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.4.3
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 4: System Secure Execution Environment
2. **<Security Requirement No & Name>** 2.4.3 Avoidance of Unspecified mode of Access
3. **<Requirement Description>** SMF shall not contain any wireless access mechanism which is unspecified or not declared.

An undertaking shall be given by the OEM as follows: "The SMF does not contain any wireless, optical, magnetic or any other component that may be used as a covert channel"

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. **DUT Configuration:** Run the command `lsblk` to check if there are any optical or magnetic components are connected to DUT. Here we can check for other components also.

```

vm2@vm2:~$ lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
loop0       7:0      0    4K  1 loop /snap/bare/5
loop1       7:1      0  63.4M  1 loop /snap/core20/1950
loop2       7:2      0  73.8M  1 loop /snap/core22/750
loop3       7:3      0  63.5M  1 loop /snap/core20/1891
loop4       7:4      0  73.9M  1 loop /snap/core22/766
loop5       7:5      0   6.4M  1 loop /snap/curl/1754
loop6       7:6      0 244.8M  1 loop /snap/firefox/2760
loop7       7:7      0 244.5M  1 loop /snap/firefox/2800
loop8       7:8      0 346.3M  1 loop /snap/gnome-3-38-2004/119
loop9       7:9      0 349.7M  1 loop /snap/gnome-3-38-2004/140
loop10      7:10     0 460.7M  1 loop /snap/gnome-42-2204/105
loop11      7:11     0 466.5M  1 loop /snap/gnome-42-2204/111
loop12      7:12     0  91.7M  1 loop /snap/gtk-common-themes/1535
loop13      7:13     0  45.9M  1 loop /snap/snap-store/638
loop14      7:14     0  12.3M  1 loop /snap/snap-store/959
loop15      7:15     0  53.3M  1 loop /snap/snapd/19457
loop16      7:16     0   304K  1 loop /snap/snapd-desktop-integration/49
loop17      7:17     0  53.3M  1 loop /snap/snapd/19361
sr0         11:0     1 1024M  0 rom
vda         252:0     0   20G  0 disk
├─vda1      252:1     0    1M  0 part
├─vda2      252:2     0   513M  0 part /boot/efi
└─vda3      252:3     0 19.5G  0 part /var/snap/firefox/common/host-hunspell

```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

6. Preconditions

- The tester should have root privileges to login to SMF.
- The tester needs to check for the undertaking provided by OEM.
- The OEM should provide an undertaking specifying that there is no wireless access mechanism used.

Securing Networks

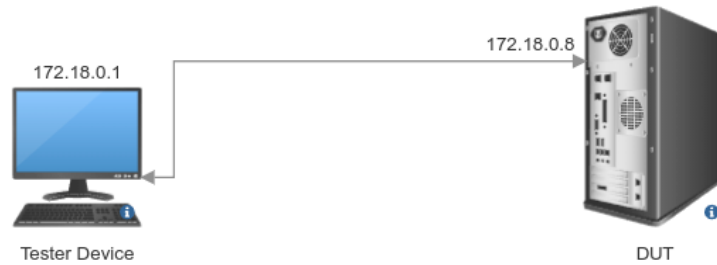
7. **Test objective:-** The network product should not contain any wireless access mechanisms.

8. Test Plan:

8.1 Number of Test case scenarios: 1

Test to check that network product does not contain any wireless access mechanisms.

8.2 Test Setup Diagram:



8.3 **Tools Used:** Command line

8.4 **Test Execution Steps**

- Check that all pre-conditions are met.
- The tester should check the undertaking provided by OEM.
- Login to the network product.
- Check if there are any wireless access mechanisms present.
- Open terminal
 - Write “**iwconfig**” command and check the output, if there are any wireless network present it will show
 - The tester can also check if there are any network device connected to DUT using the command - “**lspci | grep -i network**”.

9. **Expected Results for Pass:** There should not be any wireless access mechanisms present.

10. **Expected Format of Evidence:** Screenshots proving no wireless access mechanisms present.

11. **Test Execution:**

➤ **Test Case Number:** 1

a. **Test Case Name:** TC_AVOIDANCE_OF_UNSPECIFIED_MODE_OF_ACCESS

b. **Test Case Description:** To ensure that there is no wireless access mechanism present in SMF if not specified.

c. **Execution Steps: Success Case:**

Checking if wireless devices are connected.

```
osboxes@osboxes:~$ iwconfig
enp0s3    no wireless extensions.

lo        no wireless extensions.

osboxes@osboxes:~$
```

Checking for any other network device

```
osboxes@osboxes:~$ lspci | grep -i network
osboxes@osboxes:~$
```

Failure Case:

```
osboxes@osboxes:~$ iwconfig
lo          no wireless extensions.

eno1        no wireless extensions.

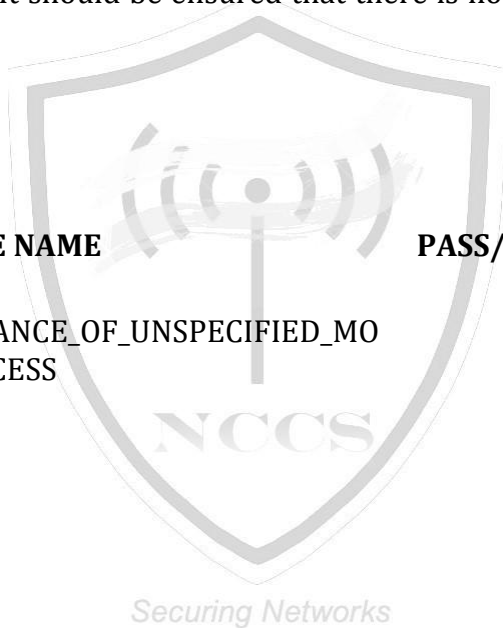
wlo1        IEEE 802.11  Mode:Master  Tx-Power=22 dBm
           Retry short limit:7   RTS thr:off   Fragment thr:off
           Power Management:on
```

Note: To check for any hidden wireless access, we need to run security tools. These tools can be used for monitoring purposes.

d. Test Observations: It should be ensured that there is no wireless access mechanisms present.

12. Test Case Result:

Sl. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_AVOIDANCE_OF_UNSPECIFIED_MODE_OF_ACCESS		



2.5.1 TSTP Report for Evaluation of Audit trail storage and protection

Session Management Function ITSAR

ITSAR No: ITSAR111092401

Clause no: 2.5.1

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. <ITSAR Section No & Name> Section 5: User Audit
2. <Security Requirement No & Name > 2.5.1 Audit trail storage and protection
3. <Requirement Description: > The security event log shall be accessing controlled (file access rights) so only privileged users have access to the log files.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.6.3]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use the command line interface to find OS name and version in Linux.
- Use the following command, to display the information about the operating system release on a Unix/Linux system. **cat /etc/os-release**

```
amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal
```

Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

5. **DUT Configuration:** Check the Operating system and its version for the DUT to know the commands for setting access permissions for data and application execution.

- To get the hash of configuration file if the file is a ASCII text file.

- Command - **sha256sum DUT_config.conf**

- **Digest Hash of Tested Configuration:**

- DUT_config.conf:

19be54b42975f7f414c53a280373aa9466398781ecd77a2e9d090dbc6dbfdd8

- To get the hash of OS if using docker

- Command - **docker images --digests**

- **Digest Hash of OS:**

- DUT_IMAGE:

fd363f0e0d146c869d60649bc4f42e7008829d9d67a5dfdaf3b38ad24af7d53a

Securing Networks

6. **Pre-Conditions:**

- Documentation describing where logs are stored and how these logs are accessed and the Network Product interfaces that these logs can be accessed from.
- OEM provides the list of users authorized to access the log files.

7. **Test Objective:** The security event log shall be access-controlled (file access rights) so only privileged users have access to the log files.

8. **Test Plan:**

8.1 **Number of test scenarios/test cases: 1**

8.2 **Tools used:** Command line of network function.

8.3 **Test case Execution:** The accredited evaluator's test lab is required to execute the following steps:

- The tester attempts to access log files using users' accounts with and without the correct permissions for accessing log files.
- Repeat the test as described in step 1 using each of the interfaces as described in the Network Product documentation.

9. **Expected Results for Pass:** The Log files are ONLY accessible when a user with the appropriate authorization attempts to access them and fails when a user without the correct permissions attempts to access them

10. **Expected Format of Evidence:** The tester checks that log files are accessible when a user with the appropriate authorization attempts to access them and fails when a user without the correct permissions attempts to access them. Screenshots showing that unauthorized users have been denied access to the log files and authorized users are able to access these files.

11. **Test Execution:**

➤ **Test Case Number:** 1

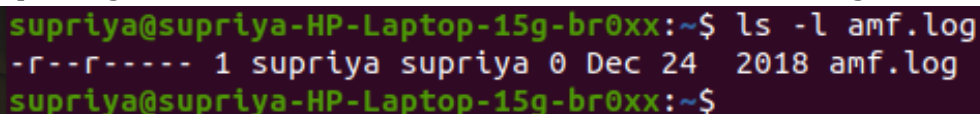
a. **Test Case Name:** TC_VERIFY_LOG_FILES_AUTH

b. **Test Case Description:** The tester checks that log files are accessible when a user with the appropriate authorization attempts to access them and fails when a user without the correct permissions attempts to access them.

c. **Test Execution:** The accredited evaluator's test lab is required to execute the following steps:

- The tester attempts to access log files using users' accounts with and without the correct permissions for accessing log files.
- Repeat the test as described in step 1 using each of the interfaces as described in the Network Product documentation.

d. **Test Observations:** In the given test case, SMF.log is the log file under test. The owner/admin is the user "supriya". As can be seen in the screenshot below, the owner has the right to view the file. Also, the owner can add users to this group and only those privileged users will be allowed to view the contents of the log file.



```
supriya@supriya-HP-Laptop-15g-br0xx:~$ ls -l amf.log
-r--r----- 1 supriya supriya 0 Dec 24 2018 amf.log
supriya@supriya-HP-Laptop-15g-br0xx:~$
```

To view file permissions: **ls -l <filename>**

➤ **Case1:** The tester attempts to access log files using users accounts without the correct permissions for accessing log files.

```
supriya@supriya-HP-Laptop-15g-br0xx:~$ su user1
Password:
$ cat amf.log
cat: amf.log: Permission denied
$
```

- **Case 2:** The tester attempts to access log files using users accounts with the correct permissions for accessing log files.

```
(base) supriya@supriya-Super-Server: $ cat amf.log
@INSTANCE@
@PID_DIRECTORY@
@MCC@
@MNC@
@REGION_ID@
@AMF_SET_ID@
@SERVED_GUAMI_MCC_0@
@SERVED_GUAMI_MNC_0@
@SERVED_GUAMI_REGION_ID_0@
@SERVED_GUAMI_AMF_SET_ID_0@
@SERVED_GUAMI_MCC_1@
@SERVED_GUAMI_MNC_1@
@SERVED_GUAMI_REGION_ID_1@
@SERVED_GUAMI_AMF_SET_ID_1@
@PLMN_SUPPORT_MCC@
@PLMN_SUPPORT_MNC@
@PLMN_SUPPORT_TAC@
@SST_0@
@SD_0@
@SST_1@
@SD_1@
@AMF_INTERFACE_NAME_FOR_NGAP@
@AMF_INTERFACE_NAME_FOR_N11@
```

12. Test Case Result:

SL. No	OUTCOME OF RUNNING THE PASS/FAIL SCRIPT (CASE 1/CASE2)	REMARKS
1	TC_VERIFY_LOG_FILES_AUTH	

Securing Networks

2.5.2 TSTP for Evaluation of Audit Event Generation

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.5.2
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<TSTP Document ID:>

<Applicant Name:> Ex: XYZ

<Application Number>

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 5– User Audit
2. **<Security Requirement No & Name >** 2.5.2 Audit Event Generation
3. **<Requirement Description: >** The SMF shall log all important Security events with unique System Reference details as given in the Table below. SMF shall record within each audit record at least information pertaining to Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event. Additional audit record information, depending on the audit event, shall also be provided as given in the Table below:

Event Types (Mandatory or optional)	Description	Event data to be logged
Incorrect login attempts (Mandatory)	Records any user incorrect login attempts to the TTE.	Username
		Source (IP address) if remote access
		Outcome of event (Success or failure)
		Timestamp
Administrator access (Mandatory)	Records any access attempts to accounts that have system privileges.	Username,
		Timestamp,
		Length of session

		Outcome of event (Success or failure)
		Source (IP address) if remote access
Account administration (Mandatory)	Records all account administration activity, i.e. configure, delete, copy, enable, and disable.	Administrator username,
		Administered account,
		Activity performed (configure, delete, enable and disable)
		Outcome of event (Success or failure)
		Timestamp
Resource Usage (Mandatory)	Records events that have been triggered when system parameter values such as disk space, CPU load over a longer period have exceeded their defined thresholds.	Value exceeded,
		Value reached
		(Here suitable threshold values shall be defined depending on the individual system.)
		Outcome of event (Threshold Exceeded)
		Timestamp
Configuration change (Mandatory)	Changes to configuration of the network device	Change made
		Timestamp
		Outcome of event (Success or failure)
		Username
Reboot/shutdown/crash (Mandatory)	This event records any action on the network device/TTE that forces a reboot or shutdown OR where the network device/TTE has crashed.	Action performed (boot, reboot, shutdown, etc.)
		Username (for intentional actions)
		Outcome of event (Success or failure)
		Timestamp
Interface status change (Mandatory)	Change to the status of interfaces on the network device/TTE (e.g. shutdown)	Interface name and type
		Status (shutdown, down missing link, etc.)
		Outcome of event (Success or failure)
		Timestamp
		Administrator username,

Change of group membership or accounts (Optional)	Any change of group membership for accounts	Administered account,
		Activity performed (group added or removed)
		Outcome of event (Success or failure)
		Timestamp.
Resetting Passwords (Optional)	Resetting of user account passwords by the Administrator	Administrator username
		Administered account
		Activity performed (configure, delete, enable and disable)
		Outcome of event (Success or failure)
		Timestamp
Services (Optional)	Starting and Stopping of Services (if applicable)	Service identity
		Activity performed (start, stop, etc.)
		Timestamp
		Outcome of event (Success or failure)
X.509 Certificate Validation (Optional)	Unsuccessful attempt to validate a certificate	Timestamp
		Reason for failure
		Subject identity
		Type of event
Secure Update (Optional)	Attempt to initiate manual update, initiation of update, completion of update	User identity
		Timestamp
		Outcome of event (Success or failure)
		Activity performed
Time change (Mandatory)	Change in time settings	Old value of time
		New value of time
		Timestamp
		origin of attempt to change time (e.g. IP address)
		Subject identity
		Outcome of event (Success or failure)
		User identity
Session unlocking/	Any attempts at unlocking of an interactive session, termination of a	User identity (wherever applicable)

termination (Optional)	remote session by the session locking mechanism, termination of an interactive session.	Timestamp
		Outcome of event (Success or failure)
		Subject identity
		Activity performed
		Type of event
Trusted Communication paths with IT entities such as Authentication Server, Audit Server, NTP Server, etc. and for authorised remote administrators (Optional)	Initiation, Termination and Failure of trusted Communication paths	Timestamp
		Initiator identity (as applicable)
		Target identity (as applicable)
		User identity (in case of Remote administrator access)
		Type of event
Audit data changes (Optional)	Changes to audit data including deletion of audit data	Outcome of event (Success or failure, as applicable)
		Timestamp
		Type of event (audit data deletion, audit data modification)
		Outcome of event (Success or failure)
		Subject identity
		User identity
		origin of attempt to change time (e.g. IP address)
Port Scan attempts	Any attempt to scan the network interface shall lead to triggering of logging of the appropriate parameters	Details of data deleted or modified
		Date
		Time Stamp
		Source IP address
User Login (Mandatory)	All use of Identification and authentication mechanisms.	Destination Port address
		User identity
		Origin of attempt (IP address)

		Outcome of event (Success or failure)
		Timestamp

[Reference: TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.6.1]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. **DUT Configuration:** The DUT must grant required privileges to the user/tester to generate the above-mentioned events and access the log files on the system.

6. **Preconditions:** The following information shall be provided by the documentation accompanying the network product:

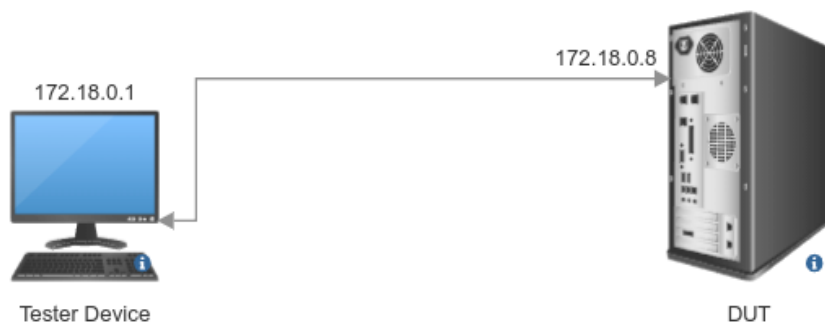
- The log where the event is recorded and how it can be accessed (e.g. the complete path).
- If the event type is enabled by default or how to enable it.
- What O&M services can be used on the Network Product in the configuration according to the pre-requisites for testing and how to use them.
- The tester has the needed administrative privileges to sufficiently perform the tests.
- If needed for testing specific O&M services, a tester machine is available.

7. **Test Objective** To verify that all the Security events are logged together with a unique system reference (e.g. host name, IP or MAC address) and the exact time the incident occurred. For each security event, the log entry shall include user name and/or timestamp and/or performed action and/or result and/or length of session and/or values exceeded and/or value reached.

Securing Networks

8. **Test Plan:**

8.1 **Test Setup Diagram:**



8.2 Tools Used

8.3 Test Execution Steps

1. The Tester sequentially triggers each security event listed in the requirement, while covering each option detailed in the individual security event descriptions.
2. The Tester verifies whether the security events, and their individual options, were correctly logged. In particular it is verified whether they include at least the event data specified as required to be logged.

9. **Expected Results to Pass:** The security events are properly logged along with the minimum required details as mentioned in the above table.

10. **Expected Evidence:** The testing report contains the following information for each security event:

- List of O&M services.
- Commands executed per O&M services
- The relevant parts of the logs in appropriate form (e.g. file, screenshot).

11. Test Execution:

➤ **Test Case Number: 1**

- a. **Test Case Name:** TC_AUDIT_EVENT_GENERATION
- b. **Test Case Description:** To verify that the network product correctly logs all required security event types.
- c. **Execution Steps:** For each O&M service perform the following test steps
 1. The Tester sequentially triggers each security event listed in the requirement, while covering each option detailed in the individual security event descriptions.
 2. The Tester verifies whether the security events, and their individual options, were correctly logged. In particular it is verified whether they include at least the event data specified as required to be logged.

NOTE: The commands in the following description are considering Ubuntu 20.04 as the Operating System. The commands may vary for different Operating Systems.

- We Simulated the test case for the 'Incorrect login attempts' scenario.
- We make an Incorrect login attempt to the DUT via SSH from the tester's machine.
- As per the requirement, the Incorrect Login attempt must be logged by the DUT.
- The basic command to list all SSH failed login attempts is:
grep "Failed password" /var/log/auth.log.
- The same can be achieved by executing the cat command:

cat /var/log/auth.log | grep "Failed password".

```
root@stark99:/var/log# cat auth.log | grep "Failed password"
Jul 24 20:21:18 stark99 sshd[26829]: Failed password for invalid user stark99 from 172.19.124.163 port 40738 ssh2
Jul 24 20:21:26 stark99 sshd[26829]: Failed password for invalid user stark99 from 172.19.124.163 port 40738 ssh2
Jul 24 20:21:39 stark99 sshd[26829]: Failed password for invalid user stark99 from 172.19.124.163 port 40738 ssh2
root@stark99:/var/log#
```

Here, the username, timestamp, source of the event (IP) and outcome of the event is logged. Similarly, the tester can simulate other security events and check if they are being logged by the DUT or not.

In the screenshot given below, the local Incorrect login attempts are logged with the username, timestamp and outcome of the event.

```
siddhesh@siddhesh:~$ journalctl -q SYSLOG_FACILITY=10 SYSLOG_FACILITY=4 | tail -f -n 10
May 09 15:43:34 siddhesh su[56744]: (to root) siddhesh on pts/0
May 09 15:43:34 siddhesh su[56744]: pam_unix(su:session): session opened for user root by (uid=0)
May 09 15:43:38 siddhesh su[56744]: pam_unix(su:session): session closed for user root
May 09 15:43:38 siddhesh sudo[56743]: pam_unix(sudo:session): session opened for user root
May 09 15:43:44 siddhesh su[12665]: pam_unix(su:session): session opened for user root
May 09 15:43:44 siddhesh sudo[12564]: pam_unix(sudo:session): session opened for user root
May 09 15:43:50 siddhesh sudo[56763]: pam_unix(sudo:auth): authentication failure; logname= uid=1000 euid=0 tty=/dev/pts/0 ruser=siddhesh rhost= user=siddhesh
May 09 15:43:53 siddhesh sudo[56763]: pam_unix(sudo:auth): conversation failed
May 09 15:43:53 siddhesh sudo[56763]: pam_unix(sudo:auth): auth could not identify password for [siddhesh]
May 09 15:43:53 siddhesh sudo[56763]: siddhesh : 1 incorrect password attempt ; TTY=pts/0 ; PWD=/home/siddhesh ; USER=root ; COMMAND=/usr/bin/su
```

Administrator & other user login/access:

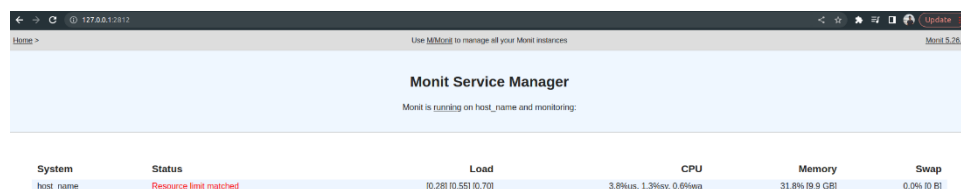
In Ubuntu 20.04, the admin access is logged at location /var/log/auth.log

```
Aug 18 04:39:01 supriya-Super-Server CRON[37357]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 04:39:01 supriya-Super-Server CRON[37357]: pam_unix(cron:session): session closed for user root
Aug 18 05:09:01 supriya-Super-Server CRON[38241]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 05:09:01 supriya-Super-Server CRON[38241]: pam_unix(cron:session): session closed for user root
Aug 18 05:17:01 supriya-Super-Server CRON[38446]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 05:17:01 supriya-Super-Server CRON[38446]: pam_unix(cron:session): session closed for user root
Aug 18 05:39:01 supriya-Super-Server CRON[39018]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 05:39:01 supriya-Super-Server CRON[39018]: pam_unix(cron:session): session closed for user root
Aug 18 06:03:45 supriya-Super-Server pkexec: pam_unix(polkit-1:session): session opened for user root(uid=0) by (uid=1000)
Aug 18 06:03:45 supriya-Super-Server pkexec[41182]: supriya: Executing command [USER=root] [TTY=unknown] [CMD=/home/supriya]
Aug 18 06:09:01 supriya-Super-Server CRON[41245]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 06:09:01 supriya-Super-Server CRON[41245]: pam_unix(cron:session): session closed for user root
Aug 18 06:17:01 supriya-Super-Server CRON[41497]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 06:17:01 supriya-Super-Server CRON[41497]: pam_unix(cron:session): session closed for user root
Aug 18 06:25:01 supriya-Super-Server CRON[41788]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 06:25:01 supriya-Super-Server CRON[41788]: pam_unix(cron:session): session closed for user root
Aug 18 06:39:01 supriya-Super-Server CRON[42495]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 06:39:01 supriya-Super-Server CRON[42495]: pam_unix(cron:session): session closed for user root
Aug 18 06:39:43 supriya-Super-Server pkexec: pam_unix(polkit-1:session): session opened for user root(uid=0) by (uid=1000)
Aug 18 06:39:43 supriya-Super-Server pkexec[42628]: supriya: Executing command [USER=root] [TTY=unknown] [CMD=/home/supriya]
Aug 18 07:09:01 supriya-Super-Server CRON[43398]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 07:09:01 supriya-Super-Server CRON[43398]: pam_unix(cron:session): session closed for user root
Aug 18 07:17:01 supriya-Super-Server CRON[43641]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 07:17:01 supriya-Super-Server CRON[43641]: pam_unix(cron:session): session closed for user root
Aug 18 07:30:01 supriya-Super-Server CRON[43961]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 07:30:01 supriya-Super-Server CRON[43961]: pam_unix(cron:session): session closed for user root
Aug 18 07:39:02 supriya-Super-Server CRON[44290]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 07:39:02 supriya-Super-Server CRON[44290]: pam_unix(cron:session): session closed for user root
Aug 18 08:09:01 supriya-Super-Server CRON[45100]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 08:09:01 supriya-Super-Server CRON[45100]: pam_unix(cron:session): session closed for user root
Aug 18 08:17:01 supriya-Super-Server CRON[45376]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Aug 18 08:17:01 supriya-Super-Server CRON[45376]: pam_unix(cron:session): session closed for user root
Aug 18 08:30:01 supriya-Super-Server CRON[45716]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
```

Resource Usage:

The resource usage limits can be set and logged using an open-source tool *Monit*. *Monit* can be configured to log usage of resources such as CPU, memory etc and raise alerts if usage goes beyond certain threshold.

Here, we can see that the CPU usage of user exceeds the threshold, and hence it gives the alert “resource limit matched”



The screenshot shows the Monit Service Manager interface. At the top, it says 'Monit is running on host_name and monitoring:'. Below this, there is a table with columns: System, Status, Load, CPU, Memory, and Swap. The 'System' column has 'host_name'. The 'Status' column has 'Resource limit matched'. The 'Load' column has '[0.28] [0.55] [0.70]'. The 'CPU' column has '3.8KHz, 1.3Hz, 0.6Hz'. The 'Memory' column has '31.8% [9.9 GB]'. The 'Swap' column has '0.0% [0 B]'.

System	Status	Load	CPU	Memory	Swap
host_name	Resource limit matched	[0.28] [0.55] [0.70]	3.8KHz, 1.3Hz, 0.6Hz	31.8% [9.9 GB]	0.0% [0 B]

It logs the events at the location /var/log/monit.log

```
siddhesh@stark99:~/TSTP/tstp_audit_event_generation$ sudo cat /var/log/monit.log
[sudo] password for siddhesh:
[IST Aug 18 12:46:07] info      : New Monit id: 45a57f266c409fe35eec80ba1a332990
Stored in '/var/lib/monit/id'
[IST Aug 18 12:46:07] info      : Starting Monit 5.26.0 daemon
[IST Aug 18 12:46:07] info      : 'stark99' Monit 5.26.0 started
[IST Aug 18 13:02:34] info      : Reinitializing monit daemon
[IST Aug 18 13:02:34] info      : Reinitializing Monit -- control file '/etc/monit/monitrc'
[IST Aug 18 13:02:34] info      : 'host_name' Monit reloaded
[IST Aug 18 13:02:55] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:02:56] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:02:57] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:04:03] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:04:07] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:04:09] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:04:10] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:04:10] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:04:10] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:04:11] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:04:11] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:05:39] info      : Monit daemon with pid [124921] stopped
[IST Aug 18 13:05:39] info      : 'host_name' Monit 5.26.0 stopped
[IST Aug 18 13:06:59] info      : Starting Monit 5.26.0 daemon with http interface at [localhost]:2812
[IST Aug 18 13:06:59] info      : 'host_name' Monit 5.26.0 started
[IST Aug 18 13:07:27] error      : Denied connection from non-authorized client [::1]
[IST Aug 18 13:07:48] error      : HttpRequest: access denied -- client [127.0.0.1]: unknown user 'monit'
[IST Aug 18 13:07:57] error      : HttpRequest: access denied -- client [127.0.0.1]: unknown user 'monit'
[IST Aug 18 13:08:38] error      : HttpRequest: access denied -- client [127.0.0.1]: unknown user 'monit'
[IST Aug 18 13:11:27] info      : Reinitializing monit daemon
[IST Aug 18 13:11:27] info      : Reinitializing Monit -- control file '/etc/monit/monitrc'
[IST Aug 18 13:11:27] info      : 'host_name' Monit reloaded
[IST Aug 18 13:11:27] error      : 'host_name' cpu user usage of 2.9% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:11:37] error      : 'host_name' cpu user usage of 3.8% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:11:48] error      : 'host_name' cpu user usage of 3.8% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:11:58] error      : 'host_name' cpu user usage of 3.6% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:12:08] error      : 'host_name' cpu user usage of 3.6% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:12:18] error      : 'host_name' cpu user usage of 7.1% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:12:28] error      : 'host_name' cpu user usage of 5.8% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:12:38] error      : 'host_name' cpu user usage of 4.5% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:12:48] error      : 'host_name' cpu user usage of 3.5% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:12:58] error      : 'host_name' cpu user usage of 6.8% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:13:08] error      : 'host_name' cpu user usage of 8.0% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:13:18] error      : 'host_name' cpu user usage of 5.9% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:13:28] error      : 'host_name' cpu user usage of 8.4% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:13:38] error      : 'host_name' cpu user usage of 5.6% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:13:48] error      : 'host_name' cpu user usage of 4.4% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:13:58] error      : 'host_name' cpu user usage of 3.2% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:14:08] error      : 'host_name' cpu user usage of 3.5% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:14:18] error      : 'host_name' cpu user usage of 2.7% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:14:28] error      : 'host_name' cpu user usage of 3.6% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:14:38] error      : 'host_name' cpu user usage of 2.9% matches resource limit [cpu user usage > 2.0%]
```

Configuration changes:

For checking changes made to configuration files, we check if hash of conf file does not change over time. If changed, the same will be detected by *monit* and user will be alerted.

Before changing cof file:

Monit Service Manager					
Monit is running on host_name and monitoring:					
System	Status	Load	CPU	Memory	Swap
host_name	Resource limit matched	[0.80] [1.11] [1.05]	4.8%us, 1.7%sy, 0.5%wa	35.9% [11.1 GB]	0.0% [0 B]
File	Status	Size	Permission	UID	GID
test_conf.txt	OK	246 B	0664	1000	1000

After changing conf file:

Monit Service Manager					
Monit is running on host_name and monitoring:					
System	Status	Load	CPU	Memory	Swap
host_name	Resource limit matched	[0.77] [0.94] [0.99]	3.9%us, 1.3%sy, 0.7%wa	36.1% [11.2 GB]	0.0% [0 B]
File	Status	Size	Permission	UID	GID
test_conf.txt	Checksum failed	246 B	0664	1000	1000

The “checksum failed” status is alerted and logged.

```

[IST Aug 18 13:38:08] error : 'host_name' cpu user usage of 4.8% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:38:18] error : 'host_name' cpu user usage of 5.3% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:38:18] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:38:28] error : 'host_name' cpu user usage of 3.9% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:38:28] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:38:38] error : 'host_name' cpu user usage of 5.8% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:38:38] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:38:48] error : 'host_name' cpu user usage of 4.9% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:38:48] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:38:58] error : 'host_name' cpu user usage of 4.6% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:38:58] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:39:08] error : 'host_name' cpu user usage of 5.5% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:39:08] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:39:18] error : 'host_name' cpu user usage of 5.2% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:39:18] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:39:28] error : 'host_name' cpu user usage of 9.3% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:39:28] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:39:38] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:39:38] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:39:48] error : 'host_name' cpu user usage of 5.0% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:39:48] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:39:58] error : 'host_name' cpu user usage of 2.0% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:39:58] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca
[IST Aug 18 13:40:08] error : 'host_name' cpu user usage of 2.1% matches resource limit [cpu user usage > 2.0%]
[IST Aug 18 13:40:08] error : 'test_conf.txt' checksum failed, expected b27dbbc954b07cca2c7a0f1c3f191b0d got 2fa7caf115a7bd80ec6c9a784fe9ca

```

Note: same can be used for logging changes in time settings by monitoring the time settings conf file, such as the *ntp.conf* file for NTP.

Reboot/Shutdown/crash:

Such events are logged and can be analyzed using the tool 'journalctl' *journalctl --list-boots*

```

siddhesh@ark99:~/TSTP/tstp_audit_event_generation$ journalctl --list-boots
-183 4ea772ef9fbb403b9c45ef9e5db335f8 Fri 2022-04-22 15:53:09 IST-Sat 2022-04-23 16:45:20 IST
-182 7d6cb41f15254e9890368db9c9500d5b1 Sat 2022-04-23 16:47:30 IST-Wed 2022-04-27 18:26:31 IST
-181 6f6c65c91378b4ff9d14a0cec2896f5d2 Wed 2022-04-27 21:53:22 IST-Fri 2022-04-29 13:04:30 IST
-180 d5f4670f321b4719b4943bd61451f010 Fri 2022-04-29 13:05:45 IST-Thu 2022-06-30 09:30:01 IST
-179 6ad29d9a711845ecb75b34d2ca28ebd0 Thu 2022-06-30 09:36:54 IST-Sun 2022-07-10 17:26:21 IST
-178 49e37bf6216a4a7f95299d0a6c896001 Sun 2022-07-10 18:42:43 IST-Sun 2022-07-24 09:43:36 IST
-177 bfef0f72b1a4474197c0ca183fc8f54c Sun 2022-07-24 11:54:20 IST-Tue 2022-08-09 17:15:17 IST
-176 dcceb8ab33bb4d3cb490b88ec63c9a83 Wed 2022-08-17 11:29:43 IST-Fri 2022-09-02 11:17:17 IST
-175 d85f53473fb24a9e96073266fe764330 Fri 2022-09-02 11:18:09 IST-Fri 2022-09-02 11:36:22 IST
-174 aaffcd3f659b44ae88e1aa9900db7144 Fri 2022-09-02 11:37:11 IST-Sat 2022-09-24 19:01:02 IST
-173 d959bd9d60c14fbbb32dace2c1444ede Sat 2022-09-24 19:14:37 IST-Wed 2022-09-28 14:20:22 IST
-172 17f5696386f84d309e3ac277277bb088 Wed 2022-09-28 14:31:36 IST-Wed 2022-09-28 14:37:21 IST
-171 f4c26c0ed8274b5bbf1a1f60879b84b6a Wed 2022-09-28 14:54:45 IST-Wed 2022-09-28 15:00:29 IST
-170 d28c46230cd41b29cabaf25e68f1b62 Wed 2022-09-28 15:08:04 IST-Thu 2022-09-29 15:32:32 IST
-169 1bba41cc7f044c69b8123fccc33bae34 Thu 2022-09-29 15:40:47 IST-Thu 2022-09-29 15:48:18 IST
-168 857a98f1557541fca7cd83347be055ab Thu 2022-09-29 15:52:01 IST-Thu 2022-09-29 18:31:57 IST
-167 d1d049efb4964ef4993317e3e473a4bf Thu 2022-09-29 18:40:51 IST-Sun 2022-10-02 10:30:44 IST
-166 00f87f6fe46747c4b30da2b3589a4a6b8 Sun 2022-10-02 10:34:45 IST-Mon 2022-10-03 08:34:58 IST
-165 a0d973091b3e4d7f83f3fed3187af3c9 Mon 2022-10-03 08:40:21 IST-Tue 2022-10-04 01:31:26 IST
-164 ea528d7ad81e4758bd72c2e8fa50f7d3 Tue 2022-10-04 02:16:06 IST-Wed 2022-10-05 17:44:19 IST
-163 54c42ae2798f4be0a4ac8cc9923f2618 Wed 2022-10-05 18:19:15 IST-Thu 2022-10-06 01:19:47 IST
-162 4b143d25d16749eeab0fda5bc4b02e5d Thu 2022-10-06 01:46:02 IST-Thu 2022-10-06 13:30:09 IST
-161 449e06e8c094590b29b9fda17023080b Thu 2022-10-06 14:07:49 IST-Sun 2022-10-09 09:17:01 IST
-160 d06500c55d6d4b71b3374a5e9b11bb73 Sun 2022-10-09 09:47:56 IST-Sun 2022-10-09 18:34:57 IST
-159 4125272a640e4071b50a8d7309012917 Sun 2022-10-09 19:05:47 IST-Sun 2022-10-09 22:32:48 IST
-158 0bf60cf7c94e499595bf76dca559a253 Sun 2022-10-09 22:40:55 IST-Mon 2022-10-10 05:17:01 IST
-157 a770ffdc3cb1b498eaeab92aa0ae734bf Mon 2022-10-10 05:31:56 IST-Mon 2022-10-10 05:37:42 IST
-156 fd344fa611c649a79174492540af17f9 Mon 2022-10-10 05:48:57 IST-Mon 2022-10-10 06:29:19 IST
-155 8aafef38d79a435193a476380035f546 Mon 2022-10-10 06:53:40 IST-Mon 2022-10-10 08:17:01 IST
-154 49472aa0a7a24438a46e6c5fa53c301d Mon 2022-10-10 08:34:30 IST-Mon 2022-10-10 13:38:40 IST
-153 2ea7196459884c259fc4914ce6743617 Mon 2022-10-10 14:08:53 IST-Tue 2022-10-11 23:17:01 IST
-152 f914b44aa54144c887a507855a0431ef Tue 2022-10-11 23:25:55 IST-Tue 2022-10-11 23:33:14 IST
-151 866bf074eedb43218af30d89ec0580ba Tue 2022-10-11 23:35:03 IST-Thu 2022-10-13 02:24:43 IST
-150 41a04f8ad36444f39d231b38add5f0ea Thu 2022-10-13 02:34:43 IST-Thu 2022-10-13 02:37:07 IST
-149 c0d022635034280a0818fa5ffacdb35 Thu 2022-10-13 02:40:29 IST-Fri 2022-10-14 22:47:11 IST
-148 6c7a07a784694d268bce00e00a05cdda Fri 2022-10-14 22:47:58 IST-Sat 2022-10-15 12:55:13 IST
-147 7d548712d46a471f96f71b8af16c729bd Sat 2022-10-15 13:13:58 IST-Sun 2022-10-16 05:17:40 IST
-146 9b0adb9294eb4c169210845e35e7e223 Mon 2022-10-17 12:51:22 IST-Wed 2022-10-26 06:22:36 IST
-145 4931f5775b18429e84dc826d4392ddb Wed 2022-10-26 06:32:45 IST-Thu 2022-10-27 03:43:52 IST
-144 4018be0824b1464c9ede6f2da234e4dd Thu 2022-10-27 04:24:58 IST-Sat 2022-11-05 17:02:35 IST
-143 e700750786044436bae61ddfdded19f08 Sat 2022-11-05 17:03:29 IST-Tue 2022-11-08 07:48:03 IST
-142 a955a5fd1b9b46c388f0ea38ac41ac2f Tue 2022-11-08 07:55:25 IST-Fri 2022-11-11 11:35:06 IST
-141 28a4a58bf7db4faaa0fcfe507ebd93a7 Fri 2022-11-11 12:20:45 IST-Fri 2022-11-11 13:23:41 IST
-140 00a0ea7aaefda4179a7f1a9d4e257128 Fri 2022-11-11 13:33:28 IST-Mon 2022-11-14 06:59:55 IST
-139 5cbbf4f995024d5d85c7714f4913b03e Mon 2022-11-14 07:08:54 IST-Tue 2022-11-15 14:40:11 IST
-138 1dff8d7c9ac343d592869d63a39a8879 Tue 2022-11-15 15:01:02 IST-Wed 2022-11-23 18:42:06 IST
-137 b6c5d25c07ca4de3940a9db36543ad71 Wed 2022-11-23 18:43:00 IST-Fri 2022-12-09 16:46:55 IST
-136 4e6702ad4730167f0463f03205f4b633 Fri 2022-12-10 14:54:43 IST-Sat 2022-12-10 14:56:15 IST

```

A specific boot can be analysed (to check for normal reboot/boot/boot after crash etc) using the command *journalctl -b {num} -n*

Here {num} will be the index given in *journalctl --list-boots* command in the first column.


```

Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for core18, revision 2708.
Mar 20 13:24:42 stark99 systemd[1]: snap-gnome\x2d3\x2d34\x2d1804-72.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for gnome-3-34-1804, revision 72.
Mar 20 13:24:42 stark99 systemd[1]: snap-core18-2714.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for core18, revision 2714.
Mar 20 13:24:42 stark99 systemd[1]: snap-gtk\x2dcommon\x2dthemes-1534.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for gtk-common-themes, revision 1534.
Mar 20 13:24:42 stark99 systemd[1]: snap-bare-5.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for bare, revision 5.
Mar 20 13:24:42 stark99 systemd[1]: snap-core20-1822.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for core20, revision 1822.
Mar 20 13:24:42 stark99 systemd[1]: snap-gnome\x2d3\x2d38\x2d2004-119.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for gnome-3-38-2004, revision 119.
Mar 20 13:24:42 stark99 systemd[1]: snap-gnome\x2d3\x2d34\x2d1804-77.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for gnome-3-34-1804, revision 77.
Mar 20 13:24:42 stark99 systemd[1]: snap-gtk\x2dcommon\x2dthemes-1535.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for gtk-common-themes, revision 1535.
Mar 20 13:24:42 stark99 systemd[1]: snap-snap\x2dstore-599.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for snap-store, revision 599.
Mar 20 13:24:42 stark99 systemd[1]: snap-snapd-17950.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for snapd, revision 17950.
Mar 20 13:24:42 stark99 systemd[1]: snap-snapd-18357.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for snapd, revision 18357.
Mar 20 13:24:42 stark99 systemd[1]: snap-vlc-2344.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for vlc, revision 2344.
Mar 20 13:24:42 stark99 systemd[1]: snap-vlc-3078.mount: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Unmounted Mount unit for vlc, revision 3078.
Mar 20 13:24:42 stark99 systemd[1]: Stopped target Local File Systems (Pre).
Mar 20 13:24:42 stark99 systemd[1]: Reached target Unmount All Filesystems.
Mar 20 13:24:42 stark99 systemd[1]: Stopping Monitoring of LVM2 mirrors, snapshots etc. using dmeventd or progress polling...
Mar 20 13:24:42 stark99 systemd[1]: systemd-tmpfiles-setup-dev.service: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Stopped Create Static Device Nodes in /dev.
Mar 20 13:24:42 stark99 systemd[1]: systemd-sysusers.service: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Stopped Create System Users.
Mar 20 13:24:42 stark99 systemd[1]: systemd-remount-fs.service: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Stopped Remount Root and Kernel File Systems.
Mar 20 13:24:42 stark99 systemd[1]: lvm2-monitor.service: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Stopped Monitoring of LVM2 mirrors, snapshots etc. using dmeventd or progress polling.
Mar 20 13:24:42 stark99 systemd[1]: Reached target Shutdown.
Mar 20 13:24:42 stark99 systemd[1]: Reached target Final Step.
Mar 20 13:24:42 stark99 systemd[1]: systemd-poweroff.service: Succeeded.
Mar 20 13:24:42 stark99 systemd[1]: Finished Power-Off.
Mar 20 13:24:42 stark99 systemd[1]: Reached target Power-Off.
Mar 20 13:24:42 stark99 systemd[1]: Shutting down.
Mar 20 13:24:42 stark99 systemd-shutdown[1]: Syncing filesystems and block devices.
Mar 20 13:24:42 stark99 systemd-shutdown[1]: Sending SIGTERM to remaining processes...
Mar 20 13:24:42 stark99 dnsmasq[1950]: exiting on receipt of SIGTERM
Mar 20 13:24:42 stark99 dnsmasq[2073]: exiting on receipt of SIGTERM
Mar 20 13:24:42 stark99 dnsmasq[1830]: exiting on receipt of SIGTERM
Mar 20 13:24:42 stark99 systemd-journal[371]: Journal stopped

```

Interface change:

The logs for network interface change status can be found using the command *journalctl -u NetworkManager*

```

Apr 22 15:59:20 stark99 NetworkManager[802]: <info> [1650623360.5688] manager: (virbr0): new Bridge device (/org/freedesktop/NetworkManager/Devices/4)
Apr 22 15:59:20 stark99 NetworkManager[802]: <info> [1650623360.5701] manager: (virbr0-nic): new Tun device (/org/freedesktop/NetworkManager/Devices/5)
Apr 22 15:59:20 stark99 NetworkManager[802]: <info> [1650623360.5869] device (virbr0-nic): state change: unmanaged -> unavailable (reason 'connection-assumed', sys-iface-state: 'external')
Apr 22 15:59:20 stark99 NetworkManager[802]: <info> [1650623360.5877] device (virbr0-nic): state change: unavailable -> disconnected (reason 'none', sys-iface-state: 'external')
Apr 22 15:59:21 stark99 NetworkManager[802]: <info> [1650623361.0444] device (virbr0): state change: unmanaged -> unavailable (reason 'connection-assumed', sys-iface-state: 'external')
Apr 22 15:59:21 stark99 NetworkManager[802]: <info> [1650623361.0465] device (virbr0): state change: unavailable -> disconnected (reason 'connection-assumed', sys-iface-state: 'external')
Apr 22 15:59:21 stark99 NetworkManager[802]: <info> [1650623361.0470] device (virbr0): Activation: starting connection 'virbr0' (a42ceb1-b377-411f-9c2b-a5e7651eb60a)
Apr 22 15:59:21 stark99 NetworkManager[802]: <info> [1650623361.0472] device (virbr0): state change: disconnected -> prepare (reason 'none', sys-iface-state: 'external')
Apr 22 15:59:21 stark99 NetworkManager[802]: <info> [1650623361.0475] device (virbr0): state change: prepare -> config (reason 'none', sys-iface-state: 'external')

```

d. Test Observations:

The SMF logs all the Security events with a unique system reference (e.g. host name, IP or MAC address) and the exact time the incident occurred, along with other details as mentioned in the table above.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_AUDIT_EVENT_GENERATION		

2.5.3 TSTP for Evaluation of Secure Log Export

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.5.3
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 5 - User Audit
2. **<Security Requirement No & Name >** 2.5.3 Secure Log Export.
3. **<Requirement Description: >**
 - a) The SMF shall support forwarding of security event logging data to an external system available in redundant configuration by push or pull mechanism through diverse links.
 - b) Log functions should support secure uploading of log files to a central location or to a system external for the SMF.
 - c) SMF shall be able to store the generated audit data locally. The memory for this purpose shall be dimensioned to cater for the continuous storage of two days of audit data. OEM shall submit justification document for sufficiency of local storage requirement.
 - d) Secure Log export shall comply the secure cryptographic controls prescribed in Table 1 of the latest document "Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)" only.

[Ref: EC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.6.2]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal
```

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Check for the TLS version supported by the DUT:

For TLS on Ubuntu 20.04:

Use the following commands to check supported cipher suites:

openssl ciphers -tls1_3 -s

openssl ciphers -tls1_2 -s

```
siddhesh@stark99:~/Desktop/ca-cert$ openssl ciphers -tls1_3 -s
TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256
siddhesh@stark99:~/Desktop/ca-cert$ openssl ciphers -tls1_2 -s
ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES256-SHA256:ECDSA-AES128-SHA256:ECDSA-AES256-SHA256:ECDSA-AES128-SHA256:DHE-RSA-AES128-SHA256:ECDSA-AES256-SHA:ECDSA-AES256-SHA:DHE-RSA-AES256-SHA:ECDSA-AES128-SHA:ECDSA-AES128-SHA:DHE-RSA-AES128-SHA:AES128-SHA:AES256-GCM-SHA384:AES128-GCM-SHA256:AES256-SHA256:AES128-SHA256:AES256-SHA:AES128-SHA
siddhesh@stark99:~/Desktop/ca-cert$
```

6. **Preconditions:**

- The manufacturer shall list the standard protocols which transfer security event logging data in the Vendor Documentation.
- The Vendor must provide the documentation for the central location / external storage specifying the storage location for log files.
- The session between network product and central location or external system for network product log functions has been set up.
- The tester has privilege to operate network product and related logs can be outputted.

7. **Test Objective:** To verify that logs are securely transferred to the centralized storage.

8. **Test Plan:**

a. **Test Bed Diagram:**

b. **Tools Used:**

c. **Test Execution Steps:**

1. The tester configures the network product to forward event logs to an external system (according to bullet a) of requirement) and related logs are sent out.
2. The tester checks whether the used transport protocol is secure protocol.
3. The tester checks whether the central location or external system for network product log functions has stored the related logs.
4. The tester configures the network product for secure upload of event log files to an external system (according to bullet b) of requirement) and performs a log file upload.
5. The tester checks whether the used transport protocol for log file upload is a secure standard protocol.
6. The tester checks whether the central location or external system for network product log functions has stored the related logs.

9. **Expected Results to Pass:**

- The listed transport protocols are secure protocols.
- The used transport protocol for log file upload is a secure standard protocol.
- The tester finds that the central location or external system for network product log functions has stored the related logs.

10. **Expected Format of Evidence:** Screenshot/packet trace of evidence for:

- Listed transport protocols are secure protocols.
- The transport protocol for log file upload is a secure standard protocol.
- The central location or external system has stored the related the logs.

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC_LOG TRANS_TO_CENTR_STORAGE

b. **Test Execution Steps:**

1. The tester configures the network product to forward event logs to an external system (according to bullet a) of requirement) and related logs are sent out.
2. The tester checks whether the used transport protocol is secure protocol. Here we capture the packet trace between Network Product and central storage/external system and check whether the log transfer is protected using a secure transport protocol (here, syslog within TLS).

Positive case: syslog traffic protected with TLS

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.16.1.117	172.16.1.130	TLSv1.3	305	Client Hello
2	0.002414	172.16.1.130	172.16.1.117	TLSv1.3	1139	Server Hello, Change Cipher Spec, Application Data, Applicati...
3	0.007418	172.16.1.117	172.16.1.130	TLSv1.3	146	Change Cipher Spec, Application Data
4	0.010377	172.16.1.130	172.16.1.117	TLSv1.3	345	Application Data
5	0.010497	172.16.1.130	172.16.1.117	TLSv1.3	345	Application Data
6	1.300780	172.16.1.117	172.16.1.130	TLSv1.3	90	Application Data
7	7.374144	172.16.1.117	172.16.1.130	TLSv1.3	680	Client Hello, Change Cipher Spec, Application Data
8	7.380894	172.16.1.130	172.16.1.117	TLSv1.3	311	Server Hello, Change Cipher Spec, Application Data, Applicati...
9	7.383140	172.16.1.117	172.16.1.130	TLSv1.3	166	Application Data, Application Data
10	7.384521	172.16.1.130	172.16.1.117	TLSv1.3	345	Application Data
11	8.907096	172.16.1.117	172.16.1.130	TLSv1.3	90	Application Data

Frame 11: 90 bytes on wire (720 bits), 90 bytes captured (720 bits)	
Ethernet II, Src: VMware_vd:61:7a (00:0c:29:dd:61:7a), Dst: Raspberr_45:99:91 (b8:27:eb:45:99:91)	
Internet Protocol Version 4, Src: 172.16.1.117, Dst: 172.16.1.130	
Transmission Control Protocol, Src Port: 35122, Dst Port: 4433, Seq: 715, Ack: 525, Len: 24	
Transport Layer Security	
- TLSv1.3 Record Layer: Application Data Protocol: Application Data	
Opaque Type: Application Data (23)	
Version: TLS 1.2 (0x0303)	
Length: 19	
Encrypted Application Data: 4a46614a4c92c2bba9729b1f4379471b869349	

Negative case: syslog traffic in plain text

No.	Time	Source	Destination	Protocol	Length	Info
50262	-12978.20740	192.168.126.123	192.168.126.123	Syslog	68	USER.INFO: I was in Manchester\000
50264	-12978.20734	192.168.126.123	192.168.126.123	Syslog	68	USER.INFO: I was in Manchester\000
50266	-12978.20731	192.168.126.123	192.168.126.123	Syslog	68	USER.INFO: I was in Manchester\000
54220	-12886.75052	192.168.126.123	192.168.126.123	Syslog	68	USER.ERR: I was in Manchester\000
54222	-12886.75046	192.168.126.123	192.168.126.123	Syslog	68	USER.ERR: I was in Manchester\000
54224	-12886.75043	192.168.126.123	192.168.126.123	Syslog	68	USER.ERR: I was in Manchester\000

Note: The tester should test for the corresponding protocol mentioned in the OEM documentation.

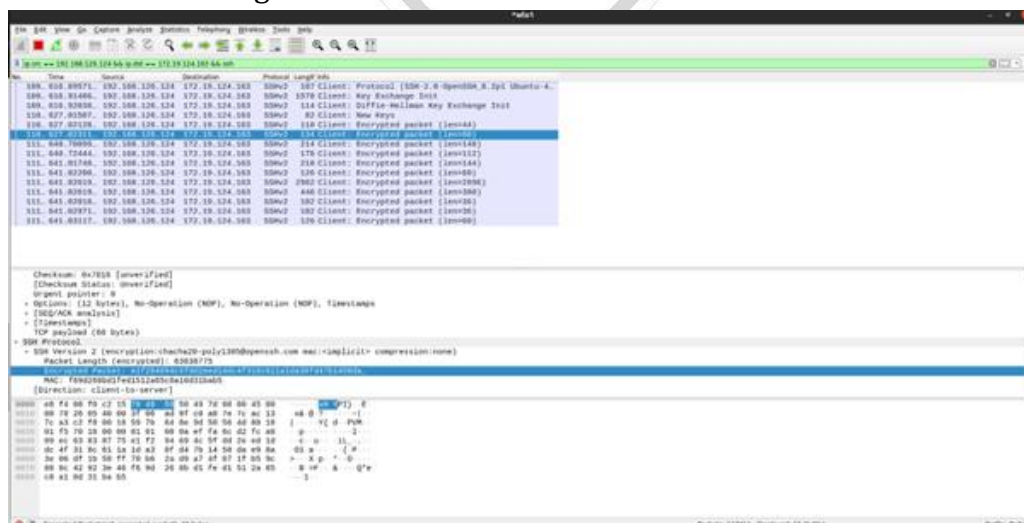
3. The tester checks whether the central location or external system for network product log functions has stored the related logs.

Logs stored in syslog.log file at server, for the user stark99.

```
siddhesh@stark99: ~/Desktop/ca-cert
root@stark99:/var/log# cd stark99/
root@stark99:/var/log/stark99# ls
syslog.log
root@stark99:/var/log/stark99#
```

Note: tester should refer the location specified in the documentation of external system/central storage to get storage path for logs.

4. The tester configures the network product for secure upload of event log files to an external system (according to bullet b) of requirement) and performs a log file upload.
5. The tester checks whether the used transport protocol for log file upload is a secure standard protocol. Here the log function uses **scp** protocol to securely transfer log files to the central storage.



Note: The tester should test for the corresponding protocol mentioned in the OEM documentation.

6. The tester checks whether the central location or external system for network product

log functions has stored the related logs.

```
siddhesh@stark99: ~/Desktop/ca-cert
root@stark99:/var/log# cd stark99/
root@stark99:/var/log/stark99# ls
syslog.log
root@stark99:/var/log/stark99#
```

12. Test Results:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_LOGTRANS_TO_CENTR_STORAGE		



2.5.4 TSTP for Evaluation of Logging access to personal data

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.5.4
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 5: User Audit
2. **<Security Requirement No & Name>** 2.5.4 Logging access to personal data
3. **<Requirement Description>** In some cases, access to personal data in a clear text might be required. If such access is required, access to this data shall be logged, and the log shall contain who accessed what data without revealing personal data in clear text. When for practical purposes, such logging is not available, a coarser grain logging is allowed. In some cases, the personal data stored in the log files may allow the direct identification of a subscriber. In such cases, the revealed personal information may not expose the subscriber to any kind of privacy violation.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.2.5]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

- With the command "`dpkg -l auditd`", we can check that the tool is installed or not

```
(base) unnati@unnati-Super-Server:~$ dpkg -l auditd
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/half-conf/Half-inst/trig-aWait/Trig-pend
|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name           Version             Architecture Description
+++-----+-----+-----+-----+
ii  auditd           1:3.0.7-1build1    amd64        User space tools for security auditing
(base) unnati@unnati-Super-Server:~$
```

- To store user identity in the logs we must make sure that this option is enabled in the configuration file of **auditd.conf**

log_format = ENRICHED

Can check using the command **sudo gedit /etc/audit/auditd.conf**

```

Open  auditd.conf
      /etc/audit
1 #
2 # This file controls the configuration of the audit daemon
3 #
4
5 local_events = yes
6 write_logs = yes
7 log_file = /var/log/audit/audit.log
8 log_group = adm
9 log_format = ENRICHED
10 flush = INCREMENTAL_ASYNC
11 freq = 50
12 max_log_file = 8

```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2dea1d08af1844429d30  AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

6. **Preconditions:** A document which provides a description of where personal data in clear text is accessible on the network product, how it can be accessed, and details of where such access attempts are logged and how to view these logs.
7. **Test Objective:** Verify that in cases where a network product presents personal data in clear text, access attempts to such data are logged and the log information includes the user identity that has accessed the data. The test case also verifies that the personal data itself is not included in clear text in the log.

8. **Test Plan:**

8.1 **Test Setup Diagram:**



8.2 **Tools Used:** Auditd

8.3 **Test Execution Steps:**

- ✗ Check that all pre-conditions are met.
- ✗ The tester needs to login to the network product.
- ✗ Check the personal data (we will get the location of this data in the documentation provided).
- ✗ Now check if this is being logged or not.
- ✗ We can check the logs using the tool **auditd**
 - The logs will be stored in **/var/log/audit/** folder
 - The tester can move to the folder and check if access to file is being logged or not.
- ✗ In the logs the identity of the user needs to be visible, not the data.
- ✗ The tester repeats the check for each case where personal data is accessible.

Note1: - to check the logs of different files, we have to check the logs at the same folder only.

Note2: - If in any logs the subscriber identity is revealed, then the personal information of the subscriber should not be there. It should be checked by the tester. And the location of these logs should be provided by OEM.

9. **Expected Results for Pass:** All access attempts to personal data (in clear text) are recorded in the described logs, with the user identity included and no personal data visible in the log.

10. **Expected Format of Evidence:** Sample copies of the log files

11. **Test Execution:**

➤ **Test Case Number:** 01

- a. **Test Case Name:** TC1_LOGGING_ACCESS_TO_PERSONAL_DATA
- b. **Test Case Description:** To verify that the log for personal data is maintained and personal data is not visible.
- c. **Execution Steps:** Here we have tested for a file named passwd stored at **/etc/passwd**

When the tester tries to open the file, it will be logged

Here we can check the path and the identity of the user is being logged not the data

To check this, we can move to folder using the command:

cd /var/log/audit/

After this it will depend on different DUT configurations, where we can find the specified log files, as there will be multiple files for storing logs.

Note: here it is in audit.logs file

2.6.1 TSTP for Cryptographic Based Secure Communication

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.6.1
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 6: Data Protection
2. **<Security Requirement No & Name >** 2.6.1 Cryptographic Based Secure Communication
3. **<Requirement Description: >** SMF shall Communicate with the connected entities strictly using the secure cryptographic controls prescribed in Table 1 of the latest document "Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)" only.

OEM shall submit to TSTP, the list of the connected entities with SMF and the method of secure communication

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)


```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: **cat /etc/os-release** (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)


```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

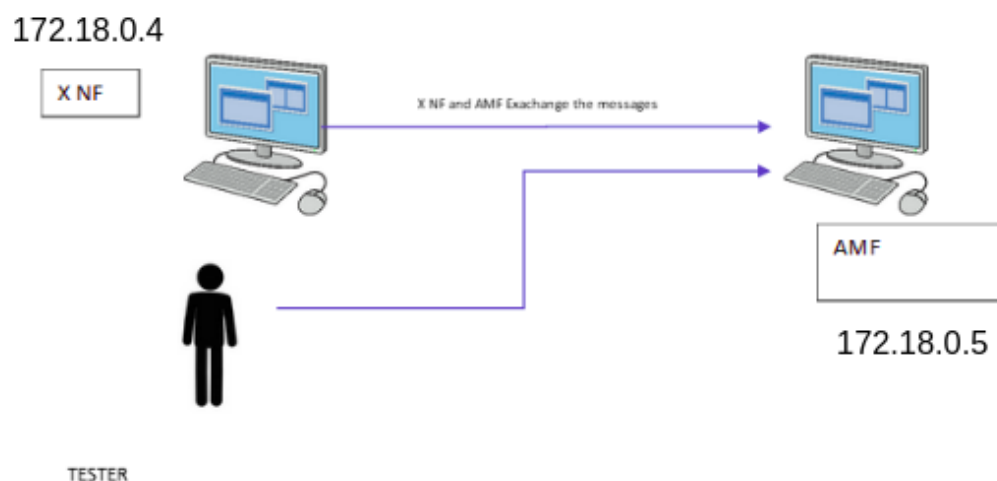
6. Preconditions

- Wireshark should be installed to view the messages exchanged between X NF and SMF, where X NF acts as tester device.
- IP Address of all the connected entities should be available before start of the test.
- OEM shall submit document with details regarding connected entities to DUT and all the methods of secure communication used.

7. **Test Objective:** To check that DUT communicates with the connected entities strictly using the secure cryptographic controls prescribed in Table 1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only.

8. Test Plan:

8.1 Test Setup Diagram:



8.2 Tools Used: Wireshark on Tester Device

8.3 Test Execution Steps

- The Tester identifies one of the secure communication protocols used as per OEM Documentation.
- Wireshark is launched on Tester Device and packet capture begins
- The Tester establishes connection with DUT from Tester Device.
- The Tester Verifies from Packet Trace that DUT communicates with the connected entities strictly using the secure cryptographic controls prescribed in Table 1 of the latest

document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only.

9. **Expected Results for Pass:-** Evidence that DUT communicates with the connected entities strictly using the secure cryptographic controls prescribed in Table 1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only.

10. **Expected Format of Evidence:-** Screenshots of packet trace and pcap file.

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC1_CRYPTOGRAHIC_BASED_SECURE_COMMUNICATION

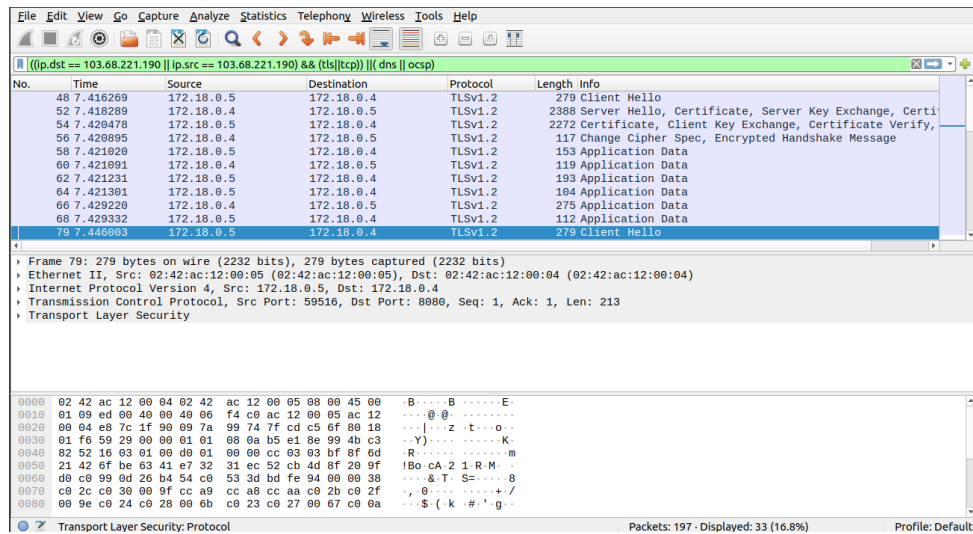
b. **Test Case Description:** As per the mentioned requirement DUT and X NF should support secure communication.

We demonstrate the steps to be followed in the case of TLS1.2/1.3 (This is only sample tester must test all communication methods)

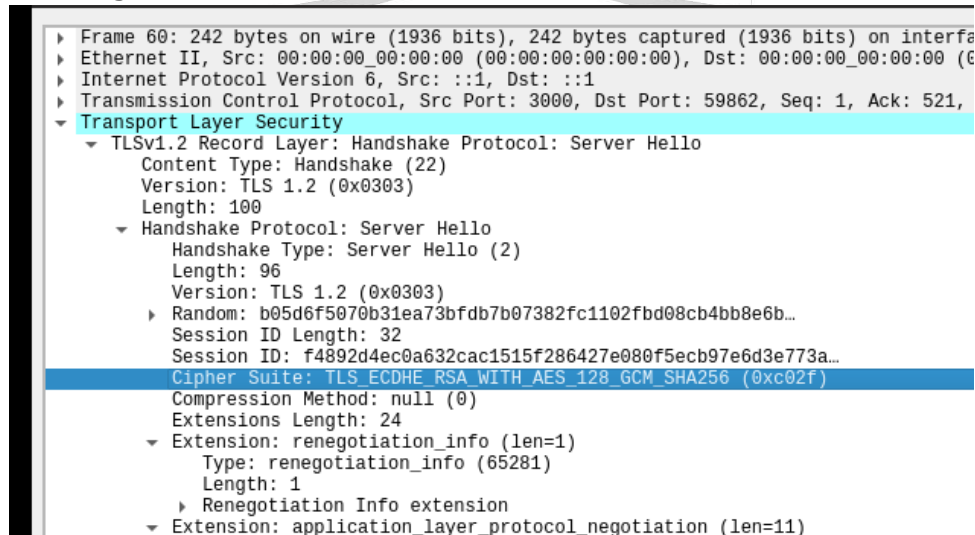
c. **Execution Steps:**

- Launch the Wireshark app to capture the packet transfer between SMF and X NF
- Establish the request and response between SMF and X NF.
- Log the transfer of packets in the pcap file.

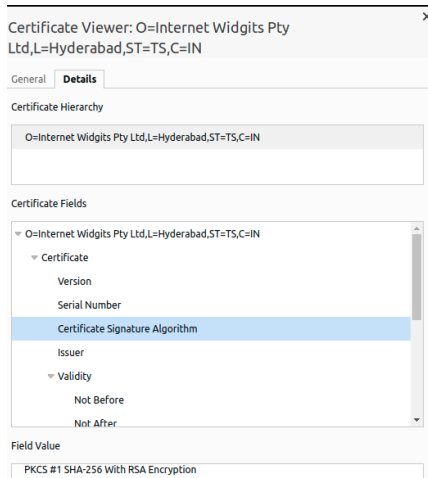
➤ **Case 1:** In case DUT is using TLS1.2/1.3, we see successfully checking the certificate of X NF. This ensures that the X NF and SMF are using Transport Layer Security. Below image can be used as a reference for understanding the flow of the connection that is established between X NF and SMF. By the snapshot of the Wireshark, we deduce the following information. client hello message followed by server hello. This is followed by Certificate and Key Exchange at both application SMF and X NF side. After the Encrypted Handshake the Application data is starts to exchange. All these exchanges of messages happen by Using TLS 1.2 Protocol. Hence the above requirement is satisfied.



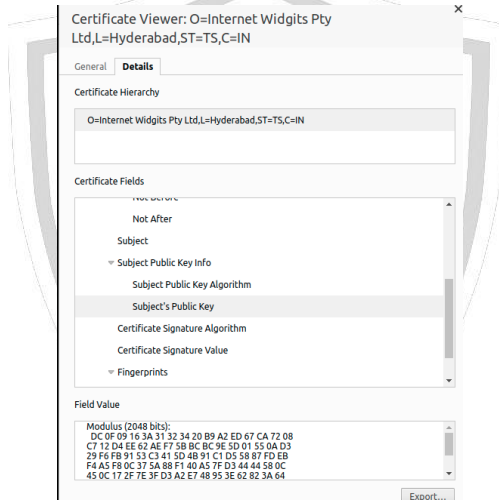
- Click on the TLS by clicking on the server_hello packet. Check the Cipher suite attribute. See Below Figure for the reference.



- Under the certificate Viewer check the Certificate Signature Algorithm it should be as per TABLE1 of the mentioned requirement.



- Under the Certificate Viewer check the Subject's Public Key length and that should be as per TABLE1 of the above requirement.



- Under the TLS1.2 Record Layer check for the attribute of Encrypted application Data.

2.6.2 TSTP for Cryptographic Module Security Assurance

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.6.2
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 6: Data Protection
2. **<Security Requirement No & Name >** 2.6.2 Cryptographic Module Security Assurance
3. **<Requirement Description: >**

Cryptographic module embedded inside the SMF (in the form of hardware, software or firmware) that provides all the necessary security services such as authentication, integrity and confidentiality is designed and implemented in compliance with FIPS 140-2 or later as prescribed by NIST standards.

Till further instructions, this clause will be considered 'complied' by submission of an undertaking by the OEM in specified format. An undertaking is to be submitted by the OEM mentioning that "Cryptographic module embedded inside the SMF (in the form of hardware, software or firmware) that provides all the necessary security services such as authentication, integrity and confidentiality is designed and implemented in compliance with FIPS 140-2 or later as prescribed by NIST standards."

4. **DUT Confirmation Details:** An undertaking is to be submitted by the OEM mentioning that "Cryptographic module embedded inside the SMF (in the form of hardware, software or firmware) that provides all the necessary security services such as authentication, integrity and confidentiality is designed and implemented in compliance with FIPS 140-2 or later as prescribed by NIST standards."
5. **DUT Configuration:** Cryptographic module embedded inside the SMF (in the form of hardware, software or firmware) that provides all the necessary security services such as

authentication, integrity and confidentiality is designed and implemented in compliance with FIPS 140-2 or later as prescribed by NIST standards.

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2dea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:-** The cryptographic module testing document and the detailed self / Lab test report along with test results should be available at the TSTP. (This Should be submitted in appropriate format expected by TSTP). An undertaking is to be submitted by the OEM mentioning that “Cryptographic module embedded inside the SMF (in the form of hardware, software or firmware) that provides all the necessary security services such as authentication, integrity and confidentiality is designed and implemented in compliance with FIPS 140-2 or later as prescribed by NIST standards” should be present with tester at TSTP

7. **Test Objective:** To check that Cryptographic module embedded inside the DUT (in the form of hardware, software or firmware) provides all the necessary security services such as authentication, integrity and confidentiality is designed and implemented in compliance with FIPS 140-2 or later as prescribed by NIST standards.

8. **Test Plan:**

8.1 **Test Setup Diagram:**

8.2 **Tools Used:**

8.3 **Test Execution Steps**

- The Tester will scrutinise the test report submitted by the OEM to check whether SMF is FIPS 140-2 or later as prescribed by NIST standards compliant or not.
- Tester will also verify the Undertaking submitted by the OEM for the same.

9. **Expected Results for Pass:** Successful verification of Undertaking submitted by the OEM

10. **Expected Format of Evidence:** An undertaking submitted by the OEM mentioning that “Cryptographic module embedded inside the SMF (in the form of hardware, software or firmware) that provides all the necessary security services such as authentication, integrity

and confidentiality is designed and implemented in compliance with FIPS 140-2 or later as prescribed by NIST standards” should be present with tester at TSTP.

11. **Test Execution:**

➤ **Test Case Number:** 01

a. Test Case Name: TC1_CRYPTO_MODULE_SECURITY

b. Test Case Description:

- To verify the undertaking.

c. Execution Steps:

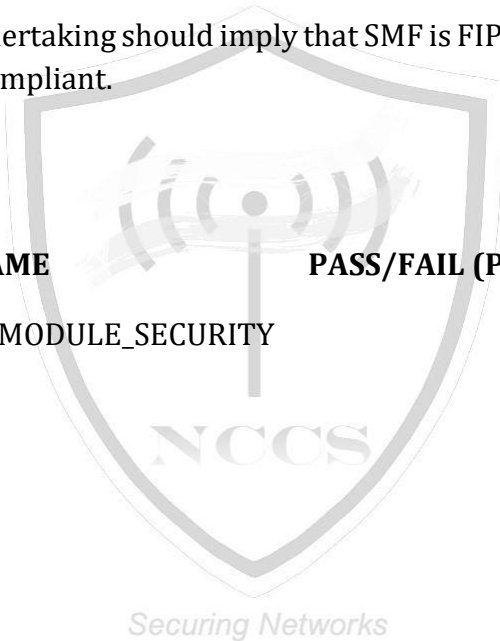
- The Tester will scrutinise the test report submitted by the OEM to check whether SMF is FIPS 140-2 or later as prescribed by NIST standards compliant or not.
- Tester will also verify the Undertaking submitted by the OEM for the same.

d. Test Observations:

- Test Reports and Undertaking should imply that SMF is FIPS 140-2 or later as prescribed by NIST standards compliant.

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL (Please see Note Below)
1	TC1_CRYPTO_MODULE_SECURITY	



2.6.3 TSTP for Cryptographic Algorithms implementation Security Assurance

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.6.3
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) **<ITSAR Section No & Name>** Section 6: Data Protection
- 2) **<Security Requirement No & Name >** 2.6.4 Cryptographic Algorithms implementation Security Assurance
- 3) **<Requirement Description: >** Cryptographic algorithm implemented inside the Crypto module of SMF shall be in compliance with the respective FIPS standards (for the specific crypto algorithm).

Till further instructions, this clause will be considered 'complied' by submission of an undertaking by the OEM in specified format.

An undertaking is to be submitted by the OEM mentioning that "Cryptographic algorithms implemented inside the Crypto module of SMF is in compliance with the respective FIPS standards (for the specific crypto algorithm embedded inside the SMF)."

- 4) **DUT Confirmation Details:** An undertaking is to be submitted by the OEM mentioning that "Cryptographic algorithms implemented inside the Crypto module of SMF is in compliance with the respective FIPS standards (for the specific crypto algorithm embedded inside the DUT)."

- 5) **DUT Configuration:** Cryptographic algorithm implemented inside the Crypto module of DUT shall be in compliance with the respective FIPS standards (for the specific crypto algorithm).

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

- 6) **Preconditions** The cryptographic module testing document and the detailed self / Lab test report along with test results should be available at the TSTP. (This Should be submitted in appropriate format expected by TSTP). An undertaking is to be submitted by the OEM mentioning that "Cryptographic algorithms implemented inside the Crypto module of SMF is in compliance with the respective FIPS standards (for the specific crypto algorithm embedded inside the DUT)" should be present with tester at TSTP
- 7) **Test Objective:** To check that Cryptographic algorithm implemented inside the Crypto module of DUT shall be in compliance with the respective FIPS standards (for the specific crypto algorithm).
- 8) **Test Plan:**
- 8.1 **Test Setup Diagram:**
 - 8.2 **Tools Used:**
 - 8.3 **Test Execution Steps**
 - The Tester will scrutinise the test report submitted by the OEM to check whether DUT is in compliance with the respective FIPS standards (for the specific crypto algorithm).
 - Tester will also verify the Undertaking submitted by the OEM for the same.
- 9) **Expected Results for Pass:** Successful verification of Undertaking submitted by the OEM
- 10) **Expected Format of Evidence:** An undertaking_submitted by the OEM mentioning that "Cryptographic algorithms implemented inside the Crypto module of SMF is in compliance with the respective FIPS standards (for the specific crypto algorithm embedded inside the DUT)"
- 11) **Test Execution:**
- **Test Case Number:** 01
 - a) **Test Case Name:** TC1_ CRYPTO_ALGO_SECURITY
 - b) **Test Case Description:**
 - To verify the undertaking.
 - c) **Execution Steps:**
 - The Tester will scrutinise the test report submitted by the OEM to check whether DUT is in compliance with the respective FIPS standards (for the specific crypto algorithm).

- Tester will also verify the Undertaking submitted by the OEM for the same.

a. **Test Observations:**

- Test Reports and Undertaking should imply that DUT is in compliance with the respective FIPS standards (for the specific crypto algorithm).

12) Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL (Please see Note Below)
1	TC1_ CRYPTO_ALGO_SECURITY	



2.6.4 TSTP For Protecting data and information – Confidential System Internal Data

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.6.4
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<TSTP Document ID:>
<Applicant Name:> Ex: XYZ
<Application Number>
<DUT Details: > Ex: Router
<Digest Hash of OS>
<Digest Hash of Configuration>
<Applicable ITSAR: >
<ITSAR Version No:>
<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 6 – Data Protection
2. **<Security Requirement No & Name >** 2.6.4 Protecting data and information – Confidential System Internal Data
3. **<Requirement Description: >**
 - a) When NF is in normal operational mode (i.e., not in maintenance mode) there shall be no system function that reveals confidential system internal data in the clear text to users and administrators.
 - b) Access to maintenance mode shall be restricted only to authorized privileged users.

[Reference: TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.2.2.]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

For SSH:

command used: **ssh -V** (To get version information)

```
amf@localhost $ ssh -V
OpenSSH_8.9p1 Ubuntu-3ubuntu0.1, OpenSSL 3.0.2 15 Mar 2022
```

For Maintenance Mode in Linux and GRUB Bootloader,

Open the GRUB configuration file with a text editor,

Command Used: **sudo nano /etc/default/grub**

Look for the line containing "GRUB_CMDLINE_LINUX" and check if it has "single" at the end.

This enables maintenance mode access only with the root password.

```
GRUB_DEFAULT=0
GRUB_TIMEOUT_STYLE=hidden
GRUB_TIMEOUT=0
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
GRUB_CMDLINE_LINUX="...single"
```

6. Preconditions:

- The vendor shall provide documentation describing how confidential system internal information that could possibly be revealed in clear-text is handled by system functions.
- A list of all system functions in the network product, information on how to enable and execute them should be provided as a part of the vendor's documentation. A system function is every function implemented in the network product needed by the services/functionalities provided by the network product itself

7. Test Objective/ Purpose: Verify that no system function reveals sensitive data in the clear and only root user has access to maintenance mode.

Securing Networks

8. Test Plan:

8.1 Number of Test Scenarios:

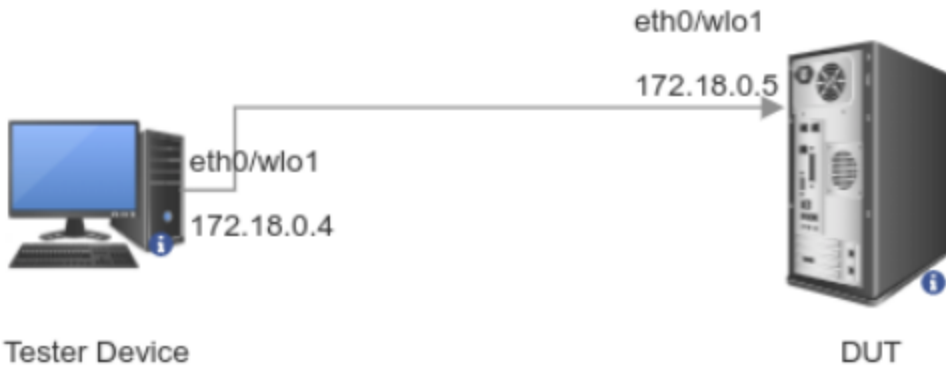
8.1.1 **Test Scenario for Remote Login using SSH:**

This test scenario is regarding Local or Remote CLI and conf files (Additional Test scenarios based on the OEM document)

8.1.2 **Test Scenario for checking only root user has access to maintenance mode:**

This test scenario is regarding restricting authorized users to access maintenance mode.

8.2 Test Bed Diagram:



8.3 **Test Execution Step:**

- Review the documentation provided by the vendor describing how confidential system internal information is handled by system functions.
- The tester checks whether any system functions as described in the product documentation (e.g. local or remote OAM CLI or GUI, logging messages, alarms, error messages, configuration file exports, stack traces) reveal any confidential system internal data in the clear (for example, passphrases).

9. **Expected Results:** There should be no confidential system internal data revealed in the clear by any system function.

10. **Expected Format of Evidence:** Evidence suitable for the interface, e.g. screenshot containing the operational results.

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC1_CONFIDENTIAL_SYSTEM_INTERNAL_DATA

b. **Test Case Description:** Tests need to be conducted to ensure that no system function should reveal confidential system internal data in the clear text to users and administrators. Such functions could be, for example, local or remote OAM CLI or GUI, logging messages, alarms, configuration file exports etc. This test case is for local and remote CLI and conf files.

c. **Execution Steps:**

- Run some command on Local CLI to ensure that no confidential information is displayed on it.
- Use SSH or any other protocol to login remotely.
- Check if any sensitive information such as passphrase is visible on Remote CLI.
- Check system log files to see if any sensitive information, such as passphrases or configuration data, is being logged in clear text.
- Use commands like **grep** to search for keywords indicating sensitive data.
- If internal data is revealed in the clear by any local or remote OAM CLI or conf files. Then this test case fails.

d. **Test Observation:**

- **Case 1(Local Login):** No confidential data revealed when trying logging to root user
The Below figure shows that user tries to login as root user, when asked for password, it is not visible on the screen as user types it.

```
priyansha@priyansha:~$ sudo su
[sudo] password for priyansha:
root@priyansha:/home/priyansha#
```

The below figure shows that even if a user enters some wrong password, the correct passphrase is not revealed in any form of error message.

```
priyansha@priyansha:~$ sudo su
[sudo] password for priyansha:
Sorry, try again.
[sudo] password for priyansha:
root@priyansha:/home/priyansha#
```

- **Case 2(Remote Login):** No confidential sensitive data must be revealed when lagged in remotely.

Here the passphrase is not visible.

```
amf@localhost $ ssh pratik@192.168.127.177
pratik@192.168.127.177's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-76-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

90 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

57 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '22.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
*** System restart required ***
Last login: Sun Jul 30 19:51:59 2023 from 192.168.126.157
(base) pratik@pratik-Super-Server:~$
```

- **Case 3(Configuration File):** Tester needs to verify that no sensitive information is visible in any log file.

Example of one log file is taken below, in which no confidential information such as password can be seen in clear text.

```
amf@localhost $ grep -i "password" /var/log/auth.log
Jul 28 16:47:14 priyansha sshd[2331094]: Accepted password for priyansha from 192.168.127.177 port 55274 ssh2
Jul 28 22:47:54 priyansha gdm-password]: gkr-pam: unlocked login keyring
Jul 28 23:50:08 priyansha gdm-password]: gkr-pam: unlocked login keyring
Jul 29 00:07:23 priyansha sshd[3781317]: Accepted password for priyansha from 192.168.126.132 port 47876 ssh2
Jul 29 00:08:35 priyansha sshd[3785691]: Failed password for root from 192.168.126.132 port 47634 ssh2
Jul 29 00:10:36 priyansha sshd[3791537]: Accepted password for priyansha from 192.168.126.132 port 33996 ssh2
Jul 29 00:12:00 priyansha sshd[3795858]: Accepted password for priyansha from 10.10.31.199 port 61274 ssh2
Jul 29 00:46:50 priyansha gdm-password]: gkr-pam: unlocked login keyring
Jul 29 00:48:46 priyansha sshd[3908874]: Accepted password for priyansha from 192.168.126.132 port 33670 ssh2
Jul 29 00:58:52 priyansha gdm-password]: gkr-pam: unlocked login keyring
```

➤ **Test Case Number: 02**

- a. **Test Case Name:** TC2_CONFIDENTIAL_SYSTEM_INTERNAL_DATA
- b. **Test Case Description:** Tests need to be conducted to ensure that access to maintenance mode is restricted to authorized privileged users only.
- c. **Execution Steps:**
 - Attempt to access maintenance mode as an authorized user.
 - Attempt to access maintenance mode as an unauthorized user.

Use command: *systemctl emergency*

d. **Test Observation:**

- **Case 1(Authorized user):** The expected behavior is that it should prompt for the privileged user's password to proceed. If it allows access without a password prompt, it would be a test failure.
- **Case 2(Unauthorized user):** The expected behavior is that it should not allow access and should prompt for the privileged user's password. If it allows unauthorized access, it would be a test failure.

```
testvm@testvm:~$ systemctl emergency
Failed to start emergency.target: Access denied
See system logs and 'systemctl status emergency.target' for details.
testvm@testvm:~$
```

e. **Evidence Provided:**

A testing report which will consist of the following information:

- Screenshot of the output of the terminal of the PC through which DUT is logins.

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_CONFIDENTIAL_SYSTEM_INTERNAL_DATA TC2_CONFIDENTIAL_SYSTEM_INTERNAL_DATA		

2.6.5 TSTP for Evaluation of Protecting data and information in storage

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.6.5
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 6: Data Protection

2. **<Security Requirement No & Name >** 2.6.5. Protecting data and information in storage

3. **<Requirement Description: >**

- a) For sensitive data (persistent or temporary) in storage, read access rights shall be restricted. Sensitive files of SMF that are needed for the functionality shall be Protected against manipulation strictly using the Secure cryptographic controls prescribed in Table 1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” with appropriate non-repudiation controls.
- b) In addition, the following rules apply for:
 - i. SMF that need access to identification and authentication data in the clear/readable form e.g. in order to perform an authentication. Such SMFs shall not store this data in the clear/readable form, but scramble or encrypt it by implementation-specific means.
 - ii. SMF that do not need access to sensitive data in the clear. Such SMFs shall hash this sensitive data strictly using the cryptographic controls prescribed in Table 1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only.
 - iii. Stored files in the SMF: Shall be protected against manipulation strictly using the NCCS approved Secure cryptographic controls prescribed in Table 1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.2.3]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./system.out** (Used in IITH testbed to get SYSTEM version. Check with OEM manufacturer document for command specific to your SYSTEM)

Here we are assuming DUT to be SYSTEM, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. **DUT Configuration:** Verify that the hashing algorithm used is compliant with the latest Document "Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)"

Command used: **sudo cat /etc/pam.d/common-password** (To get configuration info)

```
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
password    [success=1 default=ignore]      pam_unix.so obscure sha512
# here's the fallback if no module succeeds
password    requisite                       pam_deny.so
```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SYSTEM_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

There are various encryption tools available to encrypt file on the system

- OpenSSL
- Gpg
- Ccrypt
- And So on....

a) For ccrypt

command used: **ccrypt -V** (To get version information)

```
ccrypt 1.11. Secure encryption and decryption of files and streams.
Copyright (C) 2000-2018 Peter Selinger.
```

The algorithms used for encryption can be noted from here

b) Similarly check for other tools....

There are various FIM (File integrity monitoring) tools available to maintain the integrity of the file system. Some of which are

- OSSEC
- AIDE
- SolarWinds Security Event Manager
- Netwrix Auditor
- And so on....

a) For AIDE (Advanced Intrusion Detection Environment)

command used: **aide -v** (To get version information)

```
user@user:~$ aide -v
Aide 0.17.4
```

command used: **cat /etc/aide/aide.conf** (To get aide configuration)

```
# Ignore e2fs attributes that cannot be set manually
report_ignore_e2fsattrs=EhI

# Set to yes to print the checksums in the report in hex format
report_base16 = no

# if you want to sacrifice security for speed, remove some of these
# checksums.
Checksums = sha256+sha512+rm160+haval+gost+crc32+tiger+whirlpool

# The checksums of the databases to be printed in the report
# Set to 'E' to disable.
database_attrs = Checksums

# check permissions, owner, group and file type
OwnerMode = p+u+g+ftype
```

Verify that the checksums used by the DUT is in accordance with the latest Document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)”

b) Similarly check for other tools....

6. Preconditions:

- The tester be provided with the location of the files which are to be tested
- Tester should have access to privileged accounts to make changes to sensitive files
- For the files to be stored in encrypted manner, original data format should be available to differentiate between encrypted and unencrypted data

7. Test Objective: To verify if the access to sensitive files are restricted, sensitive data required in cleared text is stored in an encrypted manner and protected from manipulation using integrity checks

8. Test Plan:

8.1. Number of Test Scenarios

8.1.1. Test Case for Access Rights *Securing Networks*

Test Scenario to test if sensitive files are access protected

8.1.2. Test Case for Encrypted Files with ccrypt

Test Scenario to test if authentication data is stored in an encrypted manner

8.1.3. Test Case for File Integrity Monitoring tool AIDE

Test Scenario to test if files are integrity protected

8.1.4. Test Case for One-Way Hash

Test Scenario to verify if sensitive information (like passwords) are stored in a form of one-way hash

8.2. Testbed Diagram



8.3. **Tools Required:** NULL

8.4. **Test Execution Steps:**

- Tester tries to read a sensitive file with access permissions beyond current rights of tester
- Tester tries to read a authentication data with appropriate access rights
- Testers verifies that sensitive data which are not required in clear text are stored in hash
- Tester makes changes in a file and notes that integrity mismatch takes place

9. **Expected Results for Pass:**

- **Case 1:** Tester is not able to read a file if proper access permission is not available
- **Case 2:** Tester verifies that authentication data is stored in an encrypted form
- **Case 3:** Tester verifies that sensitive data which are not required in clear text are stored in hash
- **Case 4:** The File Integrity Monitoring system displays mismatch for hashes

10. **Expected Format of Evidence:** Screenshots of Terminal

11. **Test Execution:**

- **Test Case Number:** 01

a. **Test Case Name:** TC_PSW_STOR_SUPPORT_1

b. **Test Case Description:** DUT should not allow tester to read the content of the file

c. **Execution Steps:**

- Tester logs into the DUT with non-root access
- Tester tries to read the contents of the file using following command
 - `cat <file_name>`

```
cat: ue_context: Permission denied
```

d. **Test Observation:**

(Case 1) Tester is not able to read a file if proper access permission is not available

e. **Evidence Provided**

Screenshot of Terminal

➤ **Test Case Number: 02**

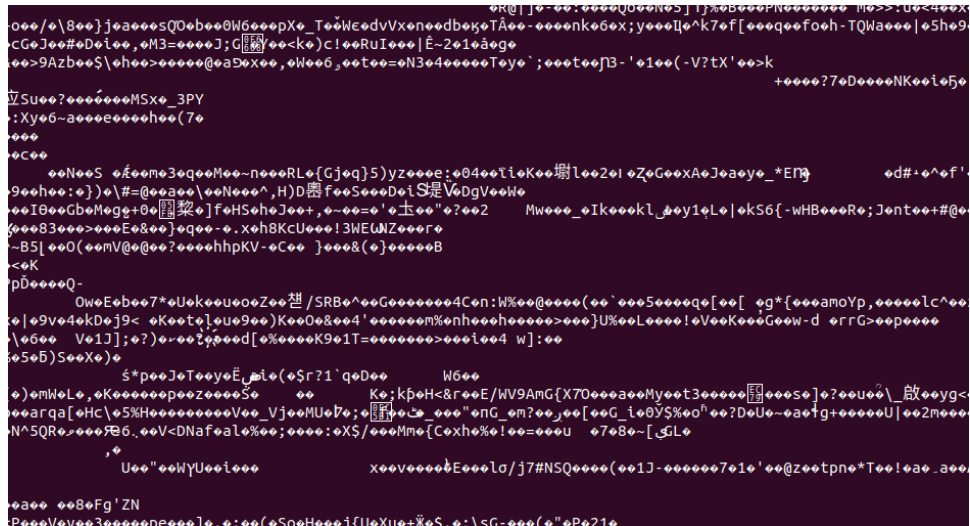
a. **Test Case Name:** TC_PSW_STOR_SUPPORT_2

b. **Test Case Description:** DUT should not display the sensitive information in clear text

c. **Execution Steps:**

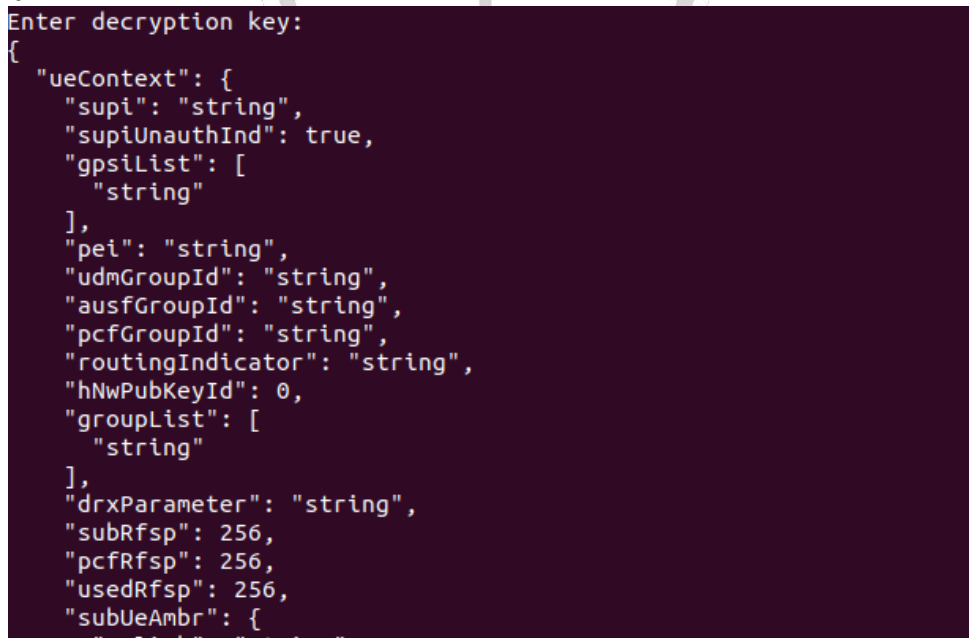
- Tester tries to read the file using following command

• `cat <file_name>`



- Tester tries to read the file using following command with appropriate passphrase

• `ccat <file_name>`



d. **Test Observation:**

(Case 2) Tester verifies that authentication data is displayed in a scrambled manner and can only be viewed when decrypted

e. **Evidence Provided:-** Screenshot of Terminal

➤ **Test Case Number:** 03

a. **Test Case Name:** TC_PSW_STOR_SUPPORT_3

b. **Test Case Description:** DUT should store the data which is not required in clear text in hash format

c. **Execution Steps:**

- Tester shall create a new user using following command appropriate password

- *sudo adduser <user_name>*

```
Adding user `iith' ...
Adding new group `iith' (1003) ...
Adding new user `iith' (1002) with group `iith' ...
The home directory `/home/iith' already exists. Not copying from `/etc/skel'.
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for iith
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n]
```

- Tester shall then observe the stored password using following command

- *sudo cat /etc/shadow*

```
nx*:18985:0:99999:7:::
ftp*:19039:0:99999:7:::
debian-tor*:19460:0:99999:7:::
sshd*:19487:0:99999:7:::
iith:$6$VhATtaf5lilIz8$ktz/6Y0dIdiSp6kP2HPvNKw/IJfpuUQLGPXlxRgnFNT7ydTR.HqI
```

d. **Test Observations:**

(Case 3): Tester verifies that password is stored in hash format (The \$6\$ at the beginning denotes the use of SHA512 which is compliant with the standards)

e. **Evidence Provided:-** Screenshot of Terminal

Securing Networks

➤ **Test Case Number:** 04

a. **Test Case Name:** TC_PSW_STOR_SUPPORT_4

b. **Test Case Description:** DUT should have a mechanism to protect the sensitive files from being manipulated

c. **Execution Steps:**

- Tester shall manipulate the contents of a file
- Tester then shall run the following command to display if any changes are made to any if the files

- *sudo aide -c /etc/aide/aide.conf --limit <directory>--check*


```

Summary:
Total number of entries:      645822
Added entries:                0
Removed entries:              0
Changed entries:              1

-----
Changed entries:
-----

f =.... mci.H.. : /home/user/amf_context/ue_context

-----
Detailed information about changes:
-----

File: /home/user/amf_context/ue_context
Mtime   : 2023-07-05 01:48:01 +0530 | 2023-07-05 03:19:52 +0530
Ctime   : 2023-07-05 01:48:01 +0530 | 2023-07-05 03:19:52 +0530
Inode    : 663059                    | 663060
SHA256   : 3fsHGuQgWbd/wws+J8y4C38nvBFzjKWI | CqblvN+fCYNWKTpPOAvUbaVC7FBSLk0w
          xyliez6UwiQw=               | Tq2c6JXZoXw=

```

d. **Test Observations:** - (Case 4): The File Integrity Monitoring system displays mismatch for hashes

e. **Evidence Provided:** - Screenshot of Terminal

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	REMARKS
1	TC_PSW_STOR_SUPPORT_1		
2	TC_PSW_STOR_SUPPORT_2		
3	TC_PSW_STOR_SUPPORT_3		
4	TC_PSW_STOR_SUPPORT_4		

2.6.6 TSTP Report for Evaluation of Protection against Copy of Data

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.6.6
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **ITSAR Section No & Name:** Section 6: Data Protection

2. **Security Requirement No & Name:** 2.6.6 Protection against Copy of Data

3. **Requirement Description:**

- a) Without authentication & authorization and except for specified purposes, SMF shall not support copying of control plane and user plane data.
- b) Protective measures should exist against use of available system functions / software residing in SMF to create copy of control plane and user plane data for illegal transmission.

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: ifconfig -a (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: ./SMF.out (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: cat /etc/os-release (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

UNIX/ all major UNIX-like derivatives, including Linux is used on the Network product.

- To get the hash of configuration file if the file is a ASCII text file.
- Command - sha256sum DUT_config.conf
- Digest Hash of Tested Configuration:
- DUT_config.conf:-
19be54b42975f7f414c53a280373aa9466398781ecd77a2e9d090dbc6dbfdd8
- To get the hash of OS if using docker

- Command - **docker images --digests**

- **Digest Hash of OS:**

- DUT_IMAGE:

fd363f0e0d146c869d60649bc4f42e7008829d9d67a5dfdaf3b38ad24af7d53a

6. **Preconditions:**

- a. OEM shall provide the list of all the files that contain sensitive data.
- b. Only root will have the permission to read or copy the contents of any file.
- c. There can be a group of privileged users with the permission to read the file. OEM shall provide a list of those users.
- d. Data that can be copied can be data in usage or data in transit or data in storage. In order to protect copy of data in storage, clause 2.6.5. Protecting data and information in storage should be satisfied. To protect copy of data in transit, clause 2.1.2 Management Traffic Protection should be satisfied. This TSTP covers steps for protection against copy of data in usage.

7. **Test Objective:** Use of available system functions / software residing in SMF to create copy of control plane and user plane data for illegal transmission.

8. **Test Plan:**

8.1 **Number of test scenarios/test cases: 1**

8.2 **Tools used:** Wireshark, OpenSSH, Valgrind

8.3 **Test Execution:**

The accredited evaluator's test lab is required to execute the following steps:

- Tester tries to create copy of files in SMF.
- System should not allow copy of any files if user is not authenticated and authorized.

9. **Expected Results for Pass:** A list showing that all the important files containing control and data plane data in SMF have the read/write/execute rights set such that any function or software can't make a copy of the data until it is authorized to do so. (Users in sudo list are only able to make a copy of data).

10. **Expected Format of Evidence:** A test report provided by the accredited evaluator's test lab which will consist of the following information:
- a. List of privileged users, who are allowed to read the files.
 - b. Screenshot/Report of the showing other users are not allowed to read/copy.
 - c. Report showing that all the below test cases have passed depending on the implementation followed.

11. **Test Execution:**

➤ **Test Case Number: 1**

a. **Test Case Name:** TC_PROTECTION_AGAINST_COPY_OF_DATA

b. **Test Case Description:**

The runtime user plane and control plane data can be stored using various mechanisms. The specific mechanisms depend on the implementation details and architectural choices made by the system. Some of the common ways to store data in usage can be:

- Data structures and variables
- Cache
- Files

Objective is to test protection of copy of data stored in above mentioned mechanisms.

c. **Execution Steps:**

1) Files

- **Access Controls:** Check that access controls is implemented at the application or system level to regulate file access. This includes authentication and authorization mechanisms to ensure that only authorized users or processes can access and modify the files.
- **Encryption:** Check that encryption techniques is applied to sensitive files to protect their contents.

2) Cache

- **Encryption:** Check that encryption techniques is used on data stored in the cache.
- **Time-based Expiration:** Check that cache is configured to have time-based expiration for the stored data.
- **Access Controls:** Check that access controls are implemented to restrict unauthorized access to the cache. Determine the directory where the cache is stored. This will depend on the specific caching system or application being used.

3) Data structures and variables

- **Memory Protection Mechanisms:** Verify that proper coding practices are used such that there are no memory leaks, buffer overflows, memory corruption, or other vulnerabilities, as it can lead to unauthorized data copying. Examples include using safe memory functions, bounds checking, and secure memory allocation techniques.
- **Memory Access Permissions:** Check that memory assigned to variables are protected.

d. **Test Observations:**

➤ **Case 1 (Files):**

A. Access Controls testing

I.In order to stop copy of data in linux, read permissions of the files should not be given.

Tester tries to create the copy of all the files in SMF. He should not be able to do so if not authorized.

```
[base] password for supriya:
supriya@supriya-Super-Server:~/phase3/5G/src$ cp amf.txt /home/supriya/
cp: cannot open 'amf.txt' for reading: Permission denied
supriya@supriya-Super-Server:~/phase3/5G/src$
```

II. Tester tries to create the copy by logging as one of the root users and he should be able to create a copy.

```
(base) supriya@supriya-Super-Server:~/phase3/5G$ sudo cat amf.txt
[sudo] password for supriya:
[ 0%] Built target AMF_SCTP
[ 0%] Built target 3GPP_COMMON_TYPES
[ 0%] Built target CONTEXTS
[ 1%] Built target AMF_SECU_NAS
[ 1%] Built target AMF_SECU_5GAKA
[ 2%] Built target AMF_UTILS
[ 20%] Built target AMF_SBI_SERVER
Scanning dependencies of target AMF
[ 20%] Building CXX object amf-app/CMakeFiles/AMF.dir/amf_sbi.cpp.o
[ 20%] Linking CXX static library libAMF.a
[ 99%] Built target AMF
[100%] Linking CXX executable amf
[100%] Built target amf
```

B. **Encryption testing:** To check whether files are encrypted or not in Linux, following methods can be used:

- **File Inspection:** Inspect the file properties and characteristics to determine if encryption is applied. Here are a few indicators to look for:
- **File Extensions:** Encrypted files often have specific file extensions associated with encryption algorithms or encrypted file formats. For example, ".gpg" for GnuPG encrypted files or ".enc" for generic encrypted files.
- **File Headers:** Some encrypted files have unique headers or magic numbers at the beginning of the file that indicate encryption. The `file` command can be used to check the file type and examine the file headers.
- **Metadata or Tags:** The file metadata or tags can be checked, if available, to see if there are any indications of encryption. This can be done using file explorers or metadata analysis tools.

➤ **Case 2 (Cache):**

Access control and encryption testing will be done as explained in case 1.

A. Time-based Expiration: The cache expiration time is typically managed by the caching software or service being used. The expiration time for cached data can be configured based on the specific caching solution being used.

Some popular examples are as follows:

- **Memcached:** If memcached is being used, check whether expiration time is set or not in the conf file.

The configuration file is usually located at `/etc/memcached.conf`. In the configuration file, find the line that specifies the `-t` option or `expire_time`. This option defines the default expiration time for items stored in the cache. Check that this value is not set to some large value. The expiration time can be set when storing data by specifying a time value in seconds.

```
memcached_set($memcached, 'key', 'value', 3600);
```

- **Redis Cache:** If this is being used, check whether expiration time is set or not using `EXPIRE` command.

For example, to set the expiration time for a key called "mykey" to 1 hour.

```
EXPIRE mykey 3600
```

➤ Case 3 (Data structures and variables):

A. Memory Access Permissions:

In Linux, a particular section of RAM assigned to a program can be protected by utilizing memory access permissions and virtual memory mechanisms. Linux provides memory access permissions through the `mprotect` system call, which allows to change the access permissions of memory pages. Specific permissions for a particular section of memory can be set, such as read-only or no access. Check that if memory is assigned to variables, such methods are used.

Ex for C/C++:

```
#include <sys/mman.h>

// Allocate memory for a section
void* section = mmap(NULL, section_size, PROT_READ | PROT_WRITE, MAP_PRIVATE | MAP_ANONYMOUS, -1, 0);

// Set memory access permissions
if (mprotect(section, section_size, PROT_READ) == -1) {
    // Error handling
}
```

- #### B. Memory Protection testing:
- Valgrind can be used to detect memory leaks, uninitialized variables, invalid memory accesses, and other memory-related errors in C, C++, and other supported languages.

Any other tool with similar capabilities can also be used.

Command: `valgrind [valgrind-options] program [program-arguments]`

```

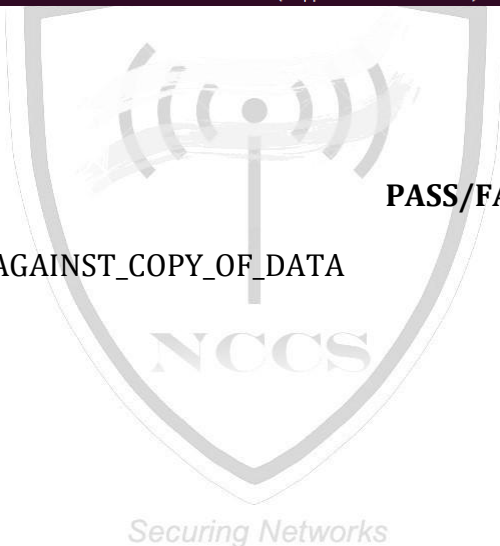
(base) supriya@supriya-Super-Server:~$ valgrind --leak-check=full ./a.out
==386824== Memcheck, a memory error detector
==386824== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==386824== Using Valgrind-3.18.1 and LibVEX; rerun with -h for copyright info
==386824== Command: ./a.out
==386824==

==386824== Invalid write of size 4
==386824==    at 0x10916B: f (in /home/supriya/a.out)
==386824==    by 0x109180: main (in /home/supriya/a.out)
==386824== Address 0x4a9a068 is 0 bytes after a block of size 40 alloc'd
==386824==    at 0x4848899: malloc (in /usr/libexec/valgrind/vgpreload_memcheck-amd64-linux.so)
==386824==    by 0x10915E: f (in /home/supriya/a.out)
==386824==    by 0x109180: main (in /home/supriya/a.out)
==386824==
==386824== HEAP SUMMARY:
==386824==    in use at exit: 40 bytes in 1 blocks
==386824==    total heap usage: 1 allocs, 0 frees, 40 bytes allocated
==386824==
==386824== 40 bytes in 1 blocks are definitely lost in loss record 1 of 1
==386824==    at 0x4848899: malloc (in /usr/libexec/valgrind/vgpreload_memcheck-amd64-linux.so)
==386824==    by 0x10915E: f (in /home/supriya/a.out)
==386824==    by 0x109180: main (in /home/supriya/a.out)
==386824==
==386824== LEAK SUMMARY:
==386824==    definitely lost: 40 bytes in 1 blocks
==386824==    indirectly lost: 0 bytes in 0 blocks
==386824==    possibly lost: 0 bytes in 0 blocks
==386824==    still reachable: 0 bytes in 0 blocks
==386824==    suppressed: 0 bytes in 0 blocks
==386824==
==386824== For lists of detected and suppressed errors, rerun with: -s
==386824== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 0 from 0)

```

12. Test Case Result:

SL.No	TEST CASE NAME	PASS/FAIL (Please see Note Below)
1	TC_PROTECTION_AGAINST_COPY_OF_DATA	



2.6.7 TSTP for Protection against Data Exfiltration - Overt Channel

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.6.7
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 6: Data Protection
2. **<Security Requirement No & Name >** 2.6.8 Protection against Data Exfiltration - Covert Channel
3. **<Requirement Description: >**
 - a) SMF shall have mechanisms to prevent data exfiltration attacks for theft of control plane and user plane data in use and data in transit.
 - b) Establishment of outbound overt channels such as, HTTPS, IM, P2P, Email etc. are to be forbidden if they are auto-initiated by / auto-originated from the SMF.
 - c) Session logs shall be generated for establishment of any session initiated by either user or SMF.

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions**

- The tester has administrative privileges
- A tester machine is available.
- Test environment with a Terminal.

7. **Test Objective:**

- To check that SMF has mechanisms to prevent data exfiltration attacks.
- To Check Establishment of Outbound Channels are forbidden if they are auto-initiated or auto-originated
- To Check that Logs maintained at SMF side for different sessions

8. **Test Plan:**

8.1 **Test Setup Diagram:**



Note: Tester Can Access DUT directly using physical console as well

8.2 **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.3 **Test Execution Steps**

- The Tester verifies that Intrusion and Extrusion Detection Mechanisms are in place as per OEM/Vendor documentation. The mechanisms must include (but not limited to):
 - Details of Intrusion detection systems in place
 - Log review details (Tester should verify Log Review timetable as recommended by Vendor)
 - Location and Usage of Network performance data to prevent DDoS attacks
 - Location of Session Logs for Routine Traffic threat assessment.
 - Mechanisms should be in place such that following information is not revealed for non-root users:
 - Metadata
 - Runtime of Cryptographic schemes
 - Memory usage of Cryptographic Schemes
 - Bitrate of protocol used
 - Buffer information

- The Tester should ensure any outbound connection should have a mechanism to require re-authentication by user and no outbound connection or inbound connection should have root privileges to system.
- Session Logs should be maintained for all connections and services

9. **Expected Results for Pass:** Evidence that all mechanisms are in place and working.

10. **Expected Format of Evidence:** Log files and screen shots of test executions.

11. **Test Execution:**

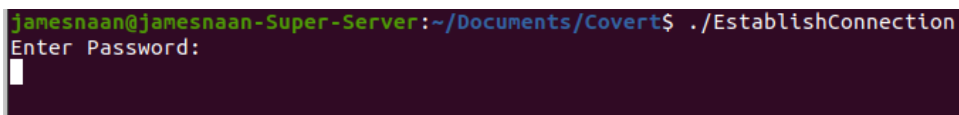
➤ **Test Case Number:** 01

a. **Test Case Name:** TC_PROTECTION_AGAINST_DATA_EXFILTRATION_OVERT_CHANNEL

b. **Test Case Description:** To verify that all mechanisms are in place.

c. **Execution Steps:**

- The Tester verifies that Intrusion and Extrusion Detection Mechanisms are in place as per OEM/Vendor documentation. The mechanisms must include (but not limited to):
 - Details of Intrusion detection systems in place
 - Log review details (Tester should verify Log Review timetable as recommended by Vendor)
 - Location and Usage of Network performance data to prevent DDoS attacks
 - Location of Session Logs for Routine Traffic threat assessment.
 - Mechanisms should be in place such that following information is not revealed for non-root users:
 - Metadata
 - Runtime of Cryptographic schemes
 - Memory usage of Cryptographic Schemes
 - Bitrate of protocol used
 - Buffer information
- The Tester should ensure any outbound connection should have a mechanism to require re-authentication by user and no outbound connection or inbound connection should have root privileges to system. Mechanism may include Password Prompts for Re-authentication.



```
jamesnaan@jamesnaan-Super-Server:~/Documents/Covert$ ./EstablishConnection
Enter Password:
█
```

- This information should be present in the logs to ensure that auto-initiation/auto origination of outbound connections is not possible with user authorization.
- Session Logs should be maintained for all connections and services and all re-authentication processes.

- (Alternate Solutions for Clause B – There are two different logs maintained. One session log for all connections manually established and one system log checking any connection maintained and other system properties. We start with 4 manual connections and one auto-initiated connection that does not show up in session logs. So, in the session logs there are only 4 connections but in the system logs 5 are shown. If 5 are shown test fails but if only 4 are shown it means auto-initiation failed)
- Tester should check that Outbound connection re-authentication has been logged by the system.

```

g3log: created log file at: wed Feb 09 04:55:00 2022
<2022-02-09 04:55:00:853 INF>[COMMON]In5gtSctpClient.cpp:76|In5gtConnectWithServer|Successfully connected to AMF on socket:7
<2022-02-09 04:55:00:854 INF>[RAN]In5gtRanNew.cpp:657|In5gtSndNGSetupReq|*****STARTING OF NG SETUP REQUEST*****
<2022-02-09 04:55:00:854 DBG>[NGAP]In5gtNgapCommon.cpp:31|In5gtSetPLMNIdentity|plmnBuf[0] ngap value is:4
<2022-02-09 04:55:00:854 DBG>[NGAP]In5gtNgapCommon.cpp:32|In5gtSetPLMNIdentity|plmnBuf[1] ngap value is:f4
<2022-02-09 04:55:00:854 DBG>[NGAP]In5gtNgapCommon.cpp:33|In5gtSetPLMNIdentity|plmnBuf[2] ngap value is:0
<2022-02-09 04:55:00:854 INF>[RAN]In5gtRanNew.cpp:673|In5gtSndNGSetupReq|Encoded Successfully
<2022-02-09 04:55:00:854 DBG>[NGAP]In5gtNGSetupRequest.cpp:182|In5gtPrintXERNGSetupReq|Value in XER Format:
<2022-02-09 04:55:00:855 DBG>[COMMON]In5gtSctpClient.cpp:151|In5gtSnd|Sending SCTP packet from stream length:0 thread id:1399025864
<2022-02-09 04:55:00:855 DBG>[COMMON]In5gtSctpClient.cpp:159|In5gtSnd|Successfully sent 66 bytes data to server
<2022-02-09 04:55:00:855 DBG>[NGAP]In5gtNGSetupRequest.cpp:283|In5gtFreeNGSetupRequestEnc|Freeing PLMN Identity octets
<2022-02-09 04:55:00:857 DBG>[COMMON]In5gtSctpClient.cpp:259|In5gtRcv|Received SCTP Packet of len :75
<2022-02-09 04:55:00:857 DBG>[COMMON]In5gtSctpClient.cpp:260|In5gtRcv|Received SCTP from thread :139902586498944
<2022-02-09 04:55:00:857 DBG>[RAN]In5gtRanNew.cpp:699|In5gtRcvNGSetupResponse|*****buffer len:4d*****75
<2022-02-09 04:55:00:857 DBG>[RAN]In5gtRanNew.cpp:781|In5gtRcvNGSetupResponse|Consumed: 75
<2022-02-09 04:55:00:857 DBG>[RAN]In5gtRanNew.cpp:787|In5gtRcvNGSetupResponse|Consumed:75
<2022-02-09 04:55:00:857 INF>[RAN]In5gtRanNew.cpp:788|In5gtRcvNGSetupResponse|Decoded Successfully:
<2022-02-09 04:55:00:857 DBG>[NGAP]In5gtNGSetupResponse.cpp:88|In5gtPrintXERNGSetupResponse|Value in XER Format:
<2022-02-09 04:55:00:862 DBG>[COMMON]In5gtTun.cpp:38|In5gtAttach|Device name tun1
<2022-02-09 04:55:00:862 INF>[RAN]In5gtRanEmulatorNew.cpp:418|In5gtRun|Traffic monitor server started
<2022-02-09 04:55:00:863 DBG>[RAN]In5gtRanEmulatorNew.cpp:419|In5gtRun|Traffic monitor server started for ip:172.17.0.8 Port:2152
<2022-02-09 04:55:00:866 DBG>[RAN]In5gtRanNew.cpp:85|In5gtInit|*****SUPI Modified:*****48480300000001:MSIN900000001

```

d. Test Observations:

All mechanisms should be up and running.

12. Test Case Result:

S. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_PROTECTION_AGAINST_DATA_EXFILTRATION_OVERT_CHANNEL		

2.6.8 TSTP for Protection against Data Exfiltration - Covert Channel

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.6.8
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 6: Data Protection
2. **<Security Requirement No & Name >** 2.6.7 Protection against Data Exfiltration - Overt Channel
3. **<Requirement Description: >**
 - a) SMF shall have mechanisms to prevent data exfiltration attacks for theft of data in use and data in transit.
 - b) Establishment of outbound covert channels and tunnels such as DNS Tunnel, HTTPS Tunnel, ICMP Tunnel, TLS, SSL, SSH, IPSEC VPN, RTP Encapsulation etc. are to be forbidden if they are auto-initiated by / auto-originated from the SMF.
 - c) Session logs shall be generated for establishment of any session initiated by either user or SMF system.

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf

```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions**

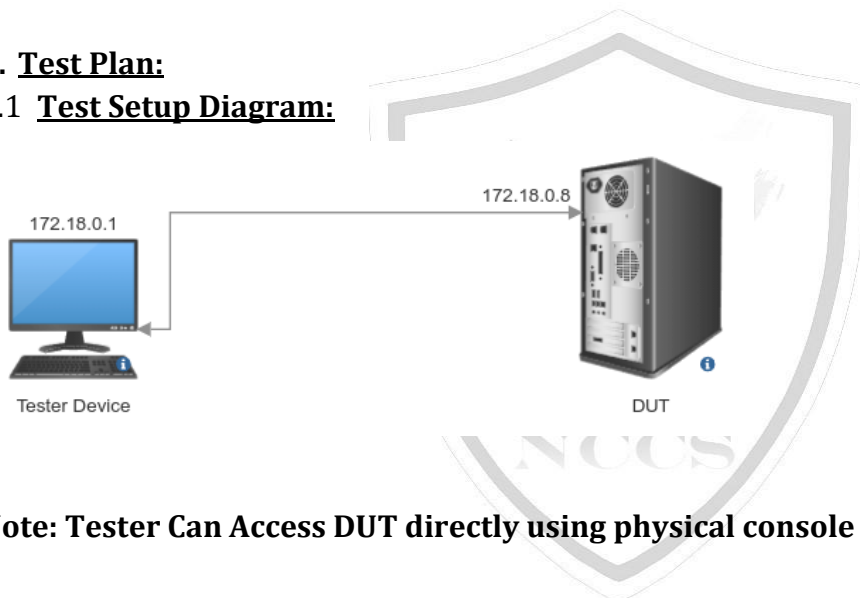
- The tester has administrative privileges
- A tester machine is available.
- Test environment with a Terminal.

7. **Test Objective:**

- To check that SMF has mechanisms to prevent data exfiltration attacks.
- To Check Establishment of Outbound Channels are forbidden if they are auto-initiated or auto-originated
- To Check that Logs maintained at SMF side for different sessions

8. **Test Plan:**

8.1 **Test Setup Diagram:**



Note: Tester Can Access DUT directly using physical console as well

8.2 Tools Used: Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.3 **Test Execution Steps**

- The Tester verifies that Intrusion and Extrusion Detection Mechanisms are in place as per OEM/Vendor documentation. The mechanisms must include (but not limited to):
 - Details of Intrusion detection systems in place
 - Log review details (Tester should verify Log Review timetable as recommended by Vendor)
 - Location and Usage of Network performance data to prevent DDoS attacks
 - Location of Session Logs for Routine Traffic threat assessment.
 - Mechanisms should be in place such that following information is not revealed for non-root users:
 - Metadata

- Runtime of Cryptographic schemes
 - Memory usage of Cryptographic Schemes
 - Bitrate of protocol used
 - Buffer information
- The Tester should ensure any outbound connection should have a mechanism to require re-authentication by user and no outbound connection or inbound connection should have root privileges to system.
- Session Logs should be maintained for all connections and services

9. **Expected Results for Pass:-** Evidence that all mechanisms are in place and working.

10. **Expected Format of Evidence:-** Log files and screen shots of test executions.

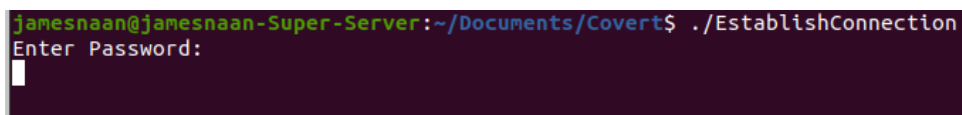
11. **Test Execution:**

➤ **Test Case Number:** 01

- a. **Test Case Name:** TC_PROTECTION_AGAINST_DATA_EXFILTRATION_COVERT_CHANNEL
- b. **Test Case Description:** To verify that all mechanisms are in place.

c. **Execution Steps:**

- The Tester verifies that Intrusion and Extrusion Detection Mechanisms are in place as per OEM/Vendor documentation. The mechanisms must include (but not limited to):
- Details of Intrusion detection systems in place
 - Log review details (Tester should verify Log Review timetable as recommended by Vendor)
 - Location and Usage of Network performance data to prevent DDoS attacks
 - Location of Session Logs for Routine Traffic threat assessment.
 - Mechanisms should be in place such that following information is not revealed for non-root users:
 - Metadata
 - Runtime of Cryptographic schemes
 - Memory usage of Cryptographic Schemes
 - Bitrate of protocol used
 - Buffer information
- The Tester should ensure any outbound connection should have a mechanism to require re-authentication by user and no outbound connection or inbound connection should have root privileges to system. Mechanism may include Password Prompts for Re-authentication.



```
jamesnaan@jamesnaan-Super-Server:~/Documents/Covert$ ./EstablishConnection
Enter Password:
█
```


- This information should be present in the logs to ensure that auto-initiation/auto origination of outbound connections is not possible with user authorization.
- Session Logs should be maintained for all connections and services and all re-authentication processes.
- (Alternate Solutions for Clause B – There are two different logs maintained. One session log for all connections manually established and one system log checking any connection maintained and other system properties. We start with 4 manual connections and one auto-initiated connection that does not show up in session logs. So, in the session logs there are only 4 connections but in the system logs 5 are shown. If 5 are shown test fails but if only 4 are shown it means auto-initiation failed)
- Tester should check that Outbound connection re-authentication has been logged by the system.

```

2 g3log: created log file at: Wed Feb 09 04:55:00 2022
3 <2022-02-09 04:55:00:853 INF>|COMMON|In5gtSctpClient.cpp:76|In5gtConnectWithServer|Successfully connected to AMF on socket:7
4
5 <2022-02-09 04:55:00:854 INF>|RAN|In5gtRanNew.cpp:657|In5gtSndNGSetupReq|*****STARTING OF NG SETUP REQUEST*****
6
7 <2022-02-09 04:55:00:854 DBG>|NGAP|In5gtNgapCommon.cpp:31|In5gtSetPLMNIdentity|plmnBuf[0] ngap value is:4
8
9 <2022-02-09 04:55:00:854 DBG>|NGAP|In5gtNgapCommon.cpp:32|In5gtSetPLMNIdentity|plmnBuf[1] ngap value is:f4
10 |
11 <2022-02-09 04:55:00:854 DBG>|NGAP|In5gtNgapCommon.cpp:33|In5gtSetPLMNIdentity|plmnBuf[2] ngap value is:0
12
13 <2022-02-09 04:55:00:854 INF>|RAN|In5gtRanNew.cpp:673|In5gtSndNGSetupReq|Encoded Successfully
14
15 <2022-02-09 04:55:00:854 DBG>|NGAP|In5gtNGSetupRequest.cpp:182|In5gtPrintXERNGSetupReq|Value in XER Format:
16
17
18 <2022-02-09 04:55:00:855 DBG>|COMMON|In5gtSctpClient.cpp:151|In5gtSnd|Sending SCTP packet from stream length:0 thread id:13990258649
19
20 <2022-02-09 04:55:00:855 DBG>|COMMON|In5gtSctpClient.cpp:159|In5gtSnd|Successfully sent 66 bytes data to server
21
22 <2022-02-09 04:55:00:855 DBG>|NGAP|In5gtNGSetupRequest.cpp:283|In5gtFreeNGSetupRequestEnc|Freeing PLMN Identity octets
23
24
25 <2022-02-09 04:55:00:857 DBG>|COMMON|In5gtSctpClient.cpp:259|In5gtRcv|Received SCTP Packet of len :75
26
27 <2022-02-09 04:55:00:857 DBG>|COMMON|In5gtSctpClient.cpp:260|In5gtRcv|Received SCTP from thread :139902586490944
28
29 <2022-02-09 04:55:00:857 DBG>|RAN|In5gtRanNew.cpp:699|In5gtRcvNGSetupResponse|*****buffer len:%d*****75
30
31 <2022-02-09 04:55:00:857 DBG>|RAN|In5gtRanNew.cpp:701|In5gtRcvNGSetupResponse|Consumed: 75
32
33 <2022-02-09 04:55:00:857 DBG>|RAN|In5gtRanNew.cpp:707|In5gtRcvNGSetupResponse|Consumed:75
34
35 <2022-02-09 04:55:00:857 INF>|RAN|In5gtRanNew.cpp:708|In5gtRcvNGSetupResponse|Decoded Successfully:
36
37 <2022-02-09 04:55:00:857 DBG>|NGAP|In5gtNGSetupResponse.cpp:80|In5gtPrintXERNGSetupResponse|Value in XER Format:
38
39
40 <2022-02-09 04:55:00:862 DBG>|COMMON|In5gtTun.cpp:38|In5gtAttach|Device name tun1
41
42 <2022-02-09 04:55:00:862 INF>|RAN|In5gtRanUeEmulatorNew.cpp:418|In5gtRun|Traffic monitor server started
43
44 <2022-02-09 04:55:00:863 DBG>|RAN|In5gtRanUeEmulatorNew.cpp:419|In5gtRun|Traffic monitor server started for ip:172.17.0.8 Port:2152
45
46 <2022-02-09 04:55:00:866 DBG>|RAN|In5gtRanNew.cpp:85|In5gtInit|*****SUPI Modified:*****40400900000001:MSIN9000000001
47

```

d. **Test Observations:** All mechanisms should be up and running.

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_PROTECTION_AGAINST_DATA_EXFI LTRATION_COVERT_CHANNEL		

2.7.1 TSTP for Traffic Filtering – Network Level Requirement

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.7.1
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 6: Data Protection
2. **<Security Requirement No & Name >** 2.7.1 Traffic Filtering – Network Level Requirement
3. **<Requirement Description: >** SMF shall provide a mechanism to filter incoming IP packets on any IP interface.

In particular the SMF shall provide a mechanism:

- a) To filter incoming IP packets on any IP interface at Network Layer and Transport Layer of the stack ISO/OSI.
- b) To allow specified actions to be taken when a filter rule matches. In particular at least the following actions should be supported:
 - Discard/Drop: the matching message is discarded; no subsequent rules are applied and no answer is sent back.
 - Accept: the matching message is accepted.
 - Account: the matching message is accounted for i.e. a counter for the rule is incremented. This action can be combined with the previous ones. This feature is useful to monitor traffic before its blocking.
- c) To enable/disable for each rule the logging for Dropped packets, i.e. details on messages matching the rule for troubleshooting.
- d) To filter on the basis of the value(s) of source IP, destination IP and port addresses of protocol header.
- e) To reset the accounting.
- f) The SMF shall provide a mechanism to disable/enable each defined rule.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.6.2.1]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal
```

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

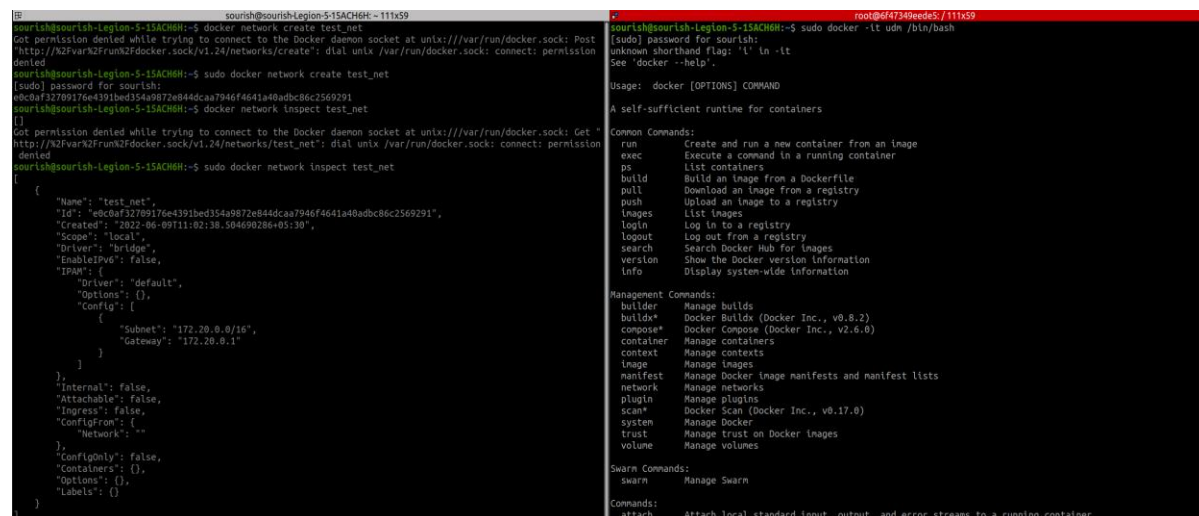
Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```



```
source@source-5-15ACMH:~$ docker network create test_net
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post
"http://172.20.0.1:2376/v1.24/networks/create": dial unix /var/run/docker.sock: connect: permission
denied
source@source-5-15ACMH:~$ sudo docker network create test_net
[sudo] password for source:
e8c0af32709176e4391bed354a9872e844caa7946f4641a40adbc86c2569291
source@source-5-15ACMH:~$ docker network inspect test_net
[
  {
    "Name": "test_net",
    "Id": "e8c0af32709176e4391bed354a9872e844caa7946f4641a40adbc86c2569291",
    "Created": "2022-06-09T11:02:38.504690286+05:30",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.20.0.0/16",
          "Gateway": "172.20.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {},
    "Options": {},
    "Labels": {}
  }
]
```

```
source@source-5-15ACMH:~$ sudo docker --help
[sudo] password for source:
unknown shorthand flag: 'l' in -lt
See 'docker --help'.

Usage: docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
run          Create and run a new container from an image
exec        Execute a command in a running container
ps          List containers
build       Build an image from a Dockerfile
pull        Download an image from a registry
push        Upload an image to a registry
images      List images
login       Log in to a registry
logout      Log out from a registry
search      Search Docker Hub for images
version     Show the Docker version information
info        Display system-wide information

Management Commands:
builder     Manage builds
buildx      Docker Buildx (Docker Inc., v0.8.2)
compose     Docker Compose (Docker Inc., v2.6.0)
container   Manage containers
context     Manage contexts
image       Manage images
manifest    Manage Docker image manifests and manifest lists
network     Manage networks
plugin      Manage plugins
scan        Docker Scan (Docker Inc., v0.17.0)
system      Manage Docker
trust       Manage trust on Docker images
volume     Manage volumes

Swarm Commands:
swarm       Manage Swarm

Commands:
attach      Attach local standard input, output, and error streams to a running container
```

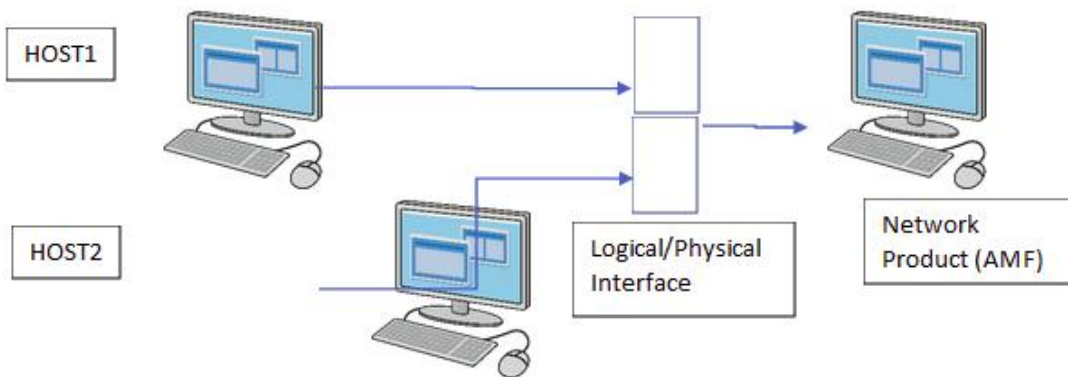
Figure : 5G Testbed Used to Emulate a 1st Host (SMF) and 1 Network Product (SMF)



- Account: the matching message is accounted for i.e. a counter for the rule is incremented. This action can be combined with the previous ones. This feature is useful to monitor traffic before its blocking.
- To enable/disable for each rule the logging for Dropped packets, i.e. details on messages matching the rule for troubleshooting.
- To filter on the basis of the value(s) of source IP, destination IP and port addresses of protocol header.
- To reset the accounting.
- The DUT shall provide a mechanism to disable/enable each defined rule.

8. **Test Plan:**

8.1 **Test Setup Diagram:**



8.2 **Tools Used:** Command line interface of HOST1 and DUT, Ip tables

8.3 **Test Execution Steps**

- OEM should submit document what are the filtering features available
- Firstly, the network should be established as such in which DUT is the Server and 2 Hosts (For simulation purpose we have taken UDM and SMF as the 2 Hosts/Clients) are connected to the Server.
- Obtain the IP Address of the DUT and the Hosts.
- Configure the Rules for Packet Filtering as follows.

Network Product Name and Function	Ip Address of Network Product	Rules
UDM(HOST1)	172.20.0.4	ACCEPT ICMP echo messages
SMF(HOST2)	172.20.0.3	DROP ICMP echo messages
SMF(SERVER/DUT)	172.20.0.2	-

- In case iptables is implemented then below command should be used in command line of DUT for accepting the icmp echo messages from HOST 1.
- **iptables -A INPUT -p icmp --icmp-type 8 -s <HOST1 ip Address> -j ACCEPT**
- In case iptable is implemented then below command should be used in command line of DUT for dropping the icmp echo messages from HOST 2.
- **iptables -A INPUT -p icmp --icmp-type 8 -s <HOST2 ip Address> -j DROP**
- The tester configures the Network Product to only allow ICMP traffic from host 1. *(Check if option is available at network and transport level)*
- The tester initiates ping traffic from host 1.
- The tester initiates ping traffic from host 2.
- We intend to execute the Test Case by dividing the single requirement in multiple smaller test cases.

SL. No	TEST CASE NAME	COVERING SUBCLAUSE
1	TC1_PACKET_FILTERING_HOST1	a,b,d
2	TC2_PACKET_FILTERING_HOST2	a,b,d
3	TC3_ENABLE/DISABLE_FILTERING	c,f
4	TC4_FILTERING_COUNTER_RESET	e

9. **Expected Results for Pass:** Host 1 will receive a 'ping' answer back, but host 2 will not.

10. **Expected Format of Evidence:** Log files and screen shots of test executions.

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC1_PACKET_FILTERING_HOST1

b. **Test Case Description:**

- As per the Rules mentioned in the table, DUT should respond to ICMP messages sent from HOST 1.

c. **Execution Steps:**

- Check that precondition is already satisfied by the DUT.
- Use the command line of the HOST1 to ping the DUT.
- Use the following command
ping -I <HOST1 IP Address> <DUT IP Address>
- Capture the response on the HOST1 for the above command.

d. **Test Observations:**

➤ **Case 1:** DUT's response is positive for the HOST1's icmp messages (PASS CASE)

Below image can be used as a reference that HOST1 is getting positive response from DUT to ICMP messages sent.


```

root@6f47349eede5:/# ping -I 172.20.0.4 172.20.0.2
PING 172.20.0.2 (172.20.0.2) from 172.20.0.4 : 56(84) bytes of data.
64 bytes from 172.20.0.2: icmp_seq=1 ttl=64 time=0.135 ms
64 bytes from 172.20.0.2: icmp_seq=2 ttl=64 time=0.138 ms
64 bytes from 172.20.0.2: icmp_seq=3 ttl=64 time=0.128 ms
64 bytes from 172.20.0.2: icmp_seq=4 ttl=64 time=0.118 ms
64 bytes from 172.20.0.2: icmp_seq=5 ttl=64 time=0.138 ms
64 bytes from 172.20.0.2: icmp_seq=6 ttl=64 time=0.130 ms
64 bytes from 172.20.0.2: icmp_seq=7 ttl=64 time=0.137 ms
64 bytes from 172.20.0.2: icmp_seq=8 ttl=64 time=0.103 ms
64 bytes from 172.20.0.2: icmp_seq=9 ttl=64 time=0.136 ms
64 bytes from 172.20.0.2: icmp_seq=10 ttl=64 time=0.147 ms
64 bytes from 172.20.0.2: icmp_seq=11 ttl=64 time=0.104 ms
64 bytes from 172.20.0.2: icmp_seq=12 ttl=64 time=0.146 ms
64 bytes from 172.20.0.2: icmp_seq=13 ttl=64 time=0.139 ms
64 bytes from 172.20.0.2: icmp_seq=14 ttl=64 time=0.148 ms
64 bytes from 172.20.0.2: icmp_seq=15 ttl=64 time=0.128 ms
64 bytes from 172.20.0.2: icmp_seq=16 ttl=64 time=0.129 ms
^C
--- 172.20.0.2 ping statistics ---
16 packets transmitted, 16 received, 0% packet loss, time 15356ms
rtt min/avg/max/mdev = 0.103/0.131/0.148/0.013 ms
root@6f47349eede5:/#

```

Figure: Packet lost

- **Case 2:** DUT is not responding to icmp echo messages by HOST1(FAIL CASE) Below image can be used as a reference that HOST1 is getting no response from DUT to ICMP messages sent.

```

root@6f47349eede5:/# ping -I 172.20.0.4 172.20.0.2
PING 172.20.0.2 (172.20.0.2) from 172.20.0.4 : 56(84) bytes of data.
^C
--- 172.20.0.2 ping statistics ---
16 packets transmitted, 0 received, 100% packet loss, time 15360ms
root@6f47349eede5:/#

```

Securing Networks

Figure: 100% Packet lost

- **Test Case Number:** 02
- a. **Test Case Name:** TC2_PACKET_FILTERING_HOST2
- b. **Test Case Description:** As per the Rules mentioned in the table, DUT should not respond to ICMP messages sent from HOST2.
- c. **Execution Steps:**
 - Check that precondition is already satisfied by the DUT.
 - Use the command line of the HOST2 to ping the DUT.
 - Use the following command
ping -I <HOST2 IP Address> <DUT IP Address>
 - Capture the response on the HOST2 for the above command.
- d. **Test Observations:**

- **Case 1:** DUT is not responding to ICMP echo messages by HOST2 (PASS CASE)

Below image can be used as a reference that HOST1 is getting no response from DUT to icmp messages sent.

```
root@edf2b545e90f:/# ping -I 172.20.0.3 172.20.0.2
PING 172.20.0.2 (172.20.0.2) from 172.20.0.3 : 56(84) bytes of data.
^C
--- 172.20.0.2 ping statistics ---
12 packets transmitted, 0 received, 100% packet loss, time 11243ms
root@edf2b545e90f:/#
```

Figure: 100% Packet Loss

- **Case 2:** DUT's response is positive for the HOST2's ICMP messages (FAIL CASE)

Below image can be used as a reference that HOST2 is getting positive response from DUT to icmp messages sent.

- **Test Case Number:** 03

```
root@edf2b545e90f:/# ping -I 172.20.0.3 172.20.0.2
PING 172.20.0.2 (172.20.0.2) from 172.20.0.3 : 56(84) bytes of data.
64 bytes from 172.20.0.2: icmp_seq=1 ttl=64 time=0.141 ms
64 bytes from 172.20.0.2: icmp_seq=2 ttl=64 time=0.149 ms
64 bytes from 172.20.0.2: icmp_seq=3 ttl=64 time=0.164 ms
64 bytes from 172.20.0.2: icmp_seq=4 ttl=64 time=0.154 ms
64 bytes from 172.20.0.2: icmp_seq=5 ttl=64 time=0.152 ms
64 bytes from 172.20.0.2: icmp_seq=6 ttl=64 time=0.151 ms
^C
--- 172.20.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5121ms
rtt min/avg/max/mdev = 0.141/0.151/0.164/0.006 ms
root@edf2b545e90f:/#
```

Figure : No Packet Loss

- **Test Case Name:** TC3_ENABLE/DISABLE_FILTERING

a. **Test Case Description:**

- DUT should have a functionality to enable or disable each filter.
- DUT should disable the desired rule for the corresponding filter.

b. **Execution Steps:**

- Check that precondition is already satisfied by the DUT.
- Use the command line of the HOST2 to ping the DUT.
- Use the following command

iptables -L

- The list of rules will be shown in the command line shown below:

```

root@4c0cc631b92b:/# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                destination
ACCEPT     icmp -- ausf.test_net         anywhere    icmp echo-request
DROP       icmp -- udm.test_net      anywhere    icmp echo-request

Chain FORWARD (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination

```

- use the below command to delete 2nd the rule.
iptables -D INPUT -p icmp --icmp-type 8 -s 172.20.0.4 -j DROP
- use the below command to see the rules listed in iptables and capture the result.
iptables -L

c. Test Observations:

- **CASE 1:** The desired rule which we intend to delete gets deleted (PASS CASE) See that the second rule got deleted from the table of rules

```

root@4c0cc631b92b:/# iptables -D INPUT -p icmp --icmp-type 8 -s 172.20.0.4 -j DROP
root@4c0cc631b92b:/# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                destination
ACCEPT     icmp -- ausf.test_net         anywhere    icmp echo-request

Chain FORWARD (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
root@4c0cc631b92b:/#

```

- **CASE 2:** The desired rule which we intend to delete is not deleted (FAIL CASE)
- **Test Case Number:** 04

a. **Test Case Name:** TC4_FILTERING_COUNTER_RESET

b. **Test Case Description:** DUT should have a functionality to reset the counter available to count the data packet received under each rule.

c. Execution Steps:

- Check that precondition is already satisfied by the DUT.
- Use the command below to see the current counter value for each filter in DUT.

iptables -vL

- The output like this will be displayed as per the below figure.

```

root@4c0cc631b92b:/# iptables -vL
Chain INPUT (policy ACCEPT 4 packets, 397 bytes)
 pkts bytes target     prot opt in     out     source                destination
  15 1260 ACCEPT     icmp -- any    any    ausf.test_net         anywhere    icmp echo-request
   8  672 DROP      icmp -- any    any    udm.test_net          anywhere    icmp echo-request

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source                destination

Chain OUTPUT (policy ACCEPT 13 packets, 1153 bytes)
 pkts bytes target     prot opt in     out     source                destination

```

Figure: List of Rules Displayed in the command line

- Use the command below to set the counters to zero.

iptables -Z

- Use the command below to see the current counter value for each filter in DUT and capture the result displayed in the command line.

iptables -vL

d. Test Observations:

- **CASE 1:** The counters for each rule are set to zero after the above command is run (PASS CASE) Please see the below screen shot for Reference.

```
root@4c0cc631b92b:/# iptables -Z
root@4c0cc631b92b:/# iptables -vL
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source            destination
    0    0 ACCEPT    icmp -- any    any    ausf.test_net     anywhere          icmp echo-request
    0    0 DROP      icmp -- any    any    udm.test_net      anywhere          icmp echo-request

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source            destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source            destination
root@4c0cc631b92b:/#
```

Figure: All the Counters for the Filter Rules are set to Zero

- **CASE 2:** The counter for each rule is not set to zero after the above command is run (FAIL CASE)

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL (Please see Note Below)
1	TC1_PACKET_FILTERING_HOST1	
2	TC2_PACKET_FILTERING_HOST2	
3	TC3_ENABLE/DISABLE_FILTERING	
4	TC4_FILTERING_COUNTER_RESET	
FINAL RESULT		

NOTE: CRITERIA FOR PASS/FAIL

- If any one of the Test case Failure occurs: FAIL
- If all Test case is passed: PASS

2.7.2TSTP for Evaluation of Traffic Separation (2.7.2 of CSR)

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.7.2
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 7: Network Services
2. **<Security Requirement No & Name >** 2.7.2 Traffic Separation
3. **<Requirement Description: >**

The DUT shall support the physical or logical separation of traffic belonging to different network domains. For example, O&M traffic and control plane traffic belong to different network domains. See RFC 3871 for further information.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.5.1].

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)


```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

```
Terminator Jul 2
root@e55
root@293deb7a309: /70x38
pratik@pratik-HP-Laptop-15g-br0xx:~$ docker run -t1 --rm --name nf1 --
cap-add NET_ADMIN --network Sgtest -v /home/pratik/Core_Ran_system/SG/
:/code Sgtestbed.docker.local:5000/g3_final_common
root@293deb7a309: /# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.2 netmask 255.255.0.0 broadcast 172.18.255.255
    ether 02:42:ac:12:00:02 txqueuelen 0 (Ethernet)
    RX packets 79 bytes 16746 (16.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.16.1.2 netmask 255.255.255.0 broadcast 172.16.1.255
    ether 02:42:ac:10:01:02 txqueuelen 0 (Ethernet)
    RX packets 38 bytes 7619 (7.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@293deb7a309: /#
```

- DUT is having 2 logical/physical interface IN OUR SETUP
- IP address of interface 1: 172.18.0.2
- IP address of interface 2: 172.16.1.2
- NF2 and NF3 are the two network functions connected at 2 separate interfaces. NF2 and NF3 here are separate network functions

6. Preconditions

- NF2 and NF3 are separate network functions, and they must be connected at 2 separate interfaces of the DUT.
- Command Line Interfaces of DUT, NF2 and NF3 must be available.

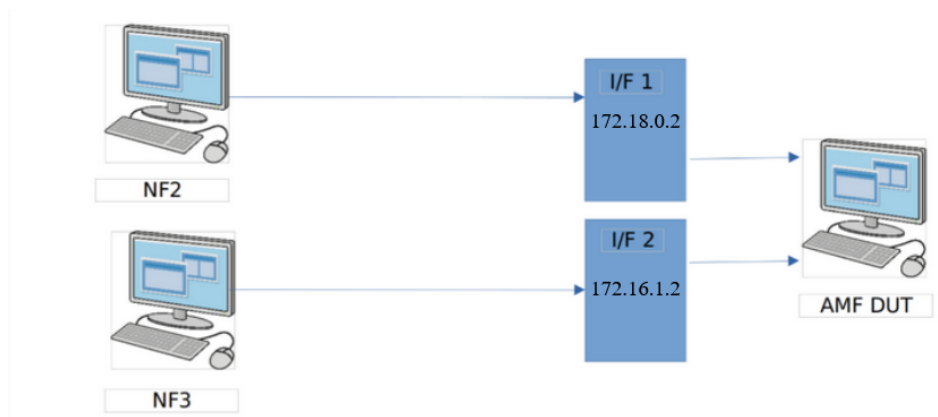
7. **Test Objective:** The DUT should support the physical or logical separation of traffic belonging to different network domains.

8. Test Plan:

8.1 Number of Test Scenarios:

8.1.1 Test Scenario for Traffic Separation

8.2 Test Setup Diagram:



8.3 **Tools Used:** Command line interface of NF2, NF3 and DUT.

8.4 **Test Execution Steps**

- NF2 and NF3 are the two-network function that will try to send the ICMP messages to SMF over separate interfaces.
- DUT should allow the traffic to separately handle at the two interfaces.
- NF2 will be sending the traffic to DUT over interface I/F 1
- NF3 will be sending the traffic of icmp messages to DUT over interface with ip address I/F 2

9. **Expected Results for Pass:**

- The ping messages for the NF2 will be successfully transmitted without any loss when sent to the interface 1 of the DUT. But the ping messages fail to reach the interface 2 of the DUT.
- The ping messages for the NF3 will be successfully transmitted without any loss when sent to the interface 2 of the DUT. But the ping messages fail to reach the interface 1 of the DUT.
- This is because the traffic is separated by DUT to different domains for handling the icmp messages from different Nfs.

Securing Networks

10. **Expected Format of Evidence:** Evidence in the form of pcap file or the screenshot of the command line can be submitted.

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC_TRAFFIC_SEPARATION

b. **Test Case Description:** The DUT should support the physical or logical separation of traffic belonging to different network domains.

c. **Execution Steps:**

- Tester will ping the DUT using command line of the NF2 ping 172.18.0.2
- Tester will ping the DUT using command line of the NF3 ping 172.16.1.2

- Tester will ping the DUT using the command line of NF2 but this time at the other interface ping 172.16.1.2
- Tester will ping the DUT using the command line of NF3 but this time at the other interface. Ping 172.18.0.2

d. Test Observations:

- The ping messages for the NF2 will be successfully transmitted without any loss when sent to the interface 1 of the SMF DUT. But the ping messages fail to reach the interface 2 of the SMF DUT.
- The ping messages for the NF3 will be successfully transmitted without any loss when sent to the interface 2 of the SMF DUT. But the ping messages fail to reach the interface 1 of the SMF DUT.
- This is because the traffic is separated by DUT to different domains for handling the ICMP messages from different NFs.

The screenshot shows two terminal windows. The left window is titled 'root@293de6b7a309: / 70x38' and shows the configuration of a Docker container named 'nf1'. It lists network interfaces eth0 and eth1 with their respective IP addresses (172.18.0.2 and 172.16.1.2) and MTU values. The right window is titled 'root@6a394ff6afd1: / 70x18' and shows the configuration of a Docker container named 'nf2'. It lists network interfaces eth0 and eth1 with their respective IP addresses (172.18.0.2 and 172.16.1.2) and MTU values. Below the configuration, it shows the results of ping tests. For NF2, ping to 172.18.0.2 is successful (2 packets transmitted, 2 received, 0% packet loss), while ping to 172.16.1.2 fails (3 packets transmitted, 0 received, 100% packet loss). For NF3, ping to 172.16.1.2 is successful (2 packets transmitted, 2 received, 0% packet loss), while ping to 172.18.0.2 fails (3 packets transmitted, 0 received, 100% packet loss).

Figure 1: NF2 and NF3 try to ping the SMF DUT at different Interfaces

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_TRAFFIC_SEPARATION		

- **PASS:** If the network traffic is sent successfully to the predefined interface as defined in the configuration of DUT.
- **FAIL:** If network traffic is sent successfully to the other interfaces which are not there in predefined configuration of the DUT.

2.7.3 TSTP Report for Traffic Protection –Anti-Spoofing

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.7.3
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **ITSAR Section No & Name:** Section 7: Network Services

2. **Security Requirement No & Name:** 2.7.3: Traffic Protection –Anti-Spoofing

3. **Requirement Description:**

SMF shall not process IP Packets if their source address is not reachable via the incoming interface. Implementation example: Use of "Reverse Path Filter" (RPF) provides this function. [Reference: TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.3.1.1]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

For checking if RPF is enabled:

```
NP
root@6872c2825c3c:/proc/sys/net/ipv4# cat /proc/sys/net/ipv4/conf/all/rp_filter
2
root@6872c2825c3c:/proc/sys/net/ipv4#
```

0 – disabled ,1 – strict filtering, 2 – loose filtering

Routes at the Network product

```
NP
root@6872c2825c3c:/proc/sys/net/ipv4# ip r
default via 172.19.0.2 dev eth1
172.18.0.0/16 dev eth0 proto kernel scope link src 172.18.0.4
172.19.0.0/16 dev eth1 scope link
172.20.0.2 via 172.18.0.2 dev eth0
```

Configuration of DUT:

- Three virtual machines with a network traffic analyzer like wireshark or tcpdump.
- Configurations of VMs should be such that traffic from vm1->vm3 should go via vm2.
- VM2 has two network interfaces, one connected to vm1 and other to vm3.
- Vm1 and vm3 should be able to pass their traffic to vm2, by default.

6. **Preconditions:**

● **Test Case Description:**

To verify that the network product provides anti-spoofing function, that is, before a packet is processed, the network product checks whether the source IP of the received packet is reachable through the interface it comes in.

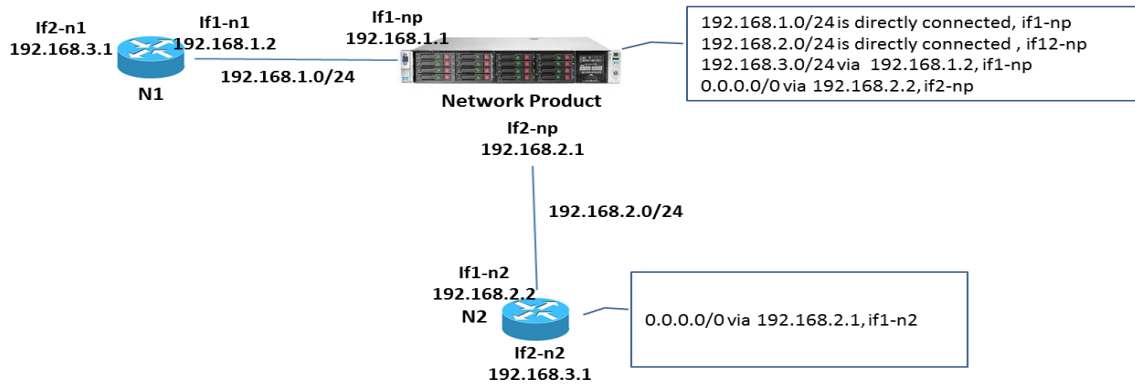
To verify that if the received packet source address is not routable through the interface on which it comes, then the network product drops this packet.

● **Pre-Conditions:**

A node N1 is available with:

- Two interfaces named respectively if1-n1 connected to the network product and if2-n1 to which the tester connects a tester machine
- routing capabilities
- if2-n1 has a static IP address (e.g. 192.168.3.1 belonging to the subnet 192.168.3.0/24)
- A node N2 is available with:
- Two interfaces named respectively if1-n2 connected to the network product and if2-n2 to which the tester connects a tester machine
- Routing capabilities. In particular N2 has a default route to if1-np subnet via if2-np (e.g. 192.168.2.1)
- if2-n2 has a static IP address . This ip is the same as if2-n1 (e.g. 192.168.3.1 belonging to the subnet 192.168.3.0/24)
- The network product has at least 2 enabled interfaces said if1-np and if2-np:
- The interface if1-np is connected to interface if1-n1 of the node N1 on the subnet, e.g., 192.168.1.0/24.
- The interface if2-np is connected to interface if1-n2 of the node N2 on the subnet, e.g., 192.168.2.0/24.

- The network product is configured with a static route for the subnet where if2-n1 is connected to (e.g. 192.168.3.0/24), so this subnet can be reached via if1-n1 through if1-np.



- The vendor shall declare, in the documentation accompanying the network product, the supported anti-spoofing mechanism (e.g. RPF or similar function) and if it is enabled for all interfaces (e.g. net.ipv4.conf.all.rp_filter = 1 and net.ipv4.conf.default.rp_filter = 1 in the linux sysctl.conf file) or per interface bases.
- The vendor shall declare if the dropped packets can be logged and how to enable this logging
- The tester has administrator privileges
- A tester machine is available and configured with:
 - A static IP address belonging to the subnet where if2-n1 and if2-n2 are connected to (e.g. 192.168.3.2/24)
 - A default gateway set to if2-n1 and if2-n2 IP Address (e.g. 192.168.3.1)
 - A network traffic analyser (e.g. tcpdump) on the network product is available

7. **Test Objective:**

Securing Networks

8. **Test Plan:**

8.1 **Number of Test Scenarios: 1**

- 8.1.1 Test scenario when the password recovery is unset in the /etc/default/grub file (for Linux).
- 8.1.2 Any other configuration file based on the OS being used. Test whether able to recover passwords for system/root.

8.2 **Tools Used:** Command line

8.3 **Test Execution Steps:**

a. **Test Case Number: 1**

b. **Test Case Name: TC_IP_SPOOFING_MITIGATION**

c. **Test Case Description:** The network product supports an anti-spoofing mechanism (e.g. the RPF function) and it accepts a packet only if it reaches the network product on the expected interface (i.e. this packet has a source ip address belonging to the same network as the interface where it came in or if it is routable through the interface on which it came in), otherwise it discards the packet.

d. **Tools Used:** virtual machines installed on a linux device.

e. **Execution Steps:**

1. The tester starts to send ping messages to the if1-np interface of the network product.

Ping sent by tester on connecting to n1

```
N1
172.20.0.0/16 dev eth1 proto kernel scope link src 172.20.0.2
root@fda402e66d4a:/# ping 172.18.0.4
PING 172.18.0.4 (172.18.0.4) 56(84) bytes of data.
64 bytes from 172.18.0.4: icmp_seq=1 ttl=64 time=0.171 ms
64 bytes from 172.18.0.4: icmp_seq=2 ttl=64 time=0.104 ms
64 bytes from 172.18.0.4: icmp_seq=3 ttl=64 time=0.113 ms
64 bytes from 172.18.0.4: icmp_seq=4 ttl=64 time=0.094 ms
^C
--- 172.18.0.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3048ms
rtt min/avg/max/mdev = 0.094/0.120/0.171/0.031 ms
root@fda402e66d4a:/# ping 172.18.0.4
PING 172.18.0.4 (172.18.0.4) 56(84) bytes of data.
64 bytes from 172.18.0.4: icmp_seq=1 ttl=64 time=0.163 ms
64 bytes from 172.18.0.4: icmp_seq=2 ttl=64 time=0.110 ms
64 bytes from 172.18.0.4: icmp_seq=3 ttl=64 time=0.108 ms
64 bytes from 172.18.0.4: icmp_seq=4 ttl=64 time=0.116 ms
64 bytes from 172.18.0.4: icmp_seq=5 ttl=64 time=0.106 ms
64 bytes from 172.18.0.4: icmp_seq=6 ttl=64 time=0.113 ms
64 bytes from 172.18.0.4: icmp_seq=7 ttl=64 time=0.118 ms
64 bytes from 172.18.0.4: icmp_seq=8 ttl=64 time=0.111 ms
64 bytes from 172.18.0.4: icmp_seq=9 ttl=64 time=0.107 ms
^C
--- 172.18.0.4 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8185ms
rtt min/avg/max/mdev = 0.106/0.116/0.163/0.022 ms
root@fda402e66d4a:/#
```

2. The tester verifies, through the network traffic analyser, that the ping correctly reaches the if1-np interface and that responses are sent back.

Responses sent back by Network product from if1-np:

```
NP
listening on br0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
root@6872c2825c3c:/# tcpdump -i eth0 -p icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
15:08:25.958513 IP n1.5gmec > 6872c2825c3c: ICMP echo request, id 8, seq 1, length 64
15:08:25.958566 IP 6872c2825c3c > n1.5gmec: ICMP echo reply, id 8, seq 1, length 64
15:08:26.975746 IP n1.5gmec > 6872c2825c3c: ICMP echo request, id 8, seq 2, length 64
15:08:26.975777 IP 6872c2825c3c > n1.5gmec: ICMP echo reply, id 8, seq 2, length 64
15:08:27.999724 IP n1.5gmec > 6872c2825c3c: ICMP echo request, id 8, seq 3, length 64
15:08:27.999755 IP 6872c2825c3c > n1.5gmec: ICMP echo reply, id 8, seq 3, length 64
15:08:29.023724 IP n1.5gmec > 6872c2825c3c: ICMP echo request, id 8, seq 4, length 64
15:08:29.023754 IP 6872c2825c3c > n1.5gmec: ICMP echo reply, id 8, seq 4, length 64
15:08:30.047702 IP n1.5gmec > 6872c2825c3c: ICMP echo request, id 8, seq 5, length 64
15:08:30.047732 IP 6872c2825c3c > n1.5gmec: ICMP echo reply, id 8, seq 5, length 64
15:08:31.071718 IP n1.5gmec > 6872c2825c3c: ICMP echo request, id 8, seq 6, length 64
15:08:31.071749 IP 6872c2825c3c > n1.5gmec: ICMP echo reply, id 8, seq 6, length 64
15:08:32.095705 IP n1.5gmec > 6872c2825c3c: ICMP echo request, id 8, seq 7, length 64
15:08:32.095737 IP 6872c2825c3c > n1.5gmec: ICMP echo reply, id 8, seq 7, length 64
15:08:33.119698 IP n1.5gmec > 6872c2825c3c: ICMP echo request, id 8, seq 8, length 64
15:08:33.119729 IP 6872c2825c3c > n1.5gmec: ICMP echo reply, id 8, seq 8, length 64
15:08:34.143724 IP n1.5gmec > 6872c2825c3c: ICMP echo request, id 8, seq 9, length 64
15:08:34.143754 IP 6872c2825c3c > n1.5gmec: ICMP echo reply, id 8, seq 9, length 64
```

3. The tester disconnects the tester machine from if2-n1 interface of the node N1 and reconnects it to the interface if2-n2 of the node N2:

- The testers use the same network configuration of the tester machine.
- The tester sends ping messages to the if1-np interface of the network product.

```

N2
root@9c138a6bed62:/proc/sys/net/ipv4# ping 172.18.0.4 -I eth2
PING 172.18.0.4 (172.18.0.4) from 172.20.0.2 eth2: 56(84) bytes of data.
From 172.20.0.2 icmp_seq=1 Destination Host Unreachable
From 172.20.0.2 icmp_seq=2 Destination Host Unreachable
From 172.20.0.2 icmp_seq=3 Destination Host Unreachable
From 172.20.0.2 icmp_seq=4 Destination Host Unreachable
From 172.20.0.2 icmp_seq=5 Destination Host Unreachable
From 172.20.0.2 icmp_seq=6 Destination Host Unreachable
From 172.20.0.2 icmp_seq=7 Destination Host Unreachable
From 172.20.0.2 icmp_seq=8 Destination Host Unreachable
From 172.20.0.2 icmp_seq=9 Destination Host Unreachable
From 172.20.0.2 icmp_seq=10 Destination Host Unreachable
From 172.20.0.2 icmp_seq=11 Destination Host Unreachable
From 172.20.0.2 icmp_seq=12 Destination Host Unreachable
From 172.20.0.2 icmp_seq=13 Destination Host Unreachable
From 172.20.0.2 icmp_seq=14 Destination Host Unreachable
From 172.20.0.2 icmp_seq=15 Destination Host Unreachable
From 172.20.0.2 icmp_seq=16 Destination Host Unreachable
From 172.20.0.2 icmp_seq=17 Destination Host Unreachable
From 172.20.0.2 icmp_seq=18 Destination Host Unreachable
From 172.20.0.2 icmp_seq=19 Destination Host Unreachable
From 172.20.0.2 icmp_seq=20 Destination Host Unreachable
^C
--- 172.18.0.4 ping statistics ---
23 packets transmitted, 0 received, +20 errors, 100% packet loss, time 22478ms
pipe 4
root@9c138a6bed62:/proc/sys/net/ipv4#

```

- The tester verifies, through the network traffic analyser, that the pings reach the if1-np interface of the network product, but they are dropped and no response is sent back since the source of the received packet is not reachable through the interface it came in.

Network packets dropped at if1-np interface of the network product

```

NP
root@6872c2825c3c:/proc/sys/net/ipv4# tcpdump -i eth1 -p icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
^

```

- The tester sends ping messages to the if2-np interface of the network product.

```

N2
root@9c138a6bed62:/proc/sys/net/ipv4# ping 172.18.0.4 -I eth2
PING 172.18.0.4 (172.18.0.4) from 172.20.0.2 eth2: 56(84) bytes of data.
From 172.20.0.2 icmp_seq=1 Destination Host Unreachable
From 172.20.0.2 icmp_seq=2 Destination Host Unreachable
From 172.20.0.2 icmp_seq=3 Destination Host Unreachable
From 172.20.0.2 icmp_seq=4 Destination Host Unreachable
From 172.20.0.2 icmp_seq=5 Destination Host Unreachable
From 172.20.0.2 icmp_seq=6 Destination Host Unreachable
From 172.20.0.2 icmp_seq=7 Destination Host Unreachable
From 172.20.0.2 icmp_seq=8 Destination Host Unreachable
From 172.20.0.2 icmp_seq=9 Destination Host Unreachable
From 172.20.0.2 icmp_seq=10 Destination Host Unreachable
From 172.20.0.2 icmp_seq=11 Destination Host Unreachable
From 172.20.0.2 icmp_seq=12 Destination Host Unreachable
From 172.20.0.2 icmp_seq=13 Destination Host Unreachable
From 172.20.0.2 icmp_seq=14 Destination Host Unreachable
From 172.20.0.2 icmp_seq=15 Destination Host Unreachable
From 172.20.0.2 icmp_seq=16 Destination Host Unreachable
From 172.20.0.2 icmp_seq=17 Destination Host Unreachable
From 172.20.0.2 icmp_seq=18 Destination Host Unreachable
From 172.20.0.2 icmp_seq=19 Destination Host Unreachable
From 172.20.0.2 icmp_seq=20 Destination Host Unreachable
^C
--- 172.18.0.4 ping statistics ---
23 packets transmitted, 0 received, +20 errors, 100% packet loss, time 22478ms
pipe 4
root@9c138a6bed62:/proc/sys/net/ipv4#

```

- The tester verifies, through the network traffic analyser, that the pings reach the if2-np interface of the network product, but they are dropped and no response is sent back since there is a default route via if2-np.

Network packets dropped at if1-np interface of network product

```
NP
root@6872c2825c3c:/proc/sys/net/ipv4# tcpdump -i eth1 -p icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
root@6872c2825c3c:/proc/sys/net/ipv4#
```

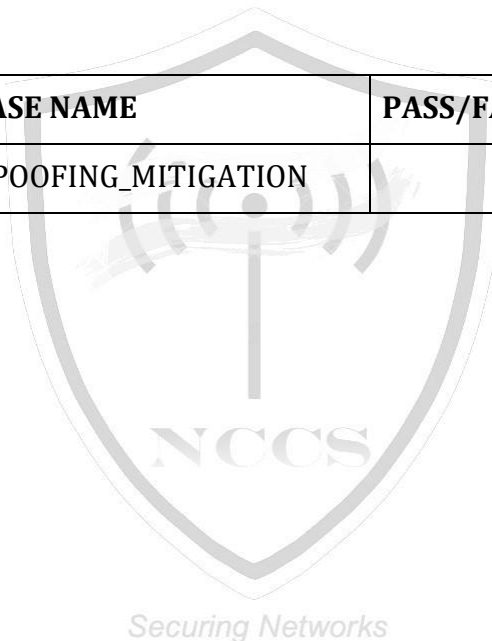
- If the dropped packets are logged, the testers verifies that these packets are recorded.

f. Test Observations:

In presence of an anti-spoofing mechanism (such as RPF filter), the accepts a packet only if it reaches the network product on the expected interface (i.e. this packet has a source ip address belonging to the same network as the interface where it came in or if it is routable through the interface on which it came in), otherwise it discards the packet.

9. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_IP_SPOOFING_MITIGATION		



2.7.4 TSTP for Evaluation of GTP-C Filtering (2.7.4 of CSR)

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.7.4
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 7: Network Services
2. **<Security Requirement No & Name >** 2.7.4 GTP-C Filtering
3. **<Requirement Description: >**

The following capability is conditionally required:

- For each message of a GTP-C-based protocol, it shall be possible to check whether the sender of this message is authorized to send a message pertaining to this protocol.
- At least the following actions should be supported when the check is satisfied:
- Discard: the matching message is discarded.
- Accept: the matching message is accepted.
- Account: the matching message is accounted for, i.e., a counter for the rule is incremented.

This action can be combined with the previous ones. This feature is useful to monitor traffic before its blocking. This requirement is conditional in the following sense: It is required that at least one of the following two statements holds:

- SMF supports the capability described above, and this is stated in the product documentation.
- The SMF's documentation states that the capability is not supported and that the SMF needs to be deployed together with a separate entity that provides the capability described above.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.6.2.3]

4. **DUT Confirmation Details**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version

- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)


```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

There might be various ways of having mentioned functionalities integrated with the system

- Iptables
- Ufw
- GTP Firewalls

a. For Iptables

Command used: **iptables -V** (To get version information)

```
iptables v1.8.4 (legacy)
```

b. Similarly check for other tools used...

6. Preconditions

- The network product has at least one physical interface named if1 and may have another physical interface named if2 .
- The tester has the privileges to configure GTP-C filtering on the network product.
- The manufacturer declares that the GTP-C filtering is supported.
- The manufacturer includes a guideline to configure the GTP-C filtering in the documentation accompanying the network product.
- A network traffic generator or a pcap file containing the GTP-C messages is available.
- A network traffic analyser on the network product (e.g. tcpdump) is available.

7. **Test Objective**:- To verify that the network product provides filtering functionalities for incoming GTP-C messages. In particular this test case verifies that:

1. The network product provides filtering of incoming GTP-C messages on any interface.
2. It is possible to block all GTP-C messages on those network product interfaces where they are unwanted.
3. It is possible to specify defined actions for each rule.

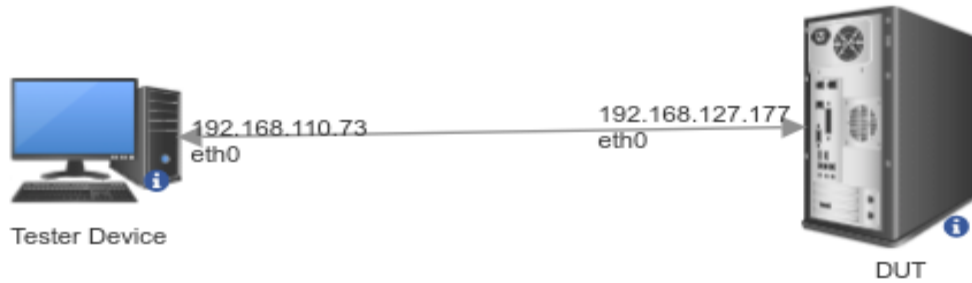
8. Test Plan

8.1. Number of Test Scenarios:

8.1.1. Test Scenario for Iptables:

- This test scenario is regarding Iptables Filtering Firewall (Additional Test scenarios based on the OEM document)

8.2. Test Bed Diagram



8.3. **Tools Required:** Wireshark and GTPing

8.4. **Test Execution Steps**

- Launch the Wireshark app on the tester device.
- Send gtp echo requests from tester device to DUT on if1
- Verify that gtp echo replies are received (Case 1)
- Change rule such that DUT drops all gtp packets
- Verify that no gtp echo replies are received but gtp requests are received by the DUT (Case 2)
- Change rule such that DUT drops all gtp packets from IP of tester
- Verify that no gtp echo replies are received but gtp requests are received by the DUT (Case 3)
- Change rule such that DUT drops some packets if provided threshold is crossed
- Verify that some gtp echo replies are received but gtp requests are received by the DUT (Case 4)

9. **Expected Results for Pass**

- **Case 1:** GTP Echo Replies are received by the tester
- **Case 2:** GTP Echo Replies are not received by the tester
- **Case 3:** GTP Echo Replies are not received by the tester
- **Case 4:** GTP Echo Replies are received by the tester but some packet loss is observed if sending frequency is beyond a particular threshold

10. **Expected Format of Evidence:** Screenshots of Wireshark, Terminal and pcap files

11. **Test Execution:**

➤ **Test Case Number:** 01

- a. **Test Case Name:** TC1_GTP_C_FOR_IPTABLES
- b. **Test Case Description:** Tester configures the rules in iptables of DUT to test the effect on incoming gtp messages from tester device
- c. **Execution Steps:**
 - Tester launches wireshark on tester device
 - Tester enables logs for gtp-c using following command

- `iptables -A INPUT -p udp --destination-port 2123 -j LOG`
- Tester uses following command to send gtp echo messages to the DUT
- `gtping -p 2123 <DUT_IP_ADDR>`

5...	14.83...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
5...	14.83...	192.168.127.177	192.168.110.73	GTP	54 Echo response
5...	15.83...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
5...	15.83...	192.168.127.177	192.168.110.73	GTP	54 Echo response
5...	16.83...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
5...	16.83...	192.168.127.177	192.168.110.73	GTP	54 Echo response

```
30 packets transmitted, 30 received, 0% packet loss, time 2900ms
0 out of order, 0 dups, 0 connection refused, 0 ICMP error
```

- Tester adds the rule in DUT to block any gtp-c traffic using following command
- `iptables -A INPUT -p udp --destination-port 2123 -j DROP`
- Tester uses following command to send gtp echo messages to the DUT
- `gtping -p 2123 <DUT_IP_ADDR>`

3...	2.184...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
4...	3.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
6...	4.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
8...	5.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request

```
30 packets transmitted, 0 received, 100% packet loss, time 10000ms
0 out of order, 0 dups, 0 connection refused, 0 ICMP error
```

- Tester can check the logs of the message received by DUT using following command
- `cat /var/log/kern.log`

```
Jun 27 03:47:36 pratik-Super-Server kernel: [564046.372040] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08
7.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=39255 DF PROTO=UDP SPT=33949 DPT=2123 LEN=20
Jun 27 03:47:37 pratik-Super-Server kernel: [564047.371852] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08
7.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=39256 DF PROTO=UDP SPT=33949 DPT=2123 LEN=20
Jun 27 03:47:38 pratik-Super-Server kernel: [564048.371846] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08
7.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=39257 DF PROTO=UDP SPT=33949 DPT=2123 LEN=20
```

- Tester deletes the previous rule using following command
- `iptables -D INPUT -p udp --destination-port 2123 -j DROP`
- Tester adds the rule in DUT to block gtp-c traffic from tester's IP address using following command
- `iptables -A INPUT -s <tester_IP_ADDR> -p udp --destination-port 2123 -j DROP`
- Tester uses following command to send gtp echo messages to the DUT
- `gtping -p 2123 <DUT_IP_ADDR>`

3...	2.184...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
4...	3.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
6...	4.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
8...	5.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request

```
30 packets transmitted, 0 received, 100% packet loss, time 10000ms
0 out of order, 0 dups, 0 connection refused, 0 ICMP error
```

- Tester can check the logs of the message received by DUT using following command
- `cat /var/log/kern.log`

```
Jun 27 03:47:36 pratik-Super-Server kernel: [564046.372040] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08
7.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=39255 DF PROTO=UDP SPT=33949 DPT=2123 LEN=20
Jun 27 03:47:37 pratik-Super-Server kernel: [564047.371852] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08
7.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=39256 DF PROTO=UDP SPT=33949 DPT=2123 LEN=20
Jun 27 03:47:38 pratik-Super-Server kernel: [564048.371846] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08
7.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=39257 DF PROTO=UDP SPT=33949 DPT=2123 LEN=20
```

- Tester deletes the previous rule using following command
 - `iptables -D INPUT -s <tester_IP_ADDR> -p udp --destination-port 2123 -j DROP`
- Tester uses following command to limit the count of packets that can be received by DUT to be 1/sec
 - `iptables -A INPUT -s <tester_IP_ADDR> -p udp --destination-port 2123 --match limit --limit 1/sec --limit-burst 2 --jump ACCEPT`
 - `iptables -A INPUT -j DROP`
- Tester uses following command to send gtp echo messages to the DUT
 - `gtping -i 0.1 -p 2123 <DUT_IP_ADDR>`

```

9... 6.855... 192.168.110.73 192.168.127.1... GTP 60 Echo request
9... 6.855... 192.168.127.177 192.168.110.73 GTP 54 Echo response
9... 6.955... 192.168.110.73 192.168.127.1... GTP 60 Echo request
9... 6.955... 192.168.127.177 192.168.110.73 GTP 54 Echo response
9... 7.055... 192.168.110.73 192.168.127.1... GTP 60 Echo request
9... 7.155... 192.168.110.73 192.168.127.1... GTP 60 Echo request
9... 7.255... 192.168.110.73 192.168.127.1... GTP 60 Echo request

30 packets transmitted, 4 received, 86% packet loss, time 3000ms
0 out of order, 0 dups, 0 connection refused, 0 ICMP error
rtt min/avg/max/mdev = 0.561/0.657/0.798/0.096 ms

```

- Tester can check the logs of the message received by DUT using following command
 - `cat /var/log/kern.log`

```

Jun 27 03:47:36 pratik-Super-Server kernel: [564046.372040] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08:7:177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=39255 DF PROTO=UDP SPT=33949 DPT=2123 LEN=20
Jun 27 03:47:37 pratik-Super-Server kernel: [564047.371852] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08:7:177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=39256 DF PROTO=UDP SPT=33949 DPT=2123 LEN=20
Jun 27 03:47:38 pratik-Super-Server kernel: [564048.371846] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08:7:177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=39257 DF PROTO=UDP SPT=33949 DPT=2123 LEN=20

```

d. Test Observations

- **Case 1:** GTP Echo Replies are received by the tester and no packet loss occurs
- **Case 2:** GTP Echo Replies are not received by the tester but GTP Echo requests are received by the DUT. Packet loss is 100%
- **Case 3:** GTP Echo Replies are not received by the tester but GTP Echo requests are received by the DUT. Packet loss is 100%
- **Case 4:** GTP Echo Replies are received by the tester but some packet loss is observed

e. Evidence Provided:- Screenshots of Wireshark, Terminal and pcap files

12. Test Case Result:

SL. No	Test Case Name	PASS/FAIL	Remarks
1	TC1_GTP_C_FOR_IPTABLES		

(Tester has to execute the test in similar fashion for other tools)

2.7.5 TSTP for Evaluation of GTP-U Filtering

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.7.3
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1) <ITSAR Section No & Name> Section 7: Network Services

2) <Security Requirement No & Name > 2.7.5 GTP-U Filtering

3) <Requirement Description: > The following capability is conditionally required:

- For each message of a GTP-U-based protocol, it shall be possible to check whether the sender of this message is authorized to send a message pertaining to this protocol.
- At least the following actions should be supported when the check is satisfied:
- Discard: the matching message is discarded.
- Accept: the matching message is accepted.
- Account: the matching message is accounted for, i.e., a counter for the rule is incremented.

This action can be combined with the previous ones. This feature is useful to monitor traffic before its blocking. This requirement is conditional in the following sense: It is required that at least one of the following two statements holds:

- SMF supports the capability described above, and this is stated in the product documentation.
- The SMF's documentation states that the capability is not supported and that the SMF needs to be deployed together with a separate entity that provides the capability described above.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.6.2.4]

4) DUT Confirmation Details

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) DUT Configuration

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```


To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

There might be various ways of having mentioned functionalities integrated with the system

- Iptables
 - Ufw
 - GTP Firewalls
- c. For Iptables**

Command used: **iptables -V** (To get version information)

```
iptables v1.8.4 (legacy)
```

d. Similarly check for other tools used...

6) Preconditions

- The network product has at least one physical interface named if1 and may have another physical interface named if2 .
- The tester has the privileges to configure GTP-U filtering on the network product. - The manufacturer declares that the GTP-U filtering is supported.
- The manufacturer includes a guideline to configure the GTP-U filtering in the documentation accompanying the network product.
- A network traffic generator or a pcap file containing the GTP-U messages is available.
- A network traffic analyser on the network product (e.g. tcpdump) is available.

7) **Test Objective:-** To verify that the network product provides filtering functionalities for incoming GTP-U messages. In particular this test case verifies that:

- a) The network product provides filtering of incoming GTP-U messages on any interface.
- b) It is possible to block all GTP-U messages on those network product interfaces where they are unwanted.
- c) It is possible to specify defined actions for each rule.

8) Test Plan

8.1 Number of Test Scenarios:

8.1.1. Test Scenario for Iptables:

This test scenario is regarding Iptables Filtering Firewall (Additional Test scenarios based on the OEM document)

8.1.2. Test Bed Diagram



8.1.3. Tools Required Wireshark and GTPing

8.1.4. Test Execution Steps

- Launch the Wireshark app on the tester device.
- Send gtp echo requests from tester device to DUT on if1
- Verify that gtp echo replies are received (Case 1)
- Change rule such that DUT drops all gtp packets
- Verify that no gtp echo replies are received but gtp requests are received by the DUT (Case 2)
- Change rule such that DUT drops all gtp packets from IP of tester
- Verify that no gtp echo replies are received but gtp requests are received by the DUT (Case 3)
- Change rule such that DUT drops some packets if provided threshold is crossed
- Verify that some gtp echo replies are received but gtp requests are received by the DUT (Case 4)

9) Expected Results for Pass

- **Case 1:** GTP Echo Replies are received by the tester
- **Case 2:** GTP Echo Replies are not received by the tester
- **Case 3:** GTP Echo Replies are not received by the tester
- **Case 4:** GTP Echo Replies are received by the tester but some packet loss is observed if sending frequency is beyond a particular threshold

10) Expected Format of Evidence: Screenshots of Wireshark, Terminal and pcap files

11) Test Execution:

- **Test Case Number:** 01
 - a) **Test Case Name:** TC1_GTP_U_FOR_IPTABLES
 - b) **Test Case Description:** Tester configures the rules in iptables of DUT to test the effect on incoming gtp messages from tester device
 - c) **Execution Steps:**
 - Tester launches wireshark on tester device
 - Tester enables logs for gtp-u using following command

- `iptables -A INPUT -p udp --destination-port 2152 -j LOG`

- Tester uses following command to send gtp echo messages to the DUT

- `gtping -p 2152 <DUT_IP_ADDR>`

5...	14.83...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
5...	14.83...	192.168.127.177	192.168.110.73	GTP	54 Echo response
5...	15.83...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
5...	15.83...	192.168.127.177	192.168.110.73	GTP	54 Echo response
5...	16.83...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
5...	16.83...	192.168.127.177	192.168.110.73	GTP	54 Echo response

```
30 packets transmitted, 30 received, 0% packet loss, time 2900ms
0 out of order, 0 dups, 0 connection refused, 0 ICMP error
```

- Tester adds the rule in DUT to block any gtp-u traffic using following command

- `iptables -A INPUT -p udp --destination-port 2152 -j DROP`

- Tester uses following command to send gtp echo messages to the DUT

- `gtping -p 2152 <DUT_IP_ADDR>`

3...	2.184...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
4...	3.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
6...	4.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
8...	5.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request

```
30 packets transmitted, 0 received, 100% packet loss, time 10000ms
0 out of order, 0 dups, 0 connection refused, 0 ICMP error
```

- Tester can check the logs of the message received by DUT using following command

- `cat /var/log/kern.log`

```
PROTO=UDP SPT=40762 DPT=2152 LEN=20
Jun 27 03:08:30 pratik-Super-Server kernel: [561701.041961] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52781 DF
PROTO=UDP SPT=39569 DPT=2152 LEN=20
Jun 27 03:08:31 pratik-Super-Server kernel: [561702.042044] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52782 DF
PROTO=UDP SPT=39569 DPT=2152 LEN=20
Jun 27 03:08:32 pratik-Super-Server kernel: [561703.041980] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52783 DF
PROTO=UDP SPT=39569 DPT=2152 LEN=20
Jun 27 03:08:33 pratik-Super-Server kernel: [561704.041958] IN=eth0 OUT= MAC=0c:c4:7a:e7:e8:2d:ec:9b:8b:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52784 DF
PROTO=UDP SPT=39569 DPT=2152 LEN=20
```

- Tester deletes the previous rule using following command

- `iptables -D INPUT -p udp --destination-port 2152 -j DROP`

- Tester adds the rule in DUT to block gtp-u traffic from tester's IP address using following command

- `iptables -A INPUT -s <tester_IP_ADDR> -p udp --destination-port 2152 -j DROP`

- Tester uses following command to send gtp echo messages to the DUT

- `gtping -p 2152 <DUT_IP_ADDR>`

3...	2.184...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
4...	3.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
6...	4.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request
8...	5.186...	192.168.110.73	192.168.127.1...	GTP	60 Echo request

```
30 packets transmitted, 0 received, 100% packet loss, time 10000ms
0 out of order, 0 dups, 0 connection refused, 0 ICMP error
```

- Tester can check the logs of the message received by DUT using following command

- `cat /var/log/kern.log`

```

PROTOUDP SPT=49702 DPT=2152 LEN=20
Jun 27 03:08:30 pratik-Super-Server kernel: [561701.041961] IN=eth0 OUT= MAC=bc:c4:7a:e7:e8:2d:ec:9b:bb:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52781 DF
PROTOUDP SPT=39509 DPT=2152 LEN=20
Jun 27 03:08:31 pratik-Super-Server kernel: [561702.042044] IN=eth0 OUT= MAC=bc:c4:7a:e7:e8:2d:ec:9b:bb:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52782 DF
PROTOUDP SPT=39509 DPT=2152 LEN=20
Jun 27 03:08:32 pratik-Super-Server kernel: [561703.041980] IN=eth0 OUT= MAC=bc:c4:7a:e7:e8:2d:ec:9b:bb:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52783 DF
PROTOUDP SPT=39509 DPT=2152 LEN=20
Jun 27 03:08:33 pratik-Super-Server kernel: [561704.041958] IN=eth0 OUT= MAC=bc:c4:7a:e7:e8:2d:ec:9b:bb:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52784 DF

```

- Tester deletes the previous rule using following command
 - `iptables -D INPUT -s <tester_IP_ADDR> -p udp --destination-port 2152 -j DROP`
- Tester uses following command to limit the count of packets that can be received by DUT to be 1/sec
 - `iptables -A INPUT -s <tester_IP_ADDR> -p udp --destination-port 2152 --match limit --limit 1/sec --limit-burst 2 --jump ACCEPT`
 - `iptables -A INPUT -j DROP`
- Tester uses following command to send gtp echo messages to the DUT
 - `gtping -i 0.1 -p 2152 <DUT_IP_ADDR>`

```

9... 6.855... 192.168.110.73 192.168.127.1... GTP 60 Echo request
9... 6.855... 192.168.127.177 192.168.110.73 GTP 54 Echo response
9... 6.955... 192.168.110.73 192.168.127.1... GTP 60 Echo request
9... 6.955... 192.168.127.177 192.168.110.73 GTP 54 Echo response
9... 7.055... 192.168.110.73 192.168.127.1... GTP 60 Echo request
9... 7.155... 192.168.110.73 192.168.127.1... GTP 60 Echo request
9... 7.255... 192.168.110.73 192.168.127.1... GTP 60 Echo request

30 packets transmitted, 4 received, 86% packet loss, time 3000ms
0 out of order, 0 dups, 0 connection refused, 0 ICMP error
rtt min/avg/max/mdev = 0.561/0.657/0.798/0.096 ms

```

- Tester can check the logs of the message received by DUT using following command
 - `cat /var/log/kern.log`

```

PROTOUDP SPT=49702 DPT=2152 LEN=20
Jun 27 03:08:30 pratik-Super-Server kernel: [561701.041961] IN=eth0 OUT= MAC=bc:c4:7a:e7:e8:2d:ec:9b:bb:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52781 DF
PROTOUDP SPT=39509 DPT=2152 LEN=20
Jun 27 03:08:31 pratik-Super-Server kernel: [561702.042044] IN=eth0 OUT= MAC=bc:c4:7a:e7:e8:2d:ec:9b:bb:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52782 DF
PROTOUDP SPT=39509 DPT=2152 LEN=20
Jun 27 03:08:32 pratik-Super-Server kernel: [561703.041980] IN=eth0 OUT= MAC=bc:c4:7a:e7:e8:2d:ec:9b:bb:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52783 DF
PROTOUDP SPT=39509 DPT=2152 LEN=20
Jun 27 03:08:33 pratik-Super-Server kernel: [561704.041958] IN=eth0 OUT= MAC=bc:c4:7a:e7:e8:2d:ec:9b:bb:66:fa:f8:08:00 SRC=192.168.110.73 DST=192.168.127.177 LEN=40 TOS=0x00 PREC=0x00 TTL=61 ID=52784 DF

```

d) Test Observations

- **Case 1:** GTP Echo Replies are received by the tester and no packet loss occurs
 - **Case 2:** GTP Echo Replies are not received by the tester but GTP Echo requests are received by the DUT. Packet loss is 100%
 - **Case 3:** GTP Echo Replies are not received by the tester but GTP Echo requests are received by the DUT. Packet loss is 100%
 - **Case 4:** GTP Echo Replies are received by the tester but some packet loss is observed
- e) **Evidence Provided:-** Screenshots of Wireshark, Terminal and pcap files

12) Test Case Result:

SL. No	Test Case Name	PASS/FAIL	Remarks
1	TC1_GTP_U_FOR_IPTABLES		

(Tester has to execute the test in similar fashion for other tools)

2.8.1 TSTP For Network Level and application-level DDoS under 5G ITSAR

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.8.1
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 8 – Attack Prevention Mechanisms
2. **<Security Requirement No & Name >** 2.8.1 Network Level and application-level DDoS
3. **<Requirement Description: >** SMF shall have protection mechanism against Network level and Application-level DDoS attacks. SMF shall provide security measures to deal with overload situations which may occur as a result of a denial-of-service attack or during periods of increased traffic. In particular, partial or complete impairment of system availability shall be avoided.

For example, potential protective measures may include:

- Restricting of available RAM per application
- Restricting of maximum sessions for a Web application
- Defining the maximum size of a dataset
- Restricting CPU resources per process
- Prioritizing processes
- Limiting of amount or size of transactions of an user or from an IP address in a specific time range
- Limiting of amount or size of transactions to an IP address/Port Address in a specific time range

[Reference: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.3.1]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details

- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: There can be many kinds of overload scenarios, this TSTP is for CPU and Memory overload.

To monitor the system's CPU and memory usage during DDOS Attack,

Check if 'htop' is installed, htop is an interactive process viewer that provides real-time monitoring of CPU, memory, and process activity.

Command Used: htop -version

```
amf@localhost $ htop --version
htop 2.2.0 - (C) 2004-2019 Hisham Muhammad
Released under the GNU GPL.
```

6. Preconditions:

- A document which provides a detailed technical description of the overload control mechanisms.
- Test results from a test execution phase of overload control mechanism testing.

7. Test Objective/ Purpose: Verify that the network product:

- has a detailed technical description of the overload control mechanisms used to deal with overload scenarios;
- has test results verifying the operation of the overload control mechanisms.

8. Test Plan:

8.1 Number of Test Scenarios:

8.1.1 Test Scenario for Excessive Overload Protection due to Network Level DDOS Attack:

This test scenario stimulates Network Level DDOS attack due to which CPU and memory overload is caused, it then verifies whether the network product reacts in a controlled way during it or not. (Additional Test scenarios based on the OEM document)

8.2 TestBed Diagram:



8.3 Tools required: Command Line Interface of the DUT, hping3.

8.4 Test Execution Step:

The tester verifies that there is:

- A technical description providing a high-level overview of the overload control design.
- An overview of the types of overload scenarios that the network product overload control mechanisms are expected to handle.
- An overview of the overload control thresholds that the network product uses to trigger overload control mechanisms.
- Description of the types of attacks that may cause an overload to the network product and how these are handled.
- A description of how the network product discards or handles input during various overload situations including excessive overloads. i.e. where the overload is significantly greater than the thresholds where overload detection is triggered.
- A description of how the network product security functions operate and perform during overload.
- A description of how the network product shuts down or performs or takes other abatement or corrective actions during excessive overload conditions.

The tester performs below steps:

- Introduce a controlled DDOS scenario with predefined levels of load and traffic.
- Observe the Network Product's response and measure how it adjusts its resource allocation and prioritization to handle the increased load.
- Validate that the NF does not crash or suffer from critical failures due to the attack.
- Assess if the NF can dynamically adapt its behavior based on the severity of the attack.
- Verify that the NF can smoothly recover and return to normal operation after the attack is removed.

The tester verifies that the test results:

- Contain details of the overload conditions used in the test execution that are consistent with the technical description document.
- Describe test procedures used to verify the overload control mechanisms.
- Contain data which demonstrates/indicates that the overload control mechanisms described in the technical description document have been implemented.
- Contain details of the test set-up including the mechanisms for creating the overload. Where simulators and/or scripts are used to artificially create a load then details of these should also be included.

9. Expected Results:

- A technical description provides a high-level overview of the overload control design.
- An overview of the types of overload scenarios and overload control thresholds that are considered.

- Description on the types of attacks that may cause an overload to the system and how these are handled.
- A description of how the network product discards or handles input during various overload situations.
- Describes if or how the network product security functions operate and perform during overload.
- If parts of the system shutdown or take other abatement or corrective actions these should be described.

Note: If some of the items listed above are not applicable to a network product then, in those cases, it should be clarified by the vendor why these items are not applicable.

The test results should:

- Contain details of the overload conditions used in the test execution that are consistent with the technical description document.
- Describe the test procedures used to verify the overload control mechanisms.
- Contain data which demonstrates/indicates that the overload control mechanisms described in the technical description document have been implemented.
- Contain details of the test set-up including the mechanisms for creating the overload.

10. **Expected Format of Evidence:** Documentation showing each of the points in the results sections.

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC1_NETWORK_LEVEL_DDOS

b. **Test Case Description:** Test needs to be conducted to verify that the Network Product can react to the Network Level DDOS in a controlled manner.

c. **Execution Steps:**

- Create an overload scenario, in this example we'll use '**hping3**' to simulate high CPU and memory usage.

Run command:

sudo hping3 -c 15000 -d 120 -S -w 64 -p <port-number> --flood --rand-source <ip-address>

'hping3' will flood ping attack to the specified IP address.

This command will generate a significant load on both the CPU and memory.

```
amf@localhost $ sudo hping3 -c 15000 -d 120 -S -w 64 -p 4567 --flood --rand-source 192.168.127.177
HPING 192.168.127.177 (eth0 192.168.127.177): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.127.177 hping statistic ---
17754554 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

- During the overload simulation, we'll use 'htop' to monitor the system's CPU and memory usage in real-time.

Open a new terminal and run command: **htop**

While 'hping3' is running, 'htop' will display the CPU and memory usage of the system. Observe how the CPU usage increases due to the workload.

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
3861546	root	20	0	43140	36384	18092	S	53.5	0.1	0:00.82	python3 /snap/microstack
4061847	root	20	0	43088	36552	10740	S	51.5	0.1	0:00.79	python3 /snap/microstack
3788550	priyansha	0	-20	251M	45448	20320	S	32.0	0.1	1:34.08	/usr/NX/bin/nxcodec.bin
3999712	priyansha	20	0	379M	103M	69912	S	7.8	0.3	14:59.29	/usr/lib/xorg/Xorg vt2 -
1142	root	20	0	2010M	40116	8740	S	7.2	0.1	3h59:58	/usr/lib/snapd/snapd
4001542	priyansha	20	0	13.6G	526M	79092	S	5.2	1.7	1h13:05	/usr/share/teams/teams -
1561974	root	16	4	11364	1520	1320	R	3.9	0.0	15:02.63	/sbin/auditd
4000859	priyansha	0	-20	3659M	211M	115M	S	3.3	0.7	19:17.74	/usr/NX/bin/nxnode.bin
4061836	priyansha	0	-20	251M	45448	20320	S	3.3	0.1	0:00.23	/usr/NX/bin/nxcodec.bin

In 'htop', look for the 'CPU%' and 'MEM%' columns to see the CPU and memory usage, respectively, of different processes.

- Monitor the network product's performance, response times, and behavior during the overload simulation.
- Validate that the network product reacts to the overload in a controlled manner and prioritizes critical tasks. (The expected behavior of the network product shall be provided in the OEM Documentation)
- Observe how the network product's security functions operate and perform during the overload.

d. Test Observation:

- **Case 1:** The network product reacts in a controlled way during overload (**Positive Testcase**)

The network product demonstrates a controlled response to the simulated DDOS Attack. It prioritizes critical tasks, maintains essential functions, and keeps security functions operational during the overload scenario. Then the test case Passes.

- **Case 2:** The network product reacts in a uncontrolled way during overload (**Negative TestCase**)

If the network product exhibits unpredictable behavior, crashes, or fails to maintain its security functions during DDOS, it will be considered a failure.

e. Evidence Provided: Documentation showing each of the points in the results sections.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_NETWORK_LEVEL_DDOS		

2.8.2 TSTP For Excessive Overload Protection under 5G ITSAR

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.8.2
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 8 – Attack Prevention Mechanisms
2. **<Security Requirement No & Name >** 2.8.2 Excessive Overload Protection
3. **<Requirement Description: >** SMF shall act in a predictable way if an overload situation cannot be prevented. SMF shall be built in this way that it can react on an overload situation in a controlled way. However, it is possible that a situation happens where the security measures are no longer sufficient. In such case it shall be ensured that SMF cannot reach an undefined and thus potentially insecure, state. OEM shall provide a technical description of the SMF's Over Load Control mechanisms. (especially whether these mechanisms rely on cooperation of other network elements e.g. RAN)

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```


Note: There can be many kinds of overload scenarios, this TSTP is for CPU and Memory overload.

To monitor the system's CPU and memory usage during overload,

Check if 'htop' is installed, htop is an interactive process viewer that provides real-time monitoring of CPU, memory, and process activity.

Command Used: htop -version

```
amf@localhost $ htop --version
htop 2.2.0 - (C) 2004-2019 Hisham Muhammad
Released under the GNU GPL.
```

6. **Preconditions:**

- A document which provides a detailed technical description of the overload control mechanisms.
- Test results from a test execution phase of overload control mechanism testing.

7. **Test Objective/ Purpose:** Verify that the network product:

- has a detailed technical description of the overload control mechanisms used to deal with overload scenarios;
- has test results verifying the operation of the overload control mechanisms.

8. **Test Plan:**

8.1 **Number of Test Scenarios:**

8.1.1 **Test Scenario for Excessive Overload Protection due to CPU Overload:**

This test scenario stimulates overload conditions in which CPU overload is performed and verifies whether the network product reacts in a controlled way during it.
(Additional Test scenarios based on the OEM document)

Securing Networks

8.2 **TestBed Diagram:**



8.3 Tools required: Command Line Interface of the DUT and other tools for overload simulation based on documentation.

8.4 Test Execution Step: The tester verifies that there is:

- A technical description providing a high-level overview of the overload control design.
- An overview of the types of overload scenarios that the network product overload control mechanisms are expected to handle.
- An overview of the overload control thresholds that the network product uses to trigger overload control mechanisms.
- Description of the types of attacks that may cause an overload to the network product and how these are handled.
- A description of how the network product discards or handles input during various overload situations including excessive overloads. i.e. where the overload is significantly greater than the thresholds where overload detection is triggered.
- A description of how the network product security functions operate and perform during overload.
- A description of how the network product shuts down or performs or takes other abatement or corrective actions during excessive overload conditions.

The tester performs below steps:

- Introduce a controlled overload scenario with predefined levels of load and traffic.
- Observe the Network Product's response and measure how it adjusts its resource allocation and prioritization to handle the increased load.
- Validate that the NF does not crash or suffer from critical failures due to the controlled overload.
- Assess if the NF can dynamically adapt its behavior based on the severity of the overload situation.
- Verify that the NF can smoothly recover and return to normal operation after the controlled overload is removed.

The tester verifies that the test results:

- Contain details of the overload conditions used in the test execution that are consistent with the technical description document.
- Describe test procedures used to verify the overload control mechanisms.
- Contain data which demonstrates/indicates that the overload control mechanisms described in the technical description document have been implemented.
- Contain details of the test set-up including the mechanisms for creating the overload. Where simulators and/or scripts are used to artificially create a load then details of these should also be included.

9. Expected Results:

- A technical description provides a high-level overview of the overload control design.
- An overview of the types of overload scenarios and overload control thresholds that are considered.
- Description on the types of attacks that may cause an overload to the system and how these are handled.
- A description of how the network product discards or handles input during various overload situations.
- Describes if or how the network product security functions operate and perform during overload.
- If parts of the system shutdown or take other abatement or corrective actions these should be described.

Note: If some of the items listed above are not applicable to a network product then, in those cases, it should be clarified by the vendor why these items are not applicable.

The test results should:

- Contain details of the overload conditions used in the test execution that are consistent with the technical description document.
- Describe the test procedures used to verify the overload control mechanisms.
- Contain data which demonstrates/indicates that the overload control mechanisms described in the technical description document have been implemented.
- Contain details of the test set-up including the mechanisms for creating the overload.

10. **Expected Format of Evidence:** Documentation showing each of the points in the results sections.

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC1_EXCESSIVE_OVERLOAD_PROTECTION

b. **Test Case Description:** Test needs to be conducted to verify that the Network Product can react to an overload situation in a controlled manner.

c. **Execution Steps:**

- Create an overload scenario, in this example we'll use 'hping3' to simulate high CPU and memory usage.

Run command:

sudo hping3 -c 15000 -d 120 -S -w 64 -p <port-number> --flood --rand-source <ip-address>

'hping3' will flood ping attack to the specified IP address.

This command will generate a significant load on both the CPU and memory.

```
amf@localhost $ sudo hping3 -c 15000 -d 120 -S -w 64 -p 4567 --flood --rand-source 192.168.127.177
HPING 192.168.127.177 (eth0 192.168.127.177): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.127.177 hping statistic ---
17754554 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

- During the overload simulation, we'll use 'htop' to monitor the system's CPU and memory usage in real-time.

Open a new terminal and run command: **htop**

While 'hping3' is running, 'htop' will display the CPU and memory usage of the system. Observe how the CPU usage increases due to the workload.

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
1061846	root	20	0	43140	36584	18092	S	53.5	0.1	0:00.82	python3 /snap/microstack
4061847	root	20	0	43088	36552	10740	S	51.5	0.1	0:00.79	python3 /snap/microstack
3788550	priyansha	0	-20	251M	45448	20320	S	32.0	0.1	1:34.08	/usr/NX/bin/nxcodec.bin
3999712	priyansha	20	0	379M	103M	69912	S	7.8	0.3	14:59.29	/usr/lib/xorg/Xorg vt2 -
1142	root	20	0	2010M	40116	8740	S	7.2	0.1	1h59:58	/usr/lib/snapd/snapd
4001542	priyansha	20	0	13.6G	526M	79092	S	5.2	1.7	1h13:05	/usr/share/teams/teams -
1561974	root	16	-4	11364	1520	1320	R	3.9	0.0	15:02.63	/sbin/auditd
4000859	priyansha	0	-20	3659M	211M	115M	S	3.3	0.7	19:17.74	/usr/NX/bin/nxnode.bin
4061836	priyansha	0	-20	251M	45448	20320	S	3.3	0.1	0:00.23	/usr/NX/bin/nxcodec.bin

In 'htop', look for the 'CPU%' and 'MEM%' columns to see the CPU and memory usage, respectively, of different processes.

- Monitor the network product's performance, response times, and behavior during the overload simulation.
- Validate that the network product reacts to the overload in a controlled manner and prioritizes critical tasks. (The expected behavior of the network product shall be provided in the OEM Documentation)
- Observe how the network product's security functions operate and perform during the overload.

d. Test Observation:

- **Case 1:** The network product reacts in a controlled way during overload (**Positive Testcase**)

The network product demonstrates a controlled response to the simulated overload. It prioritizes critical tasks, maintains essential functions, and keeps security functions operational during the overload scenario. Then the test case Passes.

- **Case 2:** The network product reacts in an uncontrolled way during overload (**Negative Testcase**)

If the network product exhibits unpredictable behavior, crashes, or fails to maintain its security functions during overload, it will be considered a failure.

e. Evidence Provided: Documentation showing each of the points in the results sections.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_EXCESSIVE_OVERLOAD_PROTECTION		

2.8.3 TSTP for Evaluation of Manipulated packets that are sent to an address of the network device shall not lead to an impairment of availability

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.8.3
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 8: Attack Prevention Mechanisms
2. **<Security Requirement No & Name >** 2.8.3 Manipulated packets that are sent to an address of the network device shall not lead to an impairment of availability
3. **<Requirement Description: >** DUT shall not be affected in its availability or robustness by incoming packets from other network elements that are manipulated or differing the norm. This means that appropriate packets shall be detected as invalid and be discarded. The process shall not be affecting the performance of the SMF. This robustness shall be just as effective for a great mass of invalid packets as for individual or a small number of packets.

Examples of such packets are:

- Mass-produced TCP packets with a set SYN flag to produce half-open TCP connections (SYN flooding attack).
- Packets with the same IP sender address and IP recipient address (Land attack).
- Mass-produced ICMP packets with the broadcast address of a network as target address (Smurf attack).
- Fragmented IP packets with overlapping offset fields (Teardrop attack).
- ICMP packets that are larger than the maximum permitted size (65,535 Bytes) of IPv4 packets (Ping-of-death attack).
- Uncorrelated reply packets (i.e. packets which cannot be correlated to any request).

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.2.6.2.2]

4. DUT Confirmation Details:

- Use the command line interface to get details of the machine on which test is conducted.
- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)


```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. Preconditions

- The tester has the privileges to log in the network product and to access all system resources (e.g., log files)
- A list of all available network services containing at least the following information shall be included in the documentation accompanying the Network Product:
 - o All interfaces providing IP-based protocols;
 - o the available transport layer protocols on these interfaces;
 - o their open ports and associated services;
 - o and a free-form description of their purposes.
- **NOTE:** This list is to be validated as part of the BVT port scanning activity.
- The robustness and fuzzing tools that are selected for this test shall utilize state-of-the-art technology to identify input which causes the Network Product to behave in an unspecified, undocumented, or unexpected manner.
- Fuzz testing tools are a highly sophisticated technology and adaptation to the individual protocols in question is needed to be effective. Therefore, there is a lack of available effective fuzz testing tools available especially for protocols proprietary to the Telco industry. Test labs shall acquire fuzz testing tools for those protocols where commercially feasible, taking into account - *"To avoid creating a monopoly for security testing tool vendors the usage of a security testing tool having specific capabilities should only be mandatory if there are at least two alternatives by different vendors available for use in most world regions."*
- It needs to be taken into account that fuzz testing tools might show drastic differences in terms of effectiveness.
- The accredited test lab is expected to have sufficient expertise to recognize the level of effectiveness of the available tools.

7. Test Objective:

- To verify that the network product provides externally reachable services which are robust against unexpected input.
- The target of this test are the protocol stacks (e.g., diameter stack) rather than the applications (e.g., web app).

8. Test Plan:

8.1 Number of Test Scenarios:

- 8.1.1 Test Scenario for SYN flooding attack
- 8.1.2 Test Scenario for Land attack
- 8.1.3 Test Scenario for Smurf attack
- 8.1.4 Test Scenario for Teardrop attack
- 8.1.5 Test Scenario for Ping-of-death attack
- 8.1.6 Test Scenario for Uncorrelated reply to packets

8.2 Test Setup Diagram:



8.3 Tools Used: Wireshark in Tester Device

8.4 Test Execution Steps The accredited evaluator's test lab is required to execute the following steps:

1. Execution of available effective fuzzing tools against the protocols available via interfaces providing IP-based protocols of the Network Product for an amount of time sufficient to be effective.
2. Execution of available effective robustness test tools against the protocols available via interfaces providing IP-based protocols of the Network Product for an amount of time sufficient to be effective.
3. For both step 1 and 2:
 - a. Using a network traffic analyser on the network product (e.g. TCPDUMP) or an external traffic analyser directly connected to the network product, the tester verifies that the packets are correctly processed by the network product.
 - b. The testers verifies that the network product and any running network service does not crash.
 - c. The execution of tests shall run sufficient times.

9. Expected Results for Pass:

- A list of all of the protocols of the network product reachable externally on an IP-based interface, together with an indication whether effective available robustness and fuzz testing

tools have been used against them, shall be part of the testing documentation. If no tool can be acquired for a protocol, a free form statement should explain why not.

- The used tool(s) name, their unambiguous version (also for plug-ins if applicable), used settings, and the relevant output is evidence and shall be part of the testing documentation.
- Any input causing unspecified, undocumented, or unexpected behaviour, and a description of this behaviour shall be highlighted in the testing documentation.
- COTS fuzzing tools, by their nature, may have an acceptable failure rate (e.g. 0.1%) due to different non-deterministic variables in their implementation. At some point the tool's documentation may even mention that the failing test shall be repeated to check whether it is really a recurring problem or not. The tester shall make best effort to determine if there is an issue with NE or the test tool and if necessary, work with the vendor of the network product to come to a consensus on the test result outcome.

10. **Expected Format of Evidence:** A testing report provided by the testing agency which will consist of the following information:

- The used tool(s) name and version information,
- Settings and configurations used
- The output log file of the chosen tool that displays the results (passed/failed).
- Screenshot
- Test result (Passed or not)
- Log/evidence tracing possible crashes
- Any input causing unspecified, undocumented, or unexpected behavior

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC_SYN_FLOOD_ATTACK

b. **Test Case Description:** Tests need to be conducted to check the effects of SYN Flood attack on the running setup

c. **Execution Steps:**

1. Note the system resources (RAM and CPU usage) of the DUT
2. The tester configures the tool to send a huge amount of TCP Syn packets against the Network Product (*e.g. here we have simulated the attack using linux command `hping3 -i <waiting time between each packet> -S -p <TCP port> -c <Number of packets> <DUT IP>`*)
3. Verify if there is unexpected **increased CPU usage** and **memory consumption**.

d. **Test Observations:**

- Case 1: Check whether Network Product experiences unexpected hoarding of resources in the presence of SYN Flood attack.

Use command **cat /proc/meminfo** to know the resources of the DUT

```

MemTotal:      8055556 kB
MemFree:       321696 kB
MemAvailable:  2791068 kB
Buffers:       448412 kB
Cached:        2881632 kB
SwapCached:    2216 kB
Active:        2090560 kB
Inactive:      4774616 kB
Active(anon):   81780 kB
Inactive(anon): 4237096 kB
Active(file):   2008780 kB
Inactive(file): 537520 kB
Unevictable:    253412 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB

```

In the testcase, we should get an external entity connected to the network product which produces huge amounts of SYN traffic in order to cause DOS at the intended Network product.

```

zmqo pbfud3 -c 12000 -q 150 -2 -M 04 -b 80 --Ljooq --L9Uq-2onLC6 10.01.3.00

```

Figure 2: Above figure shows the command that is used to flood the network product with SYN packets.

Usage of the above command should have no unexpected effect on the operation of the network product

```

MemTotal:      8055556 kB
MemFree:       243252 kB
MemAvailable:  2508652 kB
Buffers:       409428 kB
Cached:        2647128 kB
SwapCached:    480 kB
Active:        1925888 kB
Inactive:      5156488 kB
Active(anon):   90536 kB
Inactive(anon): 4636152 kB
Active(file):   1835352 kB
Inactive(file): 520336 kB
Unevictable:    111576 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB

```

1234	63.359433744	10.61.3.97	10.61.3.66	176	TCP	1458	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1377	64.347910921	10.61.3.97	10.61.3.66	176	TCP	1459	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1390	65.348088869	10.61.3.97	10.61.3.66	176	TCP	1460	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1411	66.348305233	10.61.3.97	10.61.3.66	176	TCP	1461	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1426	67.348522420	10.61.3.97	10.61.3.66	176	TCP	1462	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1441	68.348606885	10.61.3.97	10.61.3.66	176	TCP	1463	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1466	69.348761465	10.61.3.97	10.61.3.66	176	TCP	1464	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1478	70.348975229	10.61.3.97	10.61.3.66	176	TCP	1465	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1489	71.349547120	10.61.3.97	10.61.3.66	176	TCP	1466	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1504	72.350213079	10.61.3.97	10.61.3.66	176	TCP	1467	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1519	73.350498314	10.61.3.97	10.61.3.66	176	TCP	1468	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1515	74.351132988	10.61.3.97	10.61.3.66	176	TCP	1469	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1521	75.351244990	10.61.3.97	10.61.3.66	176	TCP	1470	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.
1529	76.351430325	10.61.3.97	10.61.3.66	176	TCP	1471	→ 80	[SYN]	Seq=0	Win=64	Len=120	[TCP segment of a reasse.

QAbove Wireshark trace shows that numerous TCP SYN messages were sent from 10.61.3.97 (Tester) to DUT (10.61.3.66).

SYN cookies are an innovative method of thwarting SYN flood attacks. Rather than allocating memory for each connection attempt, SYN cookies allocate a sequence number to track the connection. That sequence number is returned in the TCP SYN-ACK response to the client, and the server uses it to validate subsequent ACKs from that client. IN linux syn cookies are enabled by default to prevent Syn Flood Attacks. The “net.ipv4.tcp_syncookies = 1” entry in the “/etc/sysctl.conf” file is used to enable syn cookies. If it is set to 0, it means syn cookies are disabled.

➤ **Test Case Number:** 02

a. **Test Case Name:** TC_LAND_ATTACK

b. **Test Case Description:** Tests need to be conducted to check the effects of Land attack on the running setup

c. **Execution Steps:**

1. Note the system resources (RAM and CPU usage) of the DUT
2. The tester configures the tool to send a huge amount of IP packets with source and Destination IP as same.
3. Verify if there is unexpected **increased CPU usage** and **memory consumption**.

d. **Test Observations:**

- **Case 1:** Check whether Network Product experiences unexpected hoarding of resources in the presence of Land Attack Packets

Use command **cat /proc/meminfo** to know the resources of the DUT

```
MemTotal:      8055556 kB
MemFree:       321696 kB
MemAvailable:  2791068 kB
Buffers:       448412 kB
Cached:        2881632 kB
SwapCached:    2216 kB
Active:        2090560 kB
Inactive:      4774616 kB
Active(anon):  81780 kB
Inactive(anon): 4237096 kB
Active(file):  2008780 kB
Inactive(file): 537520 kB
Unevictable:   253412 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB
```

In the testcase, we should get an external entity connected to the network product which produces huge amounts of Land Attack packets in order to cause DOS at the intended Network product. The given hping3 command sends 100 SYN packets (TCP with SYN flag set) to destination IP port 139, with a payload size of 40 bytes, and uses source IP "10.61.3.6" and source port 80, while keeping the connection open.

```
hping -V -c 100 -d 40 -S -p 139 -s 80 -k -a 10.61.3.66
```

Figure 2: Above figure shows the command that is used to flood the network product with same src and dst IP.

Usage of the above command should have no unexpected effect on the operation of the network product

```
MemTotal:      8055556 kB
MemFree:       243252 kB
MemAvailable:  2508652 kB
Buffers:       409428 kB
Cached:        2647128 kB
SwapCached:    480 kB
Active:        1925888 kB
Inactive:      5156488 kB
Active(anon):   90536 kB
Inactive(anon): 4636152 kB
Active(file):   1835352 kB
Inactive(file): 520336 kB
Unevictable:    111576 kB
Mlocked:        32 kB
SwapTotal:     1158220 kB
```

Modern Machines or enterprise level firewalls drop LAND Packets at ISP level itself because of their suspicious nature.

➤ **Test Case Number: 03**

a. **Test Case Name:** TC_SMURF_ATTACK

b. **Test Case Description:** Tests need to be conducted to check the effects of smurf attack on the running setup

c. **Execution Steps:**

1. Note the system resources (RAM and CPU usage) of the DUT
2. The tester configures the tool to send a huge amount of smurf attack packets.
4. Verify if there is unexpected **increased CPU usage** and **memory consumption**.

d. **Test Observations:**

- **Case 1:** Check whether Network Product experiences unexpected hoarding of resources in the presence of smurf attack packets

Use command **cat /proc/meminfo** to know the resources of the DUT


```

MemTotal:      8055556 kB
MemFree:       321696 kB
MemAvailable:  2791068 kB
Buffers:       448412 kB
Cached:        2881632 kB
SwapCached:    2216 kB
Active:        2090560 kB
Inactive:      4774616 kB
Active(anon):   81780 kB
Inactive(anon): 4237096 kB
Active(file):   2008780 kB
Inactive(file): 537520 kB
Unevictable:   253412 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB

```

In the testcase, we should get an external entity connected to the network product which produces huge amounts of smurf attack packets in order to cause DOS at the intended Network product.

The given hping3 command performs an ICMP (Internet Control Message Protocol) echo request packet (ping) with a count of 1 packet. It sends the packet from the source IP "10.61.3.97" to the destination IP "10.61.3.66." Additionally, the -spooft option is included, which allows the packet to be spoofed, meaning the source IP address in the packet can be set to an arbitrary value.

```
hping3 -icmp -c 1 spoof 10.61.3.97 10.61.3.66
```

Figure 2: Above figure shows the command that is used to flood the network product with smurf attack packets.

Usage of the above command should have no unexpected effect on the operation of the network product

```

MemTotal:      8055556 kB
MemFree:       243252 kB
MemAvailable:  2508652 kB
Buffers:       409428 kB
Cached:        2647128 kB
SwapCached:    480 kB
Active:        1925888 kB
Inactive:      5156488 kB
Active(anon):   90536 kB
Inactive(anon): 4636152 kB
Active(file):   1835352 kB
Inactive(file): 520336 kB
Unevictable:   111576 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB

```

- *Smurf Attack Prevention Mechanisms:*
- *Disabling IP-directed broadcasts on all network routers. This stops attackers from using it to amplify their attacks.*
- *Configuring network devices to limit or disallow ICMP traffic in general.*

- Reconfiguring your firewall to disallow pings that do not originate from your network.
- Using anti-malware and intrusion detection software.

➤ **Test Case Number: 04**

a. **Test Case Name:** TC_TEARDROP_ATTACK

b. **Test Case Description:** Tests need to be conducted to check the effects of Teardrop attack on the running setup

c. **Execution Steps:**

1. Note the system resources (RAM and CPU usage) of the DUT
2. The tester configures the tool to send a huge amount of Teardrop attack packet with overlapping fragment offset
3. Verify if there is unexpected **increased CPU usage** and **memory consumption, because DUT is unable to reassemble and keeps on expediting resources for it.**

d. **Test Observations:**

- Case 1: Check whether Network Product experiences unexpected hoarding of resources in the presence of Teardrop Packets.

Use command **cat /proc/meminfo** to know the resources of the DUT

```
MemTotal:      8055556 kB
MemFree:       321696 kB
MemAvailable:  2791068 kB
Buffers:       448412 kB
Cached:        2881632 kB
SwapCached:    2216 kB
Active:        2090560 kB
Inactive:      4774616 kB
Active(anon):   81780 kB
Inactive(anon): 4237096 kB
Active(file):   2008780 kB
Inactive(file): 537520 kB
Unevictable:    253412 kB
Mlocked:        32 kB
SwapTotal:     1158220 kB
```

In the testcase, we should get an external entity connected to the network product which produces huge amounts of teardrop packets in order to cause DOS at the intended Network product.

Sample python script to generate teardrop packets is attached, but it is only for simulation is not an effective testing tool.

Usage of the above command should have no unexpected effect on the operation of the network product

```

MemTotal:      8055556 kB
MemFree:       243252 kB
MemAvailable:  2508652 kB
Buffers:       409428 kB
Cached:        2647128 kB
SwapCached:    480 kB
Active:        1925888 kB
Inactive:      5156488 kB
Active(anon):   90536 kB
Inactive(anon): 4636152 kB
Active(file):   1835352 kB
Inactive(file): 520336 kB
Unevictable:   111576 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB

```

Modern systems, including Ubuntu, have implemented various security measures to mitigate attacks like the Teardrop Attack. While specifics may vary based on the version of Ubuntu and the underlying kernel, here are some general ways in which modern systems mitigate this type of attack:

- i. **IP Packet Reassembly Fixes:** Modern kernels have implemented fixes to properly handle fragmented IP packets, preventing them from causing crashes or vulnerabilities. These fixes ensure that the reassembly of fragmented packets is done correctly, reducing the risk of attacks like Teardrop.
- ii. **IP Defragmentation:** The Linux kernel has mechanisms to correctly defragment incoming IP packets before processing. This helps prevent the reassembly of maliciously crafted fragmented packets

➤ **Test Case Number:** 05

a. **Test Case Name:** TC_PING_OF_DEATH_ATTACK

b. **Test Case Description:** Tests need to be conducted to check the effects of Ping-of-Death Packets attack on the running setup

c. **Execution Steps:**

1. Note the system resources (RAM and CPU usage) of the DUT
2. The tester configures the tool to send a huge amount of Ping-of-Death Packets
3. Verify if there is unexpected **increased CPU usage** and **memory consumption**.

d. **Test Observations:**

- **Case 1:** Check whether Network Product experiences unexpected hoarding of resources in the presence of Ping-of-Death Packets.

Use command **cat /proc/meminfo** to know the resources of the DUT

```

MemTotal:      8055556 kB
MemFree:       321696 kB
MemAvailable:  2791068 kB
Buffers:       448412 kB
Cached:        2881632 kB
SwapCached:    2216 kB
Active:        2090560 kB
Inactive:      4774616 kB
Active(anon):   81780 kB
Inactive(anon): 4237096 kB
Active(file):   2008780 kB
Inactive(file): 537520 kB
Unevictable:   253412 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB

```

In the testcase, we should get an external entity connected to the network product which produces huge amounts of Ping-of-Death Packets in order to cause DOS at the intended Network product.

ping <ip address> -s 65500 -t <TTL> -n <No. Of Echo Requests>

```
ping 10.61.3.66 -s 65500 -t 1 -n 1
```

Figure 2: Above figure shows the command that is used to flood the network product with Ping-of-Death Packets.

Usage of the above command should have no unexpected effect on the operation of the network product

```

MemTotal:      8055556 kB
MemFree:       243252 kB
MemAvailable:  2508652 kB
Buffers:       409428 kB
Cached:        2647128 kB
SwapCached:    480 kB
Active:        1925888 kB
Inactive:      5156488 kB
Active(anon):   90536 kB
Inactive(anon): 4636152 kB
Active(file):   1835352 kB
Inactive(file): 520336 kB
Unevictable:   111576 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB

```

Linux has a flag called ICMP rate limit which by default is set. This limits the number of packets to a safe number to prevent flooding attacks. The “n net.ipv4.icmp_ratelimit” entry in the “/etc/sysctl.conf” file is used to enable syn cookies. If it is set to 0, it rate limiting is disabled and flooding can be done.

Test Case Number: 06

a. **Test Case Name:** TC_UNCORRELATED_REPLY

b. **Test Case Description:** Tests need to be conducted to check the effects of Uncorrelated Reply attack on the running setup

c. **Execution Steps:**

1. Note the system resources (RAM and CPU usage) of the DUT

2. The tester configures the tool to send a huge amount of Uncorrelated Reply Packets.

5. Verify if there is unexpected **increased CPU usage** and **memory consumption**.

d. **Test Observations:**

- Case 1: Check whether Network Product experiences unexpected hoarding of resources in the presence of Uncorrelated Reply Packets.

Use command **cat /proc/meminfo** to know the resources of the DUT

```
MemTotal:      8055556 kB
MemFree:       321696 kB
MemAvailable:  2791068 kB
Buffers:       448412 kB
Cached:        2881632 kB
SwapCached:    2216 kB
Active:        2090560 kB
Inactive:      4774616 kB
Active(anon):   81780 kB
Inactive(anon): 4237096 kB
Active(file):   2008780 kB
Inactive(file): 537520 kB
Unevictable:    253412 kB
Mlocked:        32 kB
SwapTotal:     1158220 kB
```

In the testcase, we should get an external entity connected to the network product which produces huge amounts of Uncorrelated Reply Packets. in order to cause DOS at the intended Network product.

sudo hping3 <destination_IP> -c <number_of_packets> -K <response_delay> -S

```
sudo hping3 10.61.3.66 -c 1000000 -K 10000 -S
```

Figure 2: Above figure shows the command that is used to flood the network product with Uncorrelated Reply Packets.

Usage of the above command should have no unexpected effect on the operation of the network product

```

MemTotal:      8055556 kB
MemFree:       243252 kB
MemAvailable:  2508652 kB
Buffers:       409428 kB
Cached:        2647128 kB
SwapCached:    480 kB
Active:        1925888 kB
Inactive:      5156488 kB
Active(anon):   90536 kB
Inactive(anon): 4636152 kB
Active(file):   1835352 kB
Inactive(file): 520336 kB
Unevictable:   111576 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB

```

There is no generic mechanism for all types of packets to prevent this attack, in Linux. Given below is a mechanism which can be found in the `/etc/sysctl.conf` file. You can add or modify specific ICMP-related settings in this file. For example, you might want to adjust the following settings:

- *`net.ipv4.icmp_echo_ignore_all`: Set to 1 to ignore all ICMP echo requests (ping requests).*
- *`net.ipv4.icmp_echo_ignore_broadcasts`: Set to 1 to ignore ICMP echo requests sent to broadcast addresses.*

12. **Test** Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_SYN_FLOOD		
2	TC_LAND_ATTACK		
3	TC_SMURF_ATTACK		
4	TC_TEARDROP_ATTACK		
5	TC_PING_OF_DEATH_ATTACK		
6	TC_UNCORRELATED_REPLY		

2.9.1 TSTP for Evaluation of Fuzzing – Network and Application Level

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.9.1
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 9: Vulnerability Testing Requirements
2. **<Security Requirement No & Name >** 2.9.1 Fuzzing – Network and Application Level
3. **<Requirement Description:>**

It shall be ensured that externally reachable services of SMF are reasonably robust when receiving unexpected input.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.4.4]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: **cat /etc/os-release** (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5. DUT Configuration

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions**

- The tester has the privileges to log in the network product and to access all system resources (e.g. log files)
- A list of all available network services containing at least the following information shall be included in the documentation accompanying the Network Product:
 - all interfaces providing IP-based protocols;
 - the available transport layer protocols on these interfaces;
 - their open ports and associated services;
 - and a free-form description of their purposes.
- **NOTE:** This list is to be validated as part of the BVT port scanning activity.
- The robustness and fuzzing tools that are selected for this test shall utilize state-of-the-art technology to identify input which causes the Network Product to behave in an unspecified, undocumented, or unexpected manner.
- Fuzz testing tools are a highly sophisticated technology and adaptation to the individual protocols in question is needed to be effective. Therefore, there is a lack of available effective fuzz testing tools available especially for protocols proprietary to the Telco industry. Taking into account note 4 of TR 33.916's clause 7.2.4, test labs shall acquire fuzz testing tools for those protocols where commercially feasible.
- It needs to be taken into account that fuzz testing tools might show drastic differences in terms of effectiveness.
- The accredited test lab is expected to have sufficient expertise to recognize the level of effectiveness of the available tools.

7. **Test Objective:-** To verify that the network product provides externally reachable services which are robust against unexpected input.

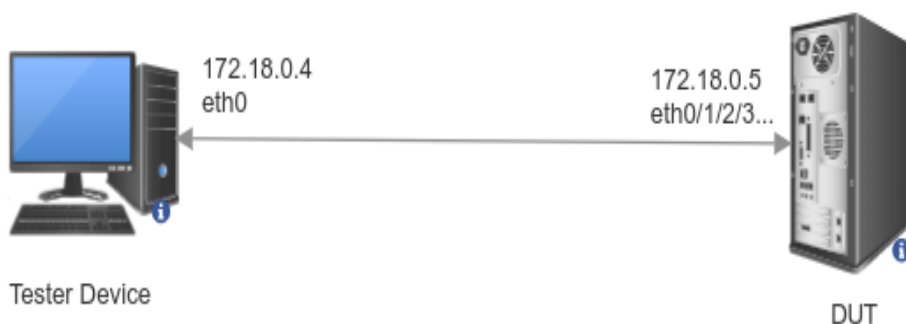
8. **Test Plan**

8.1. **Number of Test Scenarios**

8.1.1. Test Scenario for Fuzzing for Application Layer

8.1.2. Test Scenario for Fuzzing for Network Layer

8.2. **Test Bed Diagram**



8.3. **Tools Required:-** Fuzzing Tool (Defenses by Synopsys was used in our case)

8.4. **Test Execution Steps:-** The accredited evaluator's test lab is required to execute the following steps:

1. Execution of available effective fuzzing tools against the protocols available via interfaces providing Application-based protocols of the Network Product for an amount of time sufficient to be effective.
2. Execution of available effective robustness test tools against the protocols available via interfaces providing IP-based protocols of the Network Product for an amount of time sufficient to be effective. (Use of TCP/UDP Packets with underlying IP packets also works)
3. For both step 1 and 2:
 - a. Using a network traffic analyser on the network product (e.g. TCPDUMP/Wireshark) or an external traffic analyser directly connected to the network product, the tester verifies that the packets are correctly processed by the network product.
 - b. The testers verifies that the network product and any running network service does not crash.
 - c. The execution of tests shall run sufficient times.

Securing Networks

9. **Expected Results for Pass**

- A list of all of the protocols of the network product reachable externally on an IP-based interface, together with an indication whether effective available robustness and fuzz testing tools have been used against them, shall be part of the testing documentation. If no tool can be acquired for a protocol, a free form statement should explain why not.
- The used tool(s) name, their unambiguous version (also for plug-ins if applicable), used settings, and the relevant output is evidence and shall be part of the testing documentation.
- Any input causing unspecified, undocumented, or unexpected behaviour, and a description of this behaviour shall be highlighted in the testing documentation.
- COTS fuzzing tools, by their nature, may have an acceptable failure rate (e.g. 0.1%) due to different non-deterministic variables in their implementation. At some point the tool's documentation may even mention that the failing test shall be repeated to check whether it is really a recurring

problem or not. The tester shall make best effort to determine if there is an issue with NE or the test tool and if necessary, work with the vendor of the network product to come to a consensus on the test result outcome.

10. **Expected Form of Evidence:-** A testing report provided by the testing agency which will consist of the following information:

- The used tool(s) name and version information,
- Settings and configurations used
- The output log file of the chosen tool that displays the results (passed/failed).
- Screenshot
- Test result (Passed or not)
- Log/evidence tracing possible crashes
- Any input causing unspecified, undocumented, or unexpected behaviour

11. **Test Execution**

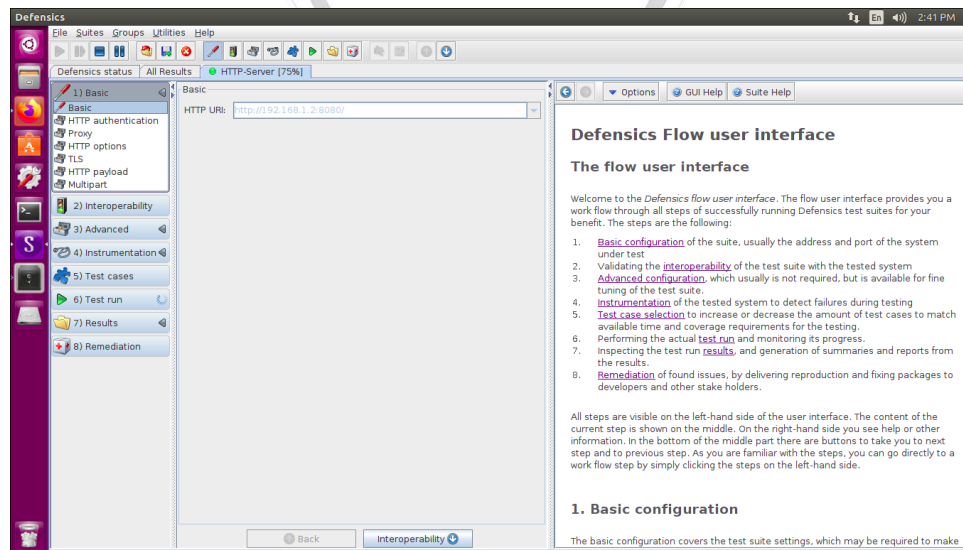
➤ **Test Case Number: 01**

a. **Test Case Name:** TC_FUZZ_TESTING_APPLICATION

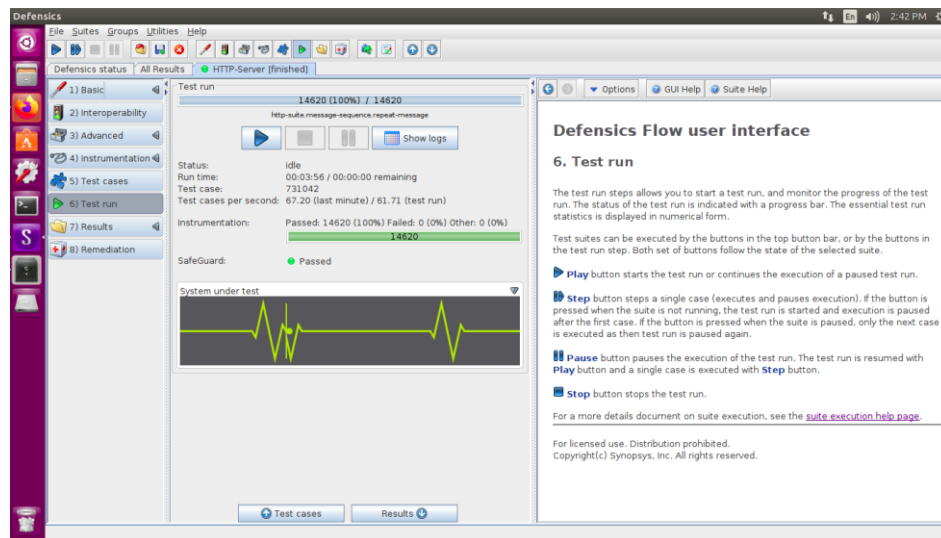
b. **Test Case Description:** To verify that the application-level protocols are robust against unexpected input.

c. **Execution Steps:**

- Tester sets the destination URL in the fuzzing tool including IP and designated incoming open port on DUT.



- Tester turns on Wireshark to capture packets received by the server.
- Tester should be familiar with the workings of designated fuzzing tools the suites used, the protocols to be tested and use maximum amount of test cases possible and repeat test numerous times and compile result.



d. Test Observations:

Pcap Trace and Report should be available. Sample Report is attached below. Pcap Trace is attached separately due to huge size. (Refer Tar ball attached):

Defensics Fuzz Test Report

Fuzz testing report

Test Setup

System Under Test

Name:

Fuzz testing report

Fuzz Test Summary

Overall verdict:

PASS

Tests executed:

14620

Test duration:

00:03:55

Reported by:

Test Run Summary

Test Run Summary

Suite	Version	Verdict	Number of test cases	Duration	Date	Instrumentation		
HTTP Server	4.14.0	PASS	14620	00:03:55	20230816	ENABLED	Protocol semantics	
						ENABLED	Connection based instrumentation	
						ENABLED	SafeGuard	
Total		PASS	14620	00:03:55				

e. Evidence Provided

A testing report provided by the testing agency which will consist of the following information:

- The used tool(s) name and version information,
- Settings and configurations used
- The output log file of the chosen tool that displays the results (passed/failed).
- Screenshot
- Test result (Passed or not)

- Log/evidence tracing possible crashes
- Any input causing unspecified, undocumented, or unexpected behaviour

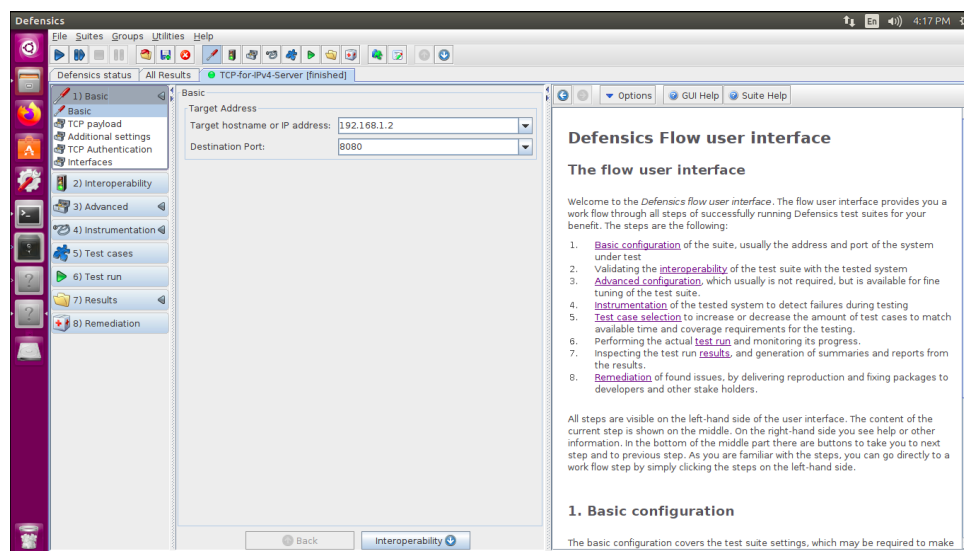
➤ **Test Case Number: 02**

a. **Test Case Name:** TC_FUZZ_TESTING_NETWORK

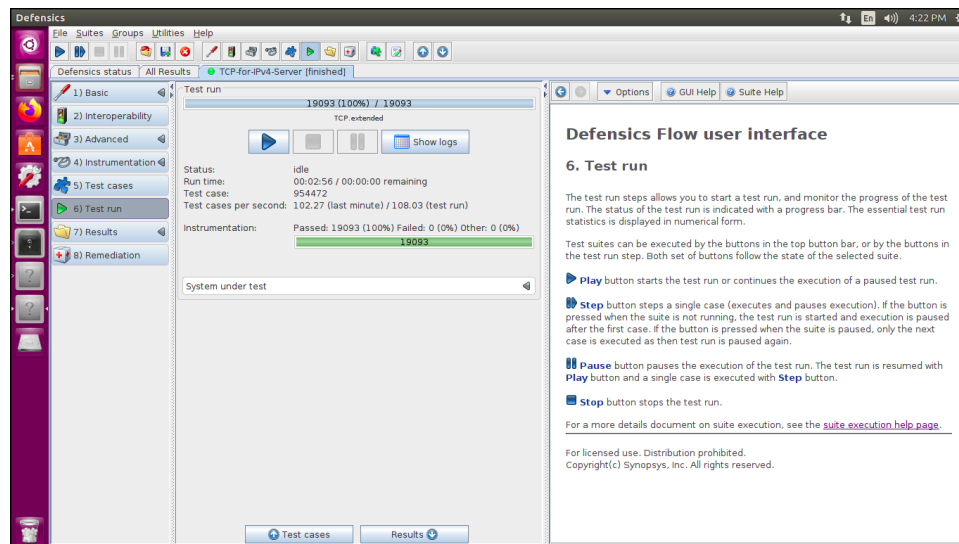
b. **Test Case Description:** To verify that the network-level protocols are robust against unexpected input. The following example uses TCP with IPv4. Testing also done with UDP with IPv4. If DUT supports IPv6, testing must also be done with TCP/UDP with IPv6.

c. **Execution Steps:**

- Tester sets the destination URL in the fuzzing tool including IP and designated incoming open port on DUT.



- Tester turns on Wireshark to capture packets received by the server.
- Tester should be familiar with the workings of designated fuzzing tools the suites used, the protocols to be tested and use maximum amount of test cases possible and repeat test numerous times and compile result.



- d. **Test Observations:** Pcap Trace and Report should be available. Sample Report is attached below. Pcap Trace is attached separately due to huge size. (Refer Tar ball attached):

TCP for IPv4 Server Test Suite(20230816-1619-28 - 20230816-1622-24)

Summary

This is a test report of 20230816-1619-28

Table of contents

- Summary
- Table of contents
- Test runs
- 20230816-1619-28 - TCP for IPv4 Server Test Suite
- Analysis
- Detail of service analysis
- Response analysis

Test runs

20230816-1619-28 : TCP for IPv4 Server Test Suite

Diagnoses

Valid case instrumentation	ENABLED
External instrumentation	DISABLED
Protocol semantics	DISABLED
Connection based instrumentation	DISABLED
SafeGuard	DISABLED
SNMP instrumentation	DISABLED
SNMP Trap instrumentation	DISABLED
ISASecure CCM	DISABLED
Syslog instrumentation	DISABLED
Agent instrumentation	DISABLED
Instrumentation fail limit	1
Instrumentation frequency	1

Total of 19093 cases were executed.

Combined overall verdict **PASS**

Verdict from valid case / instrumentation	pass	The test cases passed successfully. No flaws in the system under test were detected by the monitoring facility in context of these test cases.
0 fail		There is a reason to believe that the SUT crashed or malfunctioned in some other detectable way while executing these test cases.
18386 pass		The test cases passed successfully. No flaws in the system under test were detected by the monitoring facility in context of these test cases.
707 not-logged		Test cases passed successfully. No flaws in the system under test were detected by the monitoring facility in context of these test cases. In order to save disk space test cases are not logged.

Test run mode

- e. **Evidence Provided:-** A testing report provided by the testing agency which will consist of the following information:
- The used tool(s) name and version information,
 - Settings and configurations used
 - The output log file of the chosen tool that displays the results (passed/failed).
 - Screenshot
 - Test result (Passed or not)

- Log/evidence tracing possible crashes
- Any input causing unspecified, undocumented, or unexpected behaviour

12. **Test Case Results**

SL. No	TEST CASS NAME	PASS/FAIL	Remarks
1	TC_FUZZ_TESTING_APPLICATION		
2	TC_FUZZ_TESTING_NETWORK		



2.9.2 TSTP for Evaluation of Port Scanning

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.9.2
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) **<ITSAR Section No & Name>** Section 9: Vulnerability Testing Requirements
- 2) **<Security Requirement No & Name >** 2.9.2 Port Scanning
- 3) **<Requirement Description:>** It shall be ensured that on all network interfaces of SYSTEM, only documented ports on the transport layer respond to requests from outside the system.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.4.2]

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: **./system.out** (Used in IITH testbed to get SYSTEM version. Check with OEM manufacturer document for command specific to your SYSTEM)

Here we are assuming DUT to be SYSTEM, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: **cat /etc/os-release** (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5) DUT Configuration

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SYSTEM_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

- 6) **Preconditions:-** A list of all available network services containing at least the following information shall be included in the documentation accompanying the Network Product:
- all interfaces providing IP-based protocols;
 - the available transport layer protocols on these interfaces;
 - their open ports and associated services per transport layer protocol;
 - and a free-form description of their purposes.
- The port scanning tool that is used shall be capable to detect open ports on the relevant transport layer protocols.
- **NOTE:** It might not be possible for certain transport layer protocols (like UDP) to unambiguously detect whether a port is open or not by means of external port scanning. Also in some circumstances it might not be efficient to do external port scanning, e.g. if there are security measures to limit the rate a system can be probed. In those cases the accredited evaluator's test laboratory determines another means suitable to verify which ports are open.
- 7) **Test Objective:-** To ensure that on all network interfaces, only documented ports on the transport layer respond to requests from outside the system

8) **Test Plan**

8.1 **Number of Test Scenarios**

8.1.1. **Test Scenario for Port Scanning using Nmap**

All the ports on all supported transport layer protocols should be scanned using Nmap (Additional test scenarios are to be taken into consideration if tools like lsof, netstat are used)

8.1.2. **Test Bed Diagram**



8.1.3. **Tools Required:-** Nmap

8.1.4. **Test Execution Steps:-** Tester then shall verify if the ports mentioned as open by nmap on all the interfaces match the list provided by the vendor and also verify that no extra port is opened

9) **Expected Results for Pass**

- **Case 1:** All the ports discovered to be open by port scanning tool should match exactly to the list provided by the vendor.

10) **Expected Form of Evidence:-** Screenshot of Terminal

11) **Test Execution**

- **Test Case Number:** 01

- a) **Test Case Name:** TC_BVT_PORT_SCANNING_NMAP
- b) **Test Case Description:** On all network interfaces of DUT, only documented ports on the transport layer respond to requests from outside the system
- c) **Execution Steps:**
 - Tester shall scan all the available transport layer protocols ports of the DUT using nmap with the following command

For Example

- For scanning TCP Ports

- ***sudo nmap -p- -sT -v <DUT_IP>***

```
Starting Nmap 7.80 ( https://nmap.org ) at 2023-07-11 22:17 IST
Nmap scan report for 192.168.127.177
Host is up (0.073s latency).
Not shown: 65529 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
111/tcp   open  rpcbind
902/tcp   open  iss-realsecure
4000/tcp  open  remoteanything
7070/tcp  open  realserver
```

- For scanning UDP Ports

- ***sudo nmap -p- -sU -v <DUT_IP>***

```
Starting Nmap 7.80 ( https://nmap.org ) at 2023-07-11 23:57 IST
Nmap scan report for 192.168.127.177
Host is up (0.095s latency).

PORT      STATE      SERVICE
1234/udp   open|filtered search-agent

Nmap done: 1 IP address (1 host up) scanned in 1.34 seconds
```

- For scanning SCTP ports

- ***sudo nmap -p- -sY -v <DUT_IP>***

```

Starting Nmap 7.80 ( https://nmap.org ) at 2023-07-12 00:10 IST
Initiating Ping Scan at 00:10
Scanning 192.168.127.177 [4 ports]
Completed Ping Scan at 00:10, 0.10s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 00:10
Completed Parallel DNS resolution of 1 host. at 00:10, 0.09s elapsed
Initiating SCTP INIT Scan at 00:10
Scanning 192.168.127.177 [1 port]
Discovered open port 1234/sctp on 192.168.127.177
Completed SCTP INIT Scan at 00:10, 0.09s elapsed (1 total ports)
Nmap scan report for 192.168.127.177
Host is up (0.058s latency).

PORT      STATE SERVICE
1234/sctp open  unknown

Read data files from: /usr/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.42 seconds
Raw packets sent: 5 (204B) | Rcvd: 23 (4.288KB)

```

- Tester then shall verify if the ports mentioned as open by nmap on all the interfaces match the list provided by the vendor and also verify that no extra port is opened

d) **Test Observations:**

- **Case 1:** All the ports discovered to be open by port scanning tool match exactly to the list provided by the vendor.

e) **Evidence Provided:-** Screenshot of Terminal

12) **Test Case Results**

SL. No	TEST CASS NAME	PASS/FAIL	Remarks
1	TC_BVT_PORT_SCANNING_NMAP		

Securing Networks

2.9.3 TSTP Report for Evaluation of Vulnerability Testing Requirements

Session Management Function ITSAR	ITSAR No: ITSAR111092401	Clause no: 2.9.3
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **ITSAR Section No & Name:** Section 9: Vulnerability Testing Requirements
2. **Security Requirement No & Name:** 2.9.3 Vulnerability Scanning
3. **Requirement Description:** The vulnerabilities found during the Vulnerability Scanning/Assessment process shall be remediated as below. For other than critical vulnerabilities, OEM shall provide remediation plan.

SI No	CVSS Score	Severity	Remediation
1	9.0-10.0	Critical	To be patched immediately
2	7.0-8.9	High	To be patched within a month
3	4.0-6.9	Medium	To be patched within three months
4	0.1-3.9	Low	To be patched within a year

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.4.3]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. **DUT Configuration:** Check the Operating system and its version for the DUT to know the commands for setting access permissions for data and application execution.

- **To get the hash of configuration file if the file is a ASCII text file.**

- Command - **sha256sum DUT_config.conf**

- **Digest Hash of Tested Configuration:**

- DUT_config.conf:-

19be54b42975f7f414c53a280373aa9466398781ecd77a2e9d090dbc6dbfdd8

- **To get the hash of OS if using docker**

- Command - **docker images --digests**

- **Digest Hash of OS:**

- DUT_IMAGE:

fd363f0e0d146c869d60649bc4f42e7008829d9d67a5dfdaf3b38ad24af7d53a

A known vulnerability scanning tool that has capabilities to support the given testing should be present. Following TSTP is developed using NESSUS. Any their tool with similar capability can

be also used, but documentation should be provided that the tool meets the need of the requirement.

6. **Preconditions:** A list of all available network services containing at least the following information shall be included in the documentation accompanying the Network Product:

- all interfaces providing IP-based protocols;
- the available transport layer protocols on these interfaces;
- their open ports and associated services;
- and a free-form description of their purposes.

7. **Test Objective:** The purpose of vulnerability scanning is to ensure that there are no known vulnerabilities (or that relevant vulnerabilities are identified and remediation plans in place to mitigate them) on the Network Product that can be detected by means of automatic testing tools via the Internet Protocol enabled network interfaces.

8. **Test Plan:**

8.1 Tools Used: NESSUS

8.2 Test Execution steps: The accredited evaluator's test lab is required to execute the following steps:

- Execution of the suitable vulnerability scanning tool against all interfaces providing IP-based protocols of the Network Product.
- Evaluation of the results based on their severity.

9. **Expected Results for Pass:** Report showing that SMF any existing vulnerability in SMF, with it's CVSS score and remediation.

10. **Expected Format of Evidence:**

- The used tool(s) name, their unambiguous version (also for plug-ins if applicable), used settings, and the relevant output is evidence and shall be part of the testing documentation.
- The discovered vulnerabilities (including source, example CVE ID), together with a rating of their severity, shall be highlighted in the testing documentation.

11. **Test Execution:**

➤ **Test Case Number:** 1

a. Test Case Name: TC_BVT_VULNERABILITY_SCANNING

b. Test Case Description: To ensure that NF is free from any vulnerabilities.

c. Execution Steps: The accredited evaluator's test lab is required to execute the following steps:

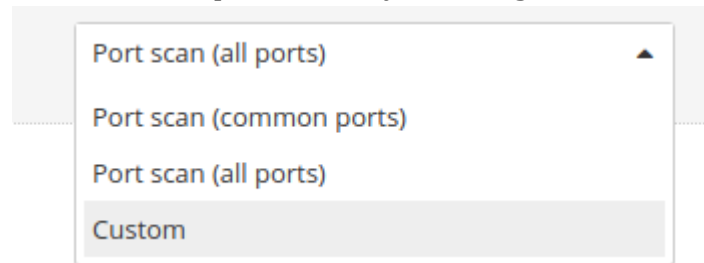
- Execution of the suitable vulnerability scanning tool against all interfaces providing IP-based protocols of the Network Product.

- Evaluation of the results based on their severity.

d. Test Observations:


- **Case 1:** Tester scans the required target (IP address) and checks the scan summary. We can see in the screenshot below; the given interface is scanned for vulnerabilities. Various vulnerabilities with their CVE ID's are shown. All the critical/high/medium vulnerabilities should be dealt with seriously.

Set the ports for which scan is to be performed by selecting the custom option.

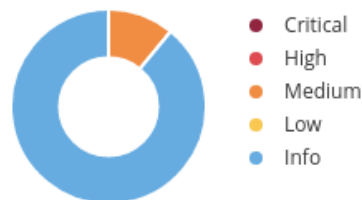


Classification of vulnerability based on the colour:-

Scan Details

Policy: Basic Network Scan
 Status: Running 
 Severity Base: CVSS v3.0
 Scanner: Local Scanner
 Start: Today at 8:55 PM

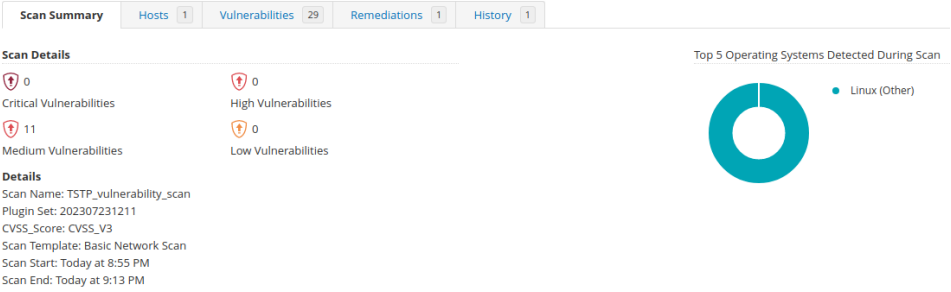
Vulnerabilities



Report as shown below is generated. It contains the CVSS score of the vulnerability as well.

Sev	CVSS	VPR	Name	Family	Count
MEDIUM	6.5		SSL Certificate Cannot Be Trusted	General	5
MEDIUM	6.5		SSL Self-Signed Certificate	General	4
INFO			SSL Certificate Information	General	5
INFO			SSL Cipher Suites Supported	General	5
INFO			SSL Perfect Forward Secrecy Cipher Suites Supported	General	5

Complete scan summary: -



2.10.1 TSTP Report for Evaluation of Growing Content Handling

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.1
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>**Section 10: Operating System

2. **<Security Requirement No & Name >** 2.10.1 Growing Content Handling

3. **<Requirement Description: >**

a) Growing or dynamic content shall not influence system functions.

b) A file system that reaches its maximum capacity shall lead to an event getting logged with appropriate message parameters and shall not stop SMF from operating properly.

Therefore, countermeasures shall be taken to ensure that this scenario is avoided.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.4.1.1.1]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

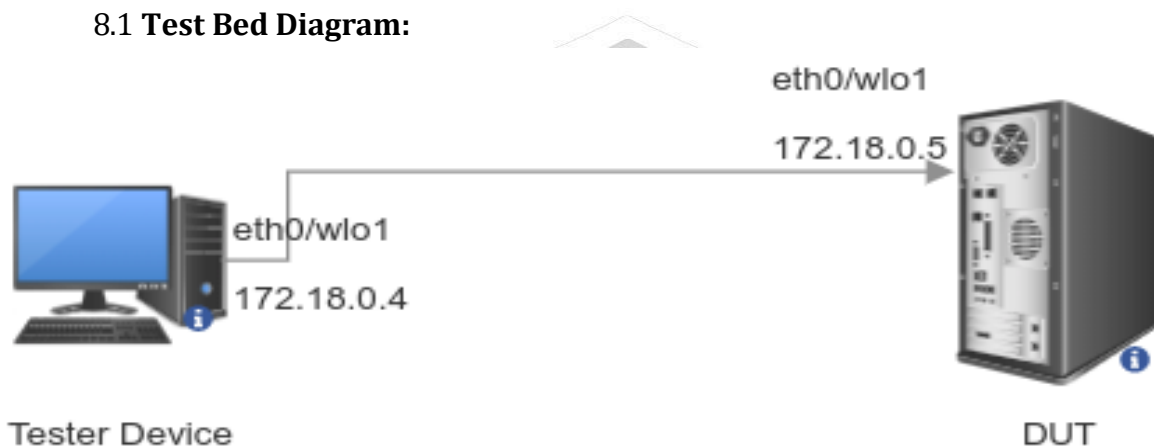
6. **Preconditions:**

- Growing or dynamic content sources like e.g. log files and their paths are documented by the Vendor.
- Measures that are taken to protect system functions from growing or dynamic content that may exhaust file system capacity are documented.
- All logging capabilities that are not enabled by default are enabled manually as per the documentation instructions.

7. **Test Objective:** To verify that the growing or dynamic content does not influence system functions.

8. **Test Plan:**

8.1 Test Bed Diagram:



8.2 Tools used: NULL.

8.3 Execution steps:

1. Tester checks that the sources that are susceptible to being exhausted have been documented and measures aimed to counter this are described.
2. Tester enables monitoring of the system operation.
3. Tester initiates traffic that causes increase of log files and monitors the system behaviour until the log file either reaches its quota or until file system is exhausted.
4. In case file uploading is allowed (e.g. via SFTP) the tester initiates file uploading and tries to exhaust the file system.

9. **Expected Results of Pass:** It is verified that the taken measures are sufficient so that system operation is not influenced by growing or dynamic content at any case.

10. **Expected Format of Evidence:** Screenshot / log files of the event getting logged with appropriate message parameters

11. Test Execution:

➤ Test Case Number: 1

a. Test Case Name: TC_GROWING_CONTENT_HANDLE

b. Test Case Description:

To verify that the growing or dynamic content does not influence system functions.

c. Test Execution Steps:

1. Tester checks that the sources that are susceptible to being exhausted have been documented and measures aimed to counter this are described.
2. Tester enables monitoring of the system operation.
3. Tester initiates traffic that causes increase of log files and monitors the system behaviour until the log file either reaches its quota or until file system is exhausted.
4. We run our 5g Core with a large no. Of UE traffic that generates log files for each NF in the logs folder.

```
siddhesh@stark99:~$ docker run -it --rm --name amf2 --cap-add NET_ADMIN --network sgmc-v /home/sidhesh/thesis_work/Implementations/iith_core_thesis/phase3/5G/src5g -v /home/sidhesh/thesis_work/Implementations/iith_core_thesis/phase3/5G/src5g:code sgstestbed.docker.local:5000/g3_f3al_common:latest
root@172.18.0.2:/# cd code/src
root@172.18.0.2:/code/src# ./amfnew.out
[5g]NasSecurity constructor invoked
File name is .....InsgTaus2023-07-25-05-47-12-898
NF IP Address is 172.18.0.6
2023-07-25 05:47:12:908 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstanceIdDocumentApi.cpp:13 operator()Successfully Deregistered : AMF
2023-07-25 05:47:12:908 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstanceIdDocumentApi.cpp:42 operator()Successfully Registered:{"$nasaliset":true,"ipAddresses":["172.18.0.6:8080","172.18.0.6:8080"],"nfInstanceId":"","nfStatus":"null","nfType":"AMF","$nasals":[{"sst":0}]}
Installing AMF State
Started AMF server at 172.18.0.6:38412
Server initialized

root@172.18.0.2:/code/src# ./ausf.out
File name is .....InsgTaus2023-07-25-05-47-57-623
2023-07-25 05:47:57:681 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstanceIdDocumentApi.cpp:138 operator()Successfully Deregistered : AUSF
2023-07-25 05:47:57:685 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstanceIdDocumentApi.cpp:422 operator()Successfully Registered:{"$nasaliset":true,"ipAddresses":["172.18.0.7:8080","172.18.0.7:8080"],"nfInstanceId":"","nfStatus":"null","nfType":"AUSF","$nasals":[{"sst":0}]}
Exiting log location: /logs/InsgTaus2023-07-25-05-47-57623.log
root@172.18.0.2:/code/src# ./ausf.out
File name is .....InsgTaus2023-07-25-05-48-10-585
2023-07-25 05:48:10:522 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstanceIdDocumentApi.cpp:138 operator()Successfully Deregistered : AUSF
2023-07-25 05:48:10:535 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstanceIdDocumentApi.cpp:422 operator()Successfully Registered:{"$nasaliset":true,"ipAddresses":["172.18.0.7:8080","172.18.0.7:8080"],"nfInstanceId":"","nfStatus":"null","nfType":"AUSF","$nasals":[{"sst":0}]}
2023-07-25 05:48:10:537 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstancesStoreApi.cpp:385 operator()Successfully Discovered:{"$supportedFeatures":"","validityPeriod":"","nfInstances":[{"Capacity":0,"fqdn":"","load":"0","locality":"","nfServicePersistence":"false","Priority":0,"RecoveryTime":"","ipAddresses":["172.18.0.8:8080","172.18.0.8:8080"],"nfInstanceId":"1","nfType":"UDM","$nasals":[{"sst":0}]}]}
Target NF IP address: 172.18.0.8 port:8080
Target NF IP address: 172.18.0.8 port:8080
Target NF Service address:

root@172.18.0.2:/code/src# ./smf.out
operator()Successfully Deregistered : SMF
2023-07-25 05:48:40:859 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstanceIdDocumentApi.cpp:422 operator()Successfully Registered:{"$nasaliset":true,"ipAddresses":["172.18.0.9:8080","172.18.0.9:8080"],"nfInstanceId":"","nfStatus":"null","nfType":"SMF","$nasals":[{"sst":0}]}
2023-07-25 05:48:40:860 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstanceIdDocumentApi.cpp:422 operator()Successfully Registered:{"$nasaliset":true,"ipAddresses":["172.18.0.9:8080","172.18.0.9:8080"],"nfInstanceId":"","nfStatus":"null","nfType":"SMF","$nasals":[{"sst":1}]}
2023-07-25 05:48:40:863 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstanceIdDocumentApi.cpp:422 operator()Successfully Registered:{"$nasaliset":true,"ipAddresses":["172.18.0.9:8080","172.18.0.9:8080"],"nfInstanceId":"","nfStatus":"null","nfType":"SMF","$nasals":[{"sst":2}]}
2023-07-25 05:48:40:865 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstanceIdDocumentApi.cpp:422 operator()Successfully Registered:{"$nasaliset":true,"ipAddresses":["172.18.0.9:8080","172.18.0.9:8080"],"nfInstanceId":"","nfStatus":"null","nfType":"SMF","$nasals":[{"sst":3}]}
Fetched 25166880
discovering UPF....
2023-07-25 05:48:40:867 INF>[NRF_CLIENT]/code/nrf_client/lib/api/NFInstancesStoreApi.cpp:385 operator()Successfully Discovered:{"$supportedFeatures":"","validityPeriod":"","nfInstances":[{"Capacity":0,"fqdn":"","load":"0","locality":"","nfServicePersistence":"false","Priority":0,"RecoveryTime":"","ipAddresses":["172.18.0.10:8085","172.18.0.10:8085"],"nfInstanceId":"1","nfType":"UPF","$nasals":[{"sst":0}]}]}
Target NF IP address: 172.18.0.10 port:8085
Target NF IP address: 172.18.0.10 port:8085
Target NF Service address: 172.18.0.10
Target NF Service address: 172.18.0.10
discovered UPF....
Obtained UPF Info Ip 172.18.0.10, port 8085
```

We have a mechanism within the core that monitors access to log folder and notifies with an alert on terminal and logs that file/folder size has exceeded the limit.

```
siddhesh@stark99:~/thesis_work/Implementations/iith_core_thesis/phase3/5G/src$ ./sizeCheck.sh
Setting up watches.
Watches established.
./logs/ ACCESS.ISDIR
size is over 100 kilobytes

siddhesh@stark99:~/thesis_work/Implementations/iith_core_thesis/phase3/5G/src$ cat sizeCheck.log
1 2023-07-25 08:51:06 INFO :...file/folder name : ./logs, EXCEEDED MAXIMUM LIMIT size is over 100 kilobytes
2 2023-07-25 08:59:28 INFO :...file/folder name : ./logs, EXCEEDED MAXIMUM LIMIT size is over 100 kilobytes
3 2023-07-25 09:01:16 INFO :...file/folder name : ./logs, EXCEEDED MAXIMUM LIMIT size is over 100 kilobytes
4 2023-07-25 09:03:31 INFO :...file/folder name : ./logs, EXCEEDED MAXIMUM LIMIT size is over 100 kilobytes
```

(Note: The tester must produce the respective proof(s) for alerting and logging in case the file size exceeds, as mentioned in the Vendor document)

Screenshot of SMF still running properly:

```
amf2
siddhesh@stark99:~$ docker run -ti --rm --name amf2 --cap-add NET_ADMIN --network 5gmec -v /home/siddhesh/thesis_work/Implementations/iith_core_thesis/phase3/5G/:/code 5gtestbed.docker.local:5000/g3_f
inal_common:latest
root@70182d7abd92:/# cd code/src
root@70182d7abd92:/code/src# ./amfnew.out
In5GtNasSecurity constructor invoked
File name is-----In5GtAmf2023-07-25-05-47-12 898
AMF IP Address is 172.18.0.6
<2023-07-25 05:47:12:906 INF>|NRF_CLIENT|/code/NRF/nrf_client/lib/api/NFInstanceIDDocumentApi.cpp:13
6|operator()|Successfully Deregistered : AMF
<2023-07-25 05:47:12:908 INF>|NRF_CLIENT|/code/NRF/nrf_client/lib/api/NFInstanceIDDocumentApi.cpp:42
2|operator()|Successfully Registered:{"SNssalsIsSet":true,"Ipv4Addresses":["172.18.0.6:8080","172
.18.0.6:8080"],"nfInstanceId":"","nfStatus":null,"nfType":"AMF","sNssals":[{"sst":0}]}
Initializing AMF State
Started AMF server at172.18.0.6:38412
server initialized
```

5. In case file uploading is allowed the tester initiates file uploading and tries to exhaust the file system.

We upload a file exceeding the size limit in the logs folder, and again the same alerts as above are generated.

(Note: The tester must produce the respective proof(s) for alerting and logging in case the file size exceeds, as mentioned in the Vendor document)

d. Test Observations: he file system on reaching its maximum capacity leads to an event getting logged with appropriate message parameters and shall not stop SMF from operating properly.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_GROWING_CONTENT_HANDLE		

Securing Networks

2.10.2 TSTP for Evaluation of Handling of ICMP

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.2
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name:>** Section 10: Operating System
2. **<Security Requirement No & Name:>** 2.10.2 Handling of ICMP
3. **<Requirement Description:>**

Processing of ICMPv4 and ICMPv6 packets which are not required for operation shall be disabled on the SMF.

SMF shall not send certain ICMP types by default, but it may support the option to enable utilization of these types which are marked as "Optional" in below table:

Type (IPv4)	Type (IPv6)	Description	Send	Respond to
0	128	Echo Reply	Optional (i.e., as automatic reply to "Echo Request")	N/A
3	1	Destination Unreachable	Permitted	N/A
8	129	Echo Request	Permitted	Optional
11	3	Time Exceeded	Optional	N/A
12	4	Parameter Problem	Permitted	N/A
N/A	2	Packet too Big	Permitted	N/A
N/A	135	Neighbour Solicitation	Permitted	Permitted

N/A	136	Neighbour Advertisement	Permitted	N/A
-----	-----	-------------------------	-----------	-----

SMF shall not respond to, or process (i.e., do changes to configuration) under any circumstances certain ICMP message types as marked in the below table.

Type (IPv4)	Type (IPv6)	Description	Send	Respond to	Process (i.e., do changes to configuration)
5	137	Redirect	N/A	N/A	Not Permitted
13	N/A	Timestamp	N/A	Not Permitted	N/A
14	N/A	Timestamp Reply	Not Permitted (i.e., as automatic reply to "Timestamp")	N/A	N/A
N/A	133	Router Solicitation	N/A	Not Permitted	Not Permitted
N/A	134	Router Advertisement	N/A	N/A	Not Permitted

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.4.1.1.2.]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

Note: There can be multiple ways to check for the disabled ICMP messages. It will depend on the DUT configuration. To generate ICMP packet, here Scapy and nping tool is used but there are other tools which can be used and should be installed on the DUT by the vendor. Proper specification of these tools should be given in the document. For network analyser Wireshark is used here but other tools can also be used.

- To check if all the options are disabled, we can go to the file

- `sudo gedit /etc/sysctl.conf`

```
Open  ▾  [icon]  sysctl.conf  /etc

35
36 #####
37 # Additional settings - these settings can improve the network
38 # security of the host and prevent against some network attacks
39 # including spoofing attacks and man in the middle attacks through
40 # redirection. Some network environments, however, require that these
41 # settings are disabled so review and enable them as needed.
42 #
43 # Do not accept Router Solicitation and Router Advertisement
44 net.ipv6.conf.all.accept_ra = 0
45 net.ipv6.conf.default.accept_ra = 0
46 #
47 # Do not accept ICMP redirects (prevent MITM attacks)
48 net.ipv4.conf.all.accept_redirects = 0
49 net.ipv6.conf.all.accept_redirects = 0
50 # _or_
51 # Accept ICMP redirects only for gateways listed in our default
52 # gateway list (enabled by default)
53 # net.ipv4.conf.all.secure_redirects = 1
54 #
55 # Do not send ICMP redirects (we accept them ourselves)
```

- To check for nping Use the command: **nping --version**

```
(base) unnati@unnati-Super-Server:~$ nping --version
Nping version 0.7.80 ( https://nmap.org/nping )
(base) unnati@unnati-Super-Server:~$
```

- To check for wireshark
Use the command: **wireshark--version**

```
(base) unnati@unnati-Super-Server:~$ wireshark --version
Wireshark 3.6.2 (Git v3.6.2 packaged as 3.6.2-2)

Copyright 1998-2022 Gerald Combs <gerald@wireshark.org> and contributors.
License GPLv2+: GNU GPL version 2 or later <https://www.gnu.org/licenses/gpl-2.0.html>
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Compiled (64-bit) using GCC 11.2.0, with Qt 5.15.2, with libpcap, with POSIX
capabilities (Linux), with libnl 3, with GLib 2.71.2, with zlib 1.2.11, with Lua
5.2.4, with GnuTLS 3.7.3 and PKCS #11 support, with Gcrypt 1.9.4, with MIT
Kerberos, with MaxMind DB resolver, with nghttp2 1.43.0, with brotli, with LZ4,
with Zstandard, with Snappy, with libxml2 2.9.12, with libsmi 0.4.8, with
QtMultimedia, without automatic updates, with SpeexDSP (using system library),
with Minizip.

Running on Linux 5.19.0-46-generic, with Intel(R) Xeon(R) W-2133 CPU @ 3.60GHz
(with SSE4.2), with 31786 MB of physical memory, with GLib 2.72.4, with zlib
1.2.11, with Qt 5.15.3, with libpcap 1.10.1 (with TPACKET_V3), with c-ares
1.18.1, with GnuTLS 3.7.3, with Gcrypt 1.9.4, with nghttp2 1.43.0, with brotli
1.0.9, with LZ4 1.9.3, with Zstandard 1.4.8, with libsmi 0.4.8, with
LC_TYPE=en_IN, binary plugins supported (0 loaded).
(base) unnati@unnati-Super-Server:~$
```

- To check for scapy

Use the command: **scapy**

```
(base) unnati@unnati-Super-Server: $ scapy
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:671: ImportWarning: _SixMetaPathImporter.exec_module() not found; falling back to load_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
INFO: Can't import PyX. Won't be able to use psdump() or pdfdump().
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:671: ImportWarning: _SixMetaPathImporter.exec_module() not found; falling back to load_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
<frozen importlib._bootstrap>:914: ImportWarning: _SixMetaPathImporter.find_spec() not found; falling back to find_module()
aSPV//YASa
apyyyyCY////////YCa
sY////////YSpCs scpCY//Pp
ayp ayyyyyySCP//Pp syY//C
AYAsAYYYYYYYY//Ps cY//S
pCCCCY//p cSSps y//Y
SPPPP//a pP//AC//Y
A//A cyP//C
p//Ac sC//a
P//Ycpc A//A
scccccp//pSP//p p//Y
sY////////y caa S//P
cayCyayP//Ya pY//Ya
sY/PSY//YcC aC//Yp
sc sccaCY//PCypaapyCP//YsS
spCPY////////YPSps
ccaacs
using IPython 7.31.1
>>> exit()

Welcome to Scapy
Version 2.4.4
https://github.com/secdev/scapy
Have Fun!

To craft a packet, you have to be a
packet, and learn how to swim in
the wires and in the waves.
-- Jean-Claude Van Damme
```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. Preconditions:

- The tester knows whether the network product supports IPv4 and/or IPv6:
- If applicable, the tester has the needed system privileges for confirming that the ICMP messages with types "Not Permitted" to process are indeed not leading to configuration changes.
- If applicable, the tester has the needed system privileges for confirming that certain ICMP message types are dropped by the network product on receipt.
- A tester machine is available and equipped with a suitable ICMP packets generator tool.

7. Test Objective:

- To verify that the network product does not reply to certain ICMP types in accordance with the requirement.
- To verify that the network product does not send 'Time Exceeded'.

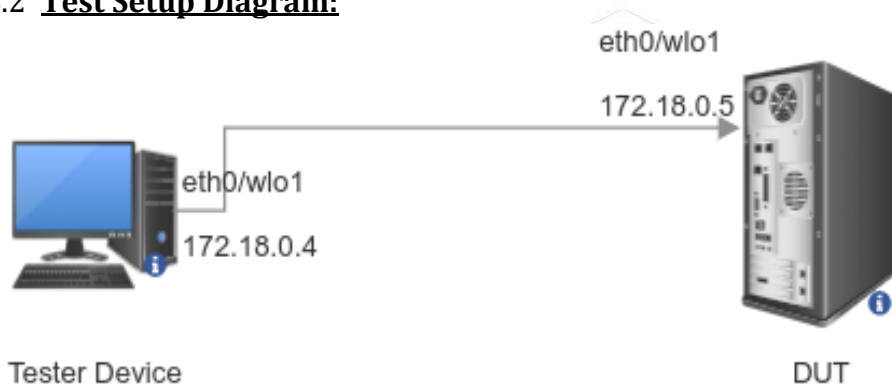
- To verify that the network product does not process the following ICMPv4 and ICMPv6 types:
 - Redirect (5)
 - Router Solicitation
 - Router Advertisement

Note: The test for this requirement can be carried out using a suitable tool or manually by performing the steps described below. If a tool is used then the tester needs to provide evidence, e.g., by referring to the documentation of the tool, that the tool provides functionality equivalent to the steps described below.

8. **Test Plan:**

8.1 **Number of Test Scenarios: 1**

8.2 **Test Setup Diagram:**



8.3 **Tools Used:**

- Scapy tool
- Wireshark
- Nping

8.4 **Test Execution Steps:**

- All preconditions should be met.
- The following needs to be done for all IP protocol versions (IPv4 and/or IPv6) supported by the network element.
- For verifying that the network product does not reply to ICMP messages with types where this is not permitted: The tester sends samples of the applicable ICMP messages from the tester machine to the network product and verifies by appropriate means that
 - The messages are dropped on receipt by the network product (e.g., by means of appropriate firewall rules)
 - or no response is sent out towards the test machine,
 - or there are other means ensuring that the ICMP messages cannot trigger a response.
- For verifying that the network product does not change its configuration due to receiving ICMP messages with types where this is not permitted: The tester sends samples of the

applicable ICMP messages from the tester machine to the network product and verifies by appropriate means that

- the messages are dropped on receipt by the network product (e.g., by means of appropriate firewall rules)
- or no response is sent out towards the test machine,
- or there are other means ensuring that the ICMP messages cannot trigger a response.
- The tester utilizes appropriate means to verify consistency between the documentation about ICMP and the network product.

9. **Expected Results for Pass:**

- The ICMP messages which are "Not Permitted" or "Optional" to generate a response from the network product do not generate a response.
- The ICMP messages which are "Not Permitted" to change the configuration of the network element do not change the configuration.
- ICMP message types which lead to responses or to configuration changes on receipt, if neither mentioned in the requirement nor in the documentation, are not enabled

10. **Expected Format of Evidence:**

- The following information needs to be retained and included into the report as appropriate:
 - Tools used and their configuration
 - Tool output
 - Test result (Passed or not)

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC1_HANDLING_OF_ICMP

b. **Test Case Description:** To ensure that the network product does not reply or process to certain ICMP messages which are listed as "not permitted" in the above table.

c. **Execution Steps:**

● **Case 1: ICMP Timestamp**

Command to send ICMP message to DUT using nping to check if it replies for timestamp messages

sudo nping --icmp --icmp-type 13 192.168.122.56

```
(base) unnati@unnati-Super-Server: $ sudo nping --icmp --icmp-type 13 192.168.122.56

Starting Nping 0.7.80 ( https://nmap.org/nping ) at 2023-07-28 17:48 IST
SENT (0.0440s) ICMP [192.168.122.1 > 192.168.122.56 Timestamp request (type=13/code=0) id=17935 seq=1 orig=0 rcv=0 trans=0] IP [ttl=64 id=10555 plen=40 ]
SENT (1.0441s) ICMP [192.168.122.1 > 192.168.122.56 Timestamp request (type=13/code=0) id=17935 seq=2 orig=0 rcv=0 trans=0] IP [ttl=64 id=10555 plen=40 ]
SENT (2.0454s) ICMP [192.168.122.1 > 192.168.122.56 Timestamp request (type=13/code=0) id=17935 seq=3 orig=0 rcv=0 trans=0] IP [ttl=64 id=10555 plen=40 ]
SENT (3.0466s) ICMP [192.168.122.1 > 192.168.122.56 Timestamp request (type=13/code=0) id=17935 seq=4 orig=0 rcv=0 trans=0] IP [ttl=64 id=10555 plen=40 ]
SENT (4.0478s) ICMP [192.168.122.1 > 192.168.122.56 Timestamp request (type=13/code=0) id=17935 seq=5 orig=0 rcv=0 trans=0] IP [ttl=64 id=10555 plen=40 ]

Max rtt: N/A | Min rtt: N/A | Avg rtt: N/A
Raw packets sent: 5 (200B) | Rcvd: 0 (0B) | Lost: 5 (100.00%)
Nping done: 1 IP address pinged in 5.10 seconds
(base) unnati@unnati-Super-Server: $ sudo adduser vmi
```

We can check in this Wireshark capture; the host machine does not reply to the Timestamp message.

icmp						
No.	Time	Source	Destination	Protocol	Length	Info
1031	5.355598380	192.168.122.1	192.168.122.56	ICMP	56	Timestamp request id=0x460f, seq=1/256, ttl=64
1032	5.355605684	192.168.122.1	192.168.122.56	ICMP	56	Timestamp request id=0x460f, seq=1/256, ttl=64
1194	6.355766192	192.168.122.1	192.168.122.56	ICMP	56	Timestamp request id=0x460f, seq=2/512, ttl=64
1195	6.355774149	192.168.122.1	192.168.122.56	ICMP	56	Timestamp request id=0x460f, seq=2/512, ttl=64
1330	7.357022750	192.168.122.1	192.168.122.56	ICMP	56	Timestamp request id=0x460f, seq=3/768, ttl=64
1331	7.357034944	192.168.122.1	192.168.122.56	ICMP	56	Timestamp request id=0x460f, seq=3/768, ttl=64
1663	8.358262942	192.168.122.1	192.168.122.56	ICMP	56	Timestamp request id=0x460f, seq=4/1024, ttl=64
1664	8.358273257	192.168.122.1	192.168.122.56	ICMP	56	Timestamp request id=0x460f, seq=4/1024, ttl=64
1702	8.358400705	192.168.122.1	192.168.122.56	ICMP	56	Timestamp request id=0x460f, seq=5/1280, ttl=64

•Case 2: Router Advertisement

Using Scapy when we send a router advertisement message to the DUT. We can check in the below screenshot; we are not getting any response.

```
(base) unnati@unnati-Super-Server:~$ sudo python3 router_advertisement.py
.
Sent 1 packets.
(base) unnati@unnati-Super-Server:~$ sudo ip link add eth2:1 type vlan id 1
```

ip6v6.addr == fe80::562b:79e:60f4:fa05						
No.	Time	Source	Destination	Protocol	Length	Info
1178	14.010626410	fe80::fefb:7e96:1d1...	fe80::562b:79e:60f4...	ICMPv6	104	Router Advertisement
16577	42.114874715	fe80::fefb:7e96:1d1...	fe80::562b:79e:60f4...	ICMPv6	104	Router Advertisement

•Case 3: Router Solicitation

Using Scapy when we send a router solicitation message from the host. We can check in the below screenshot; we are not able to send the packet.

```
vm2@vm2:~$ sudo python3 icmp_router_solicitation.py
.
Sent 1 packets.
Packet sending failed.
vm2@vm2:~$ ifconfig
```

icmpv6									
No.	Time	Source	Destination	Protocol	Length	Info			
	24.40.255790385	::	fe80::fe96:1d1...	ICMPv6	64	Router Solicitation			
Frame 24: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface any, id 0									
Linux cooked capture v1									
Internet Protocol Version 6, Src: ::, Dst: fe80::fe96:1d1:e530									
Internet Control Message Protocol v6									
Type: Router Solicitation (133)									
Code: 0									
Checksum: 0x17a8 [correct]									
[Checksum Status: Good]									
Reserved: 00000000									
0000	00 04 00 01 00 06 52 54	00 1d 2a 3c 00 00 86 ddRT	..*	<.....				
0010	60 00 00 00 00 08 3a ff	00 00 00 00 00 00 00 00:				
0020	00 00 00 00 00 00 00 00	fe 80 00 00 00 00 00 00fe	80 00 00 00 00 00 00				
0030	fe fb 7e 96 01 d1 e5 30	85 00 17 a8 00 00 00 000				

• Case 4: Redirect

To check for this redirect, we can ping from DUT to tester machine and check if it changes the IP route table.

Here we can see before pinging and after pinging the output is same.

Command: *ip route*

```

vm2@vm2:~$ ip route
default via 192.168.122.1 dev enp1s0 proto dhcp metric 100
169.254.0.0/16 dev enp1s0 scope link metric 1000
192.168.122.0/24 dev enp1s0 proto kernel scope link src 192.168.122.56 metric 10
0
vm2@vm2:~$ ping 192.168.126.132
PING 192.168.126.132 (192.168.126.132) 56(84) bytes of data.
64 bytes from 192.168.126.132: icmp_seq=1 ttl=64 time=0.169 ms
64 bytes from 192.168.126.132: icmp_seq=2 ttl=64 time=0.253 ms

vm2@vm2:~$ ip route
default via 192.168.122.1 dev enp1s0 proto dhcp metric 100
169.254.0.0/16 dev enp1s0 scope link metric 1000
192.168.122.0/24 dev enp1s0 proto kernel scope link src 192.168.122.56 metric 10
0

```

Also, we can check Wireshark capture there are no redirect messages.

icmp.type == 5						
No.	Time	Source	Destination	Protocol	Length	Info

d. Test Observations:

The DUT should ensure that the ICMP options which are there as Not Permitted should be disabled.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_HANDLING_OF_ICMP		

2.10.3 TSTP For Authenticated Privilege Escalation only

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.1
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<TSTP Document ID:>
<Applicant Name:> Ex: XYZ
<Application Number>
<DUT Details: > Ex: Router
<DUT Software Version:>
<Digest Hash of OS>
<Digest Hash of Configuration>
<Applicable ITSAR: >
<ITSAR Version No:>
<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 10 – Operating System
2. **<Security Requirement No & Name >** 2.10.3 Authenticated Privilege Escalation only
3. **<Requirement Description: >**

SMF shall not support a privilege escalation method in interactive sessions (both CLI and GUI) which allows a user to gain administrator/root privileges from another user account without re- authentication.

[Reference: TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.4.1.2.1]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:**

- The manufacturer shall provide documentation of the operating system(s) used in the network product.
- The manufacturer shall supply a list "A" of operating system functions which a system user can use to explicitly gain higher privileges, and how these functions are configured. Unix® example: sudo command and its configuration file /etc/sudoers.
- The manufacturer shall supply a list "B" of operating system commands, GUI functions, and files which will execute specifically limited tasks automatically with higher privileges, even when used by a low-privileged user. List "B" shall also contain:
 - configuration of these commands and GUI functions;
 - owner and permission settings of files;
 - justification for having the command, GUI function or file on the network product Unix® example: root-owned files with SUID and SGID permissions.

7. **Test Objective/ Purpose:** To ensure that privileged operating system functions shall not be used without successful authentication and authorization, and that violations of this requirement are documented and strictly limited in number and functionality.

8. **Test Plan:**

8.1 **Tools required:** Command Line Interface of the DUT

8.2 **Test Set Up:**

8.3 **Test Execution Step:** The accredited evaluator's test lab is required to execute the following steps:

- The tester logs into the network product and verifies that list "A" is accurate, based on his expert knowledge of the operating system(s) used in the network product, and operating system documentation.
- The tester verifies that entries in the list "A" require successful authentication for all users without exception, on basis of the user name and at least one authentication attribute.
- The tester logs into the network product and verifies that list "B" is accurate, based on his expert knowledge of the operating system(s) used in the network product, and operating system documentation. Unix® example: To list files with SUID and SGID permissions, the following commands can be used:
SUID: find / -perm -4000 -type f -exec ls {} \; > suid_files.txt
SGID: find / -perm -2000 -type f -exec ls {} \; > sgid_files.txt
- The tester verifies that file entries in the list "B" do not have write permissions for anyone else than the owner.

- The tester verifies that entries in the list "B" only allow execution of specifically limited tasks which are needed on this network product, based on his expert knowledge of the operating system(s) used in the network product, and operating system documentation.
- The tester logs into the network product and tests for every entry in the list "B" that it does not provide a means to execute arbitrary functions with administrator/root privileges, e.g. via a shell escape.

9. **Expected Results:**

- The network product does not allow a user to gain administrator/root privileges from another user account without re-authentication.
- If a network product provides functions and files which execute specifically limited tasks automatically with higher privileges, it ensures that these limits cannot be bypassed.
- The system documentation about means for a user to gain administrator/root privileges from another user account accurately describes the network product.

10. **Expected Format of Evidence:** A test report provided by the accredited evaluator's test lab which will consist of the following information:

- Documentation provided by the vendor: lists "A" and "B"
- Description of executed tests and commands
- Relevant output (e.g. screenshot or terminal log)
- Test result (passed or not passed)

11. **Test Execution:**

➤ **Test Case Number: 01**

a. Test Case Name: TC1_AUTHENTICATED_PRIVILEGE_ESCALATION_ONLY

b. Test Case Description: Tests need to be conducted to ensure that no privilege escalation method in interactive sessions (CLI or GUI) allows a user to gain administrator/root privileges from another user account without re-authentication.

c. Execution Steps:

- Identify the interactive sessions available in the system. Use the w command to display information about logged-in users and their processes.
- Attempt to gain administrator/root privileges from another user account without re-authentication. Use the sudo command to attempt to run a command that requires elevated privileges without entering the password of the target account.

d. Test Observation:

- **Case 1:** Verify that users are not able to gain administrator/root privileges from another user account without re-authentication.
Attempt to use insecure privilege escalation methods. Use the identified methods to attempt to gain elevated privileges without entering the required authentication information.

```
priyansha@priyansha:~$ su jane
Password:
$ sudo -l
[sudo] password for jane:
Sorry, user jane may not run sudo on priyansha.
$
```

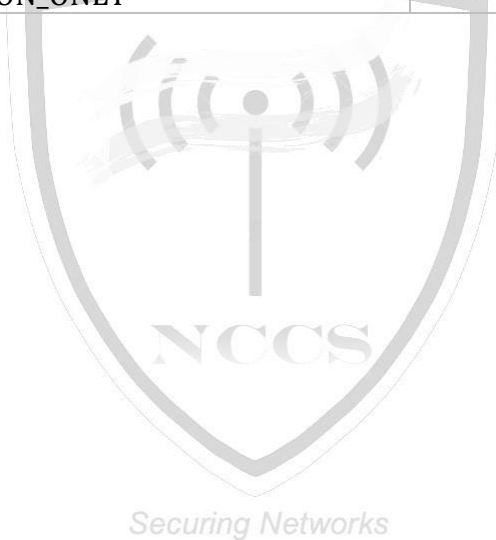
e. Evidence Provided:

A testing report which will consist of the following information:

- Screenshot of the output of the terminal of the PC through which DUT is logins.

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_AUTHENTICATED_PRIVILEGE_ESCALATION_ONLY		



2.10.4 TSTP for Evaluation of System account identification

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.4
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 10: Operating System
2. **<Security Requirement No & Name >** 2.10.4 System account identification
3. **<Requirement Description: >** Each system account in SMF shall have a unique identification.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.4.2.2]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use the command line interface to find OS name and version in Linux.
- Use command to get Application No/Version.

Use the following command, to display the information about the operating system release on a Unix/Linux system. **cat /etc/os-release**

```
supriya@supriya-HP-Laptop-15g-br0xx:~$ cat /etc/os-release
NAME="Ubuntu"
VERSION="20.04.4 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.4 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
supriya@supriya-HP-Laptop-15g-br0xx:~$
```

Check SMF version -

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

5. **DUT Configuration:** UNIX/ all major UNIX-like derivatives, including Linux is used on the Network product.

- **To get the hash of configuration file if the file is a ASCII text file.**

- Command - `sha256sum DUT_config.conf`

- **Digest Hash of Tested Configuration:**

- DUT_config.conf:

19be54b42975f7f414c53a280373aa9466398781ecd77a2e9d090dbc6dbfdd8

- **To get the hash of OS if using docker**

- Command - `docker images --digests`

- **Digest Hash of OS:**

- DUT_IMAGE:

fd363f0e0d146c869d60649bc4f42e7008829d9d67a5dfdaf3b38ad24af7d53a

6. **Preconditions:** UNIX/ UNIX like OS is used on the Network product.

7. **Test Objective:** To verify that UNIX account UIDs are assigned uniquely.

8. **Test Plan:**

8.1 **Number of test scenarios/test cases: 1**

8.2 **Tools used:** Command line of network function.

8.3 **Test Execution:**

- Create several UNIX accounts.
- Check UIDs of created accounts and of existing system accounts and the root account.

9. **Expected Results for Pass:**

- **Case 1:** All the accounts should have unique UID.

10. **Expected Format of Evidence:** Screenshot of terminal showing UID's of all the system accounts in the network functions.

11. **Test Execution:**

- **Test Case Number: 1**

a. Test Case Name: TC_UNIQUE_SYSTEM_ACCOUNT_IDENTIFICATION

b. Test Case Description: To verify that UNIX account UIDs are assigned uniquely.

c. Execution Steps:

1. Create several UNIX accounts.

Command - **sudo adduser <userNameHere>** (Where userNameHere is desired user account name)

2. Check UIDs of created accounts and of existing system accounts and the root account.

Command - **id userNameHere**

d. Test Observations: The UIDs are all different and only the root account has UID = 0.

• **Case 1: UIDs are unique (OK CASE)**

A. The below figure is from the PC through which the Unix Account is created, and each is assigned a unique UID.

```
supriya@supriya-HP-Laptop-15g-br0xx:~$ id user2
uid=1002(user2) gid=1004(user2) groups=1004(user2)
supriya@supriya-HP-Laptop-15g-br0xx:~$ id user1
uid=1001(user1) gid=1001(user1) groups=1001(user1)
supriya@supriya-HP-Laptop-15g-br0xx:~$
```

Create a script that will check that UID's of all the accounts is unique. Following python script, "check_uid_duplicates.py" iterates the file /etc/passwd and checks that all the accounts have unique UID.

Run the script as a superuser (root) to access the /etc/passwd file:

```
sudo ./check_uid_duplicates.py
```

Negative Output (No Duplicate UIDs Found):

```
No duplicate UIDs found.
```

B. Check that UID for root is 0.

command - **cat /etc/passwd**

Third parameter corresponding to each account represents the UID of user.

```

supriya@supriya-HP-Laptop-15g-br0xx:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mail Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/netif:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nologin
syslog:x:102:106:/:/home/syslog:/usr/sbin/nologin
messagebus:x:103:107:/:/nonexistent:/usr/sbin/nologin
_apt:x:104:65534:/:/nonexistent:/usr/sbin/nologin
uuidd:x:105:111:/:/run/uuidd:/usr/sbin/nologin
avahi-autoipd:x:106:112:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
usbmux:x:107:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
dnsmasq:x:108:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
rtkit:x:109:114:RealtimeKit,,,:/proc:/usr/sbin/nologin
speech-dispatcher:x:110:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false
whoopsie:x:111:117:/:/nonexistent:/bin/false
kernoops:x:112:65534:Kernel Oops Tracking Daemon,,,:/usr/sbin/nologin
saned:x:113:119:/:/var/lib/saned:/usr/sbin/nologin
pulse:x:114:120:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin
avahi:x:115:122:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin

```

- **Case 2: UIDs are not unique (NOT OK CASE)**

If the UID of any two accounts is not different, then this test case fails.

```

Duplicate UIDs found:
UID: 1001
UID: 1003
UID: 1005

```

e. Evidence Provided:

Securing Networks

A testing report which will consist of the following information:

- Screenshot of the result of script. The UIDs are all different for users and only the root account has UID= 0.

12. Test Case Result:

SL. No	OUTCOME OF RUNNING THE SCRIPT (CASE 1/CASE2)	PASS/FAIL	Remarks
1	TC_UNIQUE_SYSTEM_ACCOUNT_ID ENTIFICATION		

2.10.5 TSTP for Evaluation of OS Hardening - Minimized kernel network functions

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.5
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name:>** Section 10: Operating System
2. **<Security Requirement No & Name:>** 2.10.5 OS Hardening - Minimized kernel network functions
3. **<Requirement Description: >**Kernel-based network functions not needed for the operation of the network element shall be deactivated. In particular, the following ones shall be disabled by default:
 - IP Packet Forwarding between different interfaces of the network product.
 - Proxy ARP
 - Directed broadcast
 - IPv4 Multicast handling
 - Gratuitous ARP messages

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section - 4.3.3.1.2]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration: Network interface

```

unnati@VM2:~$ ifconfig
enp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.101.136 netmask 255.255.255.0 broadcast 192.168.101.255
    inet6 fe80::971a:b291:a403:35f8 prefixlen 64 scopeid 0x20<link>
    ether 52:54:00:3e:48:4f txqueuelen 1000 (Ethernet)
    RX packets 1945 bytes 860156 (860.1 KB)
    RX errors 0 dropped 156 overruns 0 frame 0
    TX packets 1523 bytes 1596238 (1.5 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp6s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.102.214 netmask 255.255.255.0 broadcast 192.168.102.255
    inet6 fe80::e2af:32aa:1e97:b5b0 prefixlen 64 scopeid 0x20<link>
    ether 52:54:00:6b:d4:3a txqueuelen 1000 (Ethernet)
    RX packets 433 bytes 52474 (52.4 KB)
    RX errors 0 dropped 170 overruns 0 frame 0
    TX packets 184 bytes 18942 (18.9 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 269 bytes 23555 (23.5 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 269 bytes 23555 (23.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

unnati@VM2:~$

```

Host A configuration

```

Rhythmbox ~$ ifconfig
enp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.101.144 netmask 255.255.255.0 broadcast 192.168.101.255
    inet6 fe80::b098:925c:bc59:de8d prefixlen 64 scopeid 0x20<link>
    ether 52:54:00:cd:29:41 txqueuelen 1000 (Ethernet)
    RX packets 5261 bytes 752223 (752.2 KB)
    RX errors 0 dropped 3090 overruns 0 frame 0
    TX packets 6422 bytes 616253 (616.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 257 bytes 23081 (23.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 257 bytes 23081 (23.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

unnati@VM1:~$

```

Host B configuration

Securing Networks

```

Thunderbird Mail: ifconfig
eth0: flags=4099<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.102.215 netmask 255.255.255.0 broadcast 192.168.102.255
    inet6 fe80::ed88:6064:b5a7:b1d1 prefixlen 64 scopeid 0x20<link>
    ether 52:54:00:5d:5a:74 txqueuelen 1000 (Ethernet)
    RX packets 212042 bytes 452954801 (452.9 MB)
    RX errors 0 dropped 5694 overruns 0 frame 0
    TX packets 96384 bytes 7062162 (7.0 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 415 bytes 37067 (37.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 415 bytes 37067 (37.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

unnati@VM3:~$

```

- To add default gateway at host A, use the command
route add default gw <interface 1 IP>

```

root@VM1:/home/unnati# route add default gw 192.168.101.136
root@VM1:/home/unnati# route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
0.0.0.0 192.168.101.136 0.0.0.0 UG 0 0 0 enp1s0
0.0.0.0 192.168.101.1 0.0.0.0 UG 100 0 0 enp1s0
169.254.0.0 0.0.0.0 255.255.0.0 U 1000 0 0 enp1s0
192.168.101.0 0.0.0.0 255.255.255.0 U 100 0 0 enp1s0
root@VM1:/home/unnati#

```

- To add default gateway at host B, use the command
route add default gw <interface 2 IP>

```

root@VM3:/home/unnati# route add default gw 192.168.102.214
root@VM3:/home/unnati# route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
0.0.0.0 192.168.102.214 0.0.0.0 UG 0 0 0 enp1s0
0.0.0.0 192.168.102.1 0.0.0.0 UG 20100 0 0 enp1s0
169.254.0.0 0.0.0.0 255.255.0.0 U 1000 0 0 enp1s0
192.168.102.0 0.0.0.0 255.255.255.0 U 100 0 0 enp1s0
root@VM3:/home/unnati#

```

To get the hash of configuration file if the file is a ASCII text file

Command used: ***sha256sum SMF_config.conf*** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf

```

To get the hash of OS if using docker

Command used: ***docker images --digests*** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

6. Preconditions:

- SMF should have 2 different physical or logical ethernet interfaces.

- Host 1 is connected to Interface 1 on subnet A and Host 2 is connected to Interface 2 on subnet B.
- A network traffic analyzer on the network product (e.g., TCPDUMP) or an external traffic analyzer directly connected to the network product is available.

Note: A network analyzer tool needs to be installed on the DUT. TCPDUMP, Wireshark or a similar type of tool can be installed. Here, Wireshark is used. Steps to install and configure the tool should be given in the documentation provided.

7. **Test Objective:** Verify that all the given options need to be disabled,

- IP Packet Forwarding between different interfaces of the network product.
- Proxy ARP
- Directed broadcast
- IPv4 Multicast handling
- Gratuitous ARP messages

8. **Test Plan:**

8.1 **Number of Test Scenarios: 5**

8.1.1 **Test Scenario for IP forward**

This test scenario verifies that the IP Packet Forwarding is disabled by default in SMF.

8.1.2 **Test Scenario for Proxy ARP**

This test scenario verifies that the Proxy ARP is disabled by default in SMF.

8.1.3 **Test Scenario for Directed broadcasting**

This test scenario verifies that the Directed broadcast is disabled by default in SMF.

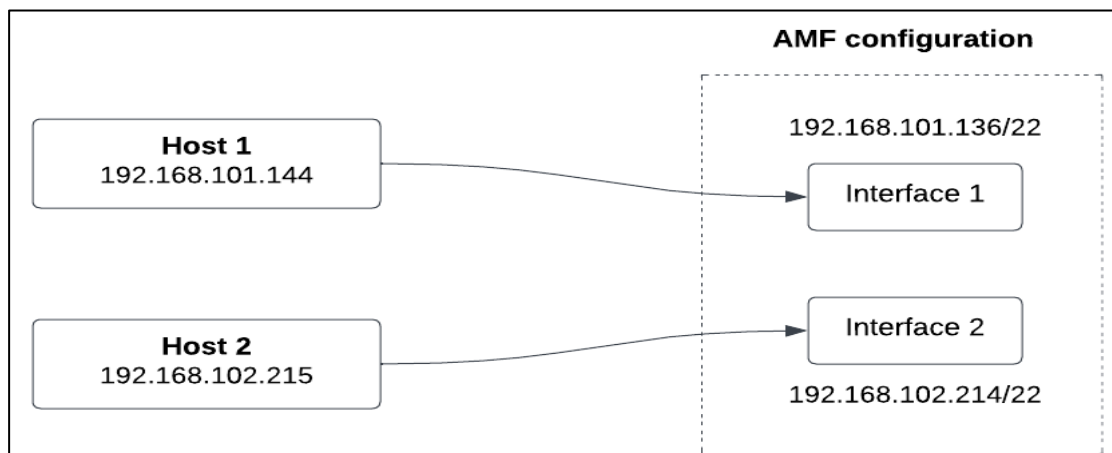
8.1.4 **Test Scenario for Multicast handling**

This test scenario verifies that the IP Multicast is disabled by default in SMF.

8.1.5 **Test Scenario for Gratuitous ARP**

This test scenario verifies that the Gratuitous ARP is disabled by default in SMF.

8.2 **Test Setup Diagram:**



8.3 **Tools Used:** Wireshark

8.4 **Test Execution Steps:**

- Login to the network product
- Follow the preconditions given
- For sending packets from host A to host B through SMF, make the default gateway of host A as interface 1 of SMF.
- To delete default gateway: **sudo route del default**
- To add gateway: **sudo route add default gw 192.168.101.136**
- For each of the test cases, make sure the functionality is disabled by a command or by editing a configuration file.
- Send packet from host A to host B via SMF interface and capture the network traffic through Wireshark.
- Verify that the packet is correctly captured by network product but not received by host B.

Note: The functionality should be disabled by default. If the functionality is not disabled by default, steps should be provided to disable it in the OEM document.

9. **Expected Results for Pass:**

- Case 1: The packet is not routed by the network product and Host 2 does not receive it.
- Case 2: No Arp Reply is received by Host 1.
- Case 3: The packet is not broadcast by the network product and Host 2 cannot receive it.
- Case 4: No interface is running multicast protocols
- Case 5: The network product ARP Cache is not updated.

10. **Expected Format of Evidence:** Screenshots of the PCAP trace of the received packets for each of the cases.

Securing Networks

11. **Test Execution:**

➤ **Test Case Number:** 01

- a. **Test Case Name:** TC1_IP_FWD_DISABLING
- b. **Test Case Description:** Verify that IP Packet Forwarding is disabled by default on the network product. This test case verifies that a packet received by a network product interface but directed to a host on a different network is not routed by the network product.
- c. **Execution Steps:**
Pinging from host A to host B


```

root@VM1:/home/unnati# ping 192.168.102.215
PING 192.168.102.215 (192.168.102.215) 56(84) bytes of data.
^C
--- 192.168.102.215 ping statistics ---
18 packets transmitted, 0 received, 100% packet loss, time 17392ms

root@VM1:/home/unnati#

```

Wireshark capture showing packets are captured at interface 1 but could not reach host B.

No.	Time	Source	Destination	Protocol	Length	Info
72	30.026051870	192.168.101.144	192.168.102.215	ICMP	100	Echo (ping) request id=0x000f, seq=15/3840, ttl=64 (no response found!)
74	31.040003069	192.168.101.144	192.168.102.215	ICMP	100	Echo (ping) request id=0x000f, seq=16/4096, ttl=64 (no response found!)

Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT) Total Length: 84 Identification: 0x8f55 (36693) Flags: 0x40, Don't fragment ...0 0000 0000 0000 = Fragment Offset: 0 Time to Live: 64 Protocol: ICMP (1) Header Checksum: 0x5d9b [validation disabled] [Header checksum status: Unverified] Source Address: 192.168.101.144 Destination Address: 192.168.102.215	Internet Control Message Protocol Type: 8 (Echo (ping) request) Code: 0 Checksum: 0xdea2 [correct] [Checksum Status: Good] Identifier (BE): 15 (0x000f) Identifier (LE): 3840 (0x0f00) Sequence Number (BE): 8 (0x0008) Sequence Number (LE): 2048 (0x0800)
[No response seen] Timestamp from icmp data: Jul 8, 2022 15:19:13.000000000 IST [Timestamp from icmp data (relative): -1.317939905 seconds]	

d. Test Observations: The DUT should ensure that the IP forwarding is disabled.

➤ Test Case Number: 02

a. **Test Case Name:** TC2_PROXY_ARP_DISABLING

b. **Test Case Description:** Verify that the Proxy ARP feature is disabled by default on the network product. This test case verifies that the network product does not respond to ARP requests intended for another host.

c. **Execution Steps:**

Sending ARP packets from host A to host B. Here we can see no reply coming from host B.

```

root@VM1:/home/unnati# arpsend -D -e 192.168.101.136 enp1s0
arpsend: 192.168.101.136 is detected on another computer : 52:54:00:3e:48:4f
root@VM1:/home/unnati# arpsend -D -e 192.168.102.214 enp1s0
arpsend: 192.168.102.214 is detected on another computer : 52:54:00:3e:48:4f
root@VM1:/home/unnati# arpsend -D -e 192.168.102.215 enp1s0

```

Wireshark capture showing no reply coming from host B

arp						
No.	Time	Source	Destination	Protocol	Length	Info
114	21.072717022	RealtekU_cd:29:41		ARP	62	Who has 192.168.102.215? Tell 192.168.101.144
115	22.072683599	RealtekU_cd:29:41		ARP	62	Who has 192.168.102.215? Tell 192.168.101.144
120	23.072800078	RealtekU_cd:29:41		ARP	62	Who has 192.168.102.215? Tell 192.168.101.144
123	24.073014297	RealtekU_cd:29:41		ARP	62	Who has 192.168.102.215? Tell 192.168.101.144
134	29.297124642	RealtekU_cd:29:41		ARP	62	Who has 192.168.101.136? Tell 192.168.101.144
135	29.297155056	RealtekU_3e:48:4f		ARP	44	192.168.101.136 is at 52:54:00:3e:48:4f
142	33.161073995	RealtekU_cd:29:41		ARP	62	Who has 192.168.102.214? Tell 192.168.101.144
143	33.161096381	RealtekU_3e:48:4f		ARP	44	192.168.102.214 is at 52:54:00:3e:48:4f
177	36.882066845	RealtekU_cd:29:41		ARP	62	Who has 192.168.102.215? Tell 192.168.101.144

d. Test Observations:

The DUT should ensure that the Proxy ARP is disabled.

➤ **Test Case Number: 03**

- a. **Test Case Name:** TC3_DIRECTED_BROAD_DISABLING
- b. **Test Case Description:** Verify that the Directed broadcast is disabled by default on the network product. This test case verifies that a packet received by a network product whose destination address is a valid broadcast address is dropped.
- c. **Execution Steps:**

Broadcasting packets from host A to host B

```
root@VM1:/home/unnati# ping -b 192.168.102.255
PING 192.168.102.255 (192.168.102.255) 56(84) bytes of data.
^C
--- 192.168.102.255 ping statistics ---
27 packets transmitted, 0 received, 100% packet loss, time 26625ms

root@VM1:/home/unnati#
```

Wireshark capture

102.41.309370000	192.168.101.144	192.168.102.255	ICMP	100 Echo (ping) request	id=0x0010, seq=34/8704, ttl=64 (no response found!)
105.42.333315706	192.168.101.144	192.168.102.255	ICMP	100 Echo (ping) request	id=0x0010, seq=35/8960, ttl=64 (no response found!)
106.43.357303877	192.168.101.144	192.168.102.255	ICMP	100 Echo (ping) request	id=0x0010, seq=36/9216, ttl=64 (no response found!)

Frame 9: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 192.168.101.144, Dst: 192.168.102.255
Internet Control Message Protocol
Type: 8 (Echo (ping) request)
Code: 0
Checksum: 0x00e0 [correct]
[Checksum Status: Good]
Identifier (BE): 16 (0x0010)
Identifier (LE): 4096 (0x1000)
Sequence Number (BE): 1 (0x0001)
Sequence Number (LE): 256 (0x0100)
[No response seen]
Timestamp from loop data: Jul 13, 2022 23:20:08.060000000 IST
[Timestamp from loop data (relative): 0.306348916 seconds]

- d. **Test Observations:** The DUT should ensure that the Directed Broadcasting is disabled.

➤ **Test Case Number: 04**

- a. **Test Case Name:** TC4_IP_MULTICAST_HANDLING
- b. **Test Case Description:** Verify that IP Multicast is disabled by default on the network product. This test case verifies that packets with IP source or destination address belonging to the multicast IP ranges (224.0.0.0 through 239.255.255.255) are not handled by the network product.
- c. **Execution Steps:**

We can check here that multicast is not there.

```
root@VM2:/home/unnati# ifconfig enp1s0
enp1s0: flags=67<UP,BROADCAST,RUNNING> mtu 1500
    inet 192.168.101.136 netmask 255.255.255.0 broadcast 192.168.101.255
    inet6 fe80::971a:b291:a403:35f8 prefixlen 64 scopeid 0x20<link>
    ether 52:54:00:3e:48:4f txqueuelen 1000 (Ethernet)
    RX packets 16760 bytes 29691736 (29.6 MB)
    RX errors 0 dropped 1404 overruns 0 frame 0
    TX packets 9367 bytes 1551638 (1.5 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

d. **Test Observations:** The DUT should ensure that the Multicast Handling is disabled.

➤ **Test Case Number:** 05

a. **Test Case Name:** TC5_GRATUITOUS_ARP_DISABLING

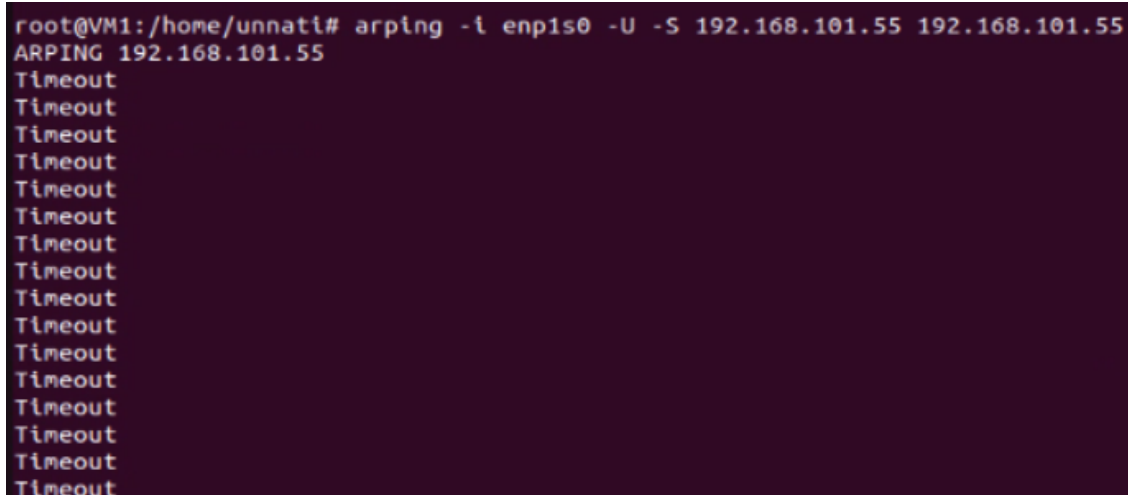
b. **Test Case Description:** Verify that the Gratuitous ARP feature is disabled by default on the network product. This test case verifies that the network product cannot send gratuitous ARP requests and that the network product discards incoming Gratuitous ARP requests.

c. **Execution Steps:**

Note: Send a Gratuitous ARP request and reply from Host 1, i.e., an ARP request where the source and destination IP are both set to an IP address different from the one already cached in the network product ARP Cache for Host 1 and the destination MAC is the broadcast address ff:ff:ff:ff:f

To send ARP request:

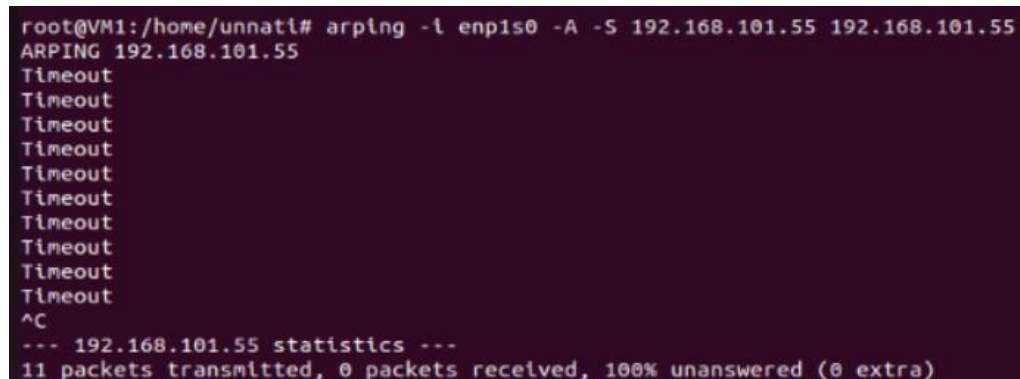
arping -i enp1s0 -U -S 192.168.101.55 192.168.101.55



```
root@VM1:/home/unnati# arping -i enp1s0 -U -S 192.168.101.55 192.168.101.55
ARPING 192.168.101.55
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
```

● To send ARP response:

arping -i enp1s0 -A -S 192.168.101.55 192.168.101.55



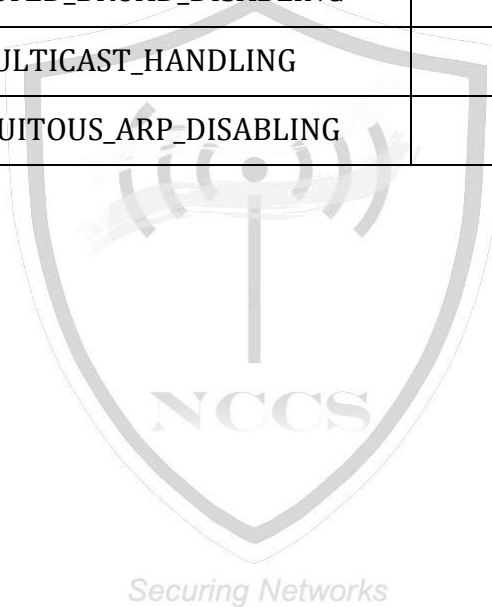
```
root@VM1:/home/unnati# arping -i enp1s0 -A -S 192.168.101.55 192.168.101.55
ARPING 192.168.101.55
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
Timeout
^C
--- 192.168.101.55 statistics ---
11 packets transmitted, 0 packets received, 100% unanswered (0 extra)
```

```
root@VM2:/home/unnati# arp -a
unnati (192.168.102.1) at 52:54:00:bb:29:56 [ether] on enp6s0
? (192.168.101.144) at 52:54:00:cd:29:41 [ether] PERM on enp1s0
```

d. Test Observations: The DUT should ensure that the Gratuitous ARP is disabled.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_HANDLING_OF_ICMP		
2	TC2_PROXY_ARP_DISABLING		
3	TC3_DIRECTED_BROAD_DISABLING		
4	TC4_IP_MULTICAST_HANDLING		
5	TC5_GRATUITOUS_ARP_DISABLING		



2.10.6 TSTP for Evaluation of No automatic launch of removable media

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.6
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **ITSAR Section No & Name:** Section 10: Operating System

2. **Security Requirement No & Name:** 2.10.6 No automatic launch of removable media

3. **Requirement Description:** SMF shall not automatically launch any application when a removable media device is connected.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section - 4.3.3.1.3]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

- Command used: **lsusb** (To list all the information about the USB bus in the system.)

```
vm2@vm2:~$ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 002: ID 0627:0001 Adomax Technology Co., Ltd QEMU USB Tablet
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
vm2@vm2:~$
```


- Use the command “**usb-devices**”. This command provides detailed information about the USB devices, such as their vendor ID, product ID, device speed, device class, and configuration settings.

```

vn2@vn2: ~$ usb-devices
T:  Bus=01 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=480 MxCh=15
D:  Ver= 2.00 Cls=09(hub ) Sub=00 Prot=01 MxPS=64 #Cfgs=  1
P:  Vendor=1d6b ProdID=0002 Rev=05.19
S:  Manufacturer=Linux 5.19.0-45-generic xhci-hcd
S:  Product=xHCI Host Controller
S:  SerialNumber=0000:02:00:0
C:  #Ifs= 1 Cfg#= 1 Atr=e0 MxPwr=0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub ) Sub=00 Prot=00 Driver=hub
E:  Ad=81(I) Atr=03(Int.) MxPS=  4 IvL=256ms

T:  Bus=01 Lev=01 Prnt=01 Port=00 Cnt=01 Dev#=  2 Spd=480 MxCh= 0
D:  Ver= 2.00 Cls=00(>ifc ) Sub=00 Prot=00 MxPS=64 #Cfgs=  1
P:  Vendor=0627 ProdID=0001 Rev=00.00
S:  Manufacturer=QEMU
S:  Product=QEMU USB Tablet
S:  SerialNumber=28754-0000:00:02:1:00:0-1
C:  #Ifs= 1 Cfg#= 1 Atr=a0 MxPwr=100mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=03(HID ) Sub=00 Prot=00 Driver=usbhid
E:  Ad=81(I) Atr=03(Int.) MxPS=  8 IvL=1ms

T:  Bus=02 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=5000 MxCh=15
D:  Ver= 3.00 Cls=09(hub ) Sub=00 Prot=03 MxPS= 9 #Cfgs=  1
P:  Vendor=1d6b ProdID=0003 Rev=05.19
S:  Manufacturer=Linux 5.19.0-45-generic xhci-hcd
S:  Product=xHCI Host Controller
S:  SerialNumber=0000:02:00:0
C:  #Ifs= 1 Cfg#= 1 Atr=e0 MxPwr=0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub ) Sub=00 Prot=00 Driver=hub
E:  Ad=81(I) Atr=03(Int.) MxPS=  4 IvL=256ms

```

Details of USB:-

- **T:** Describes the overall topology of the USB device, including the bus number, device level, and parent device.
- **D:** Describes the configuration of the USB device, including its device class, subclass, and protocol.
- **P:** Describes the physical properties of the USB device, such as its vendor ID, product ID, and revision number.
- **The first S** line provides the manufacturer name for the USB device.
- **The second S** line provides the product name for the USB device.
- **C:** Describes the available configurations for the device
- **I:** Describes the available interfaces.
- **E:** Describes the available endpoints.
- **Command to check all the removable devices connected: - lsblk**
It will display all the connected devices, type will tell which type of device it is

```

vm2@vm2:~$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0       7:0      0    4K  1 loop /snap/bare/5
loop1       7:1      0   63.4M 1 loop /snap/core20/1950
loop2       7:2      0   73.8M 1 loop /snap/core22/750
loop3       7:3      0   63.5M 1 loop /snap/core20/1891
loop4       7:4      0   73.9M 1 loop /snap/core22/766
loop5       7:5      0    6.4M 1 loop /snap/curl/1754
loop6       7:6      0  244.8M 1 loop /snap/firefox/2760
loop7       7:7      0  244.5M 1 loop /snap/firefox/2800
loop8       7:8      0  346.3M 1 loop /snap/gnome-3-38-2004/119
loop9       7:9      0  349.7M 1 loop /snap/gnome-3-38-2004/140
loop10      7:10     0  460.7M 1 loop /snap/gnome-42-2204/105
loop11      7:11     0  466.5M 1 loop /snap/gnome-42-2204/111
loop12      7:12     0   91.7M 1 loop /snap/gtk-common-themes/1535
loop13      7:13     0   45.9M 1 loop /snap/snap-store/638
loop14      7:14     0   12.3M 1 loop /snap/snap-store/959
loop15      7:15     0   53.3M 1 loop /snap/snapd/19457
loop16      7:16     0   304K  1 loop /snap/snapd-desktop-integration/49
loop17      7:17     0   53.3M 1 loop /snap/snapd/19361
sr0         11:0     1  1024M  0 rom
vda         252:0     0    20G  0 disk
├─vda1      252:1     0     1M  0 part
├─vda2      252:2     0   513M  0 part /boot/efi
└─vda3      252:3     0   19.5G  0 part /var/snap/firefox/common/host-hunspell

```

- Autorun option should be disabled, checking if autoplay is enabled or not

```

vm2@vm2:~$ gsettings get org.gnome.desktop.media-handling autorun-never
true

```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2eda1d08af1844429d30  AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

6. Preconditions:

- If the network product is provisioned with the necessary physical ports/drives (CD/DVD drive, USB port, etc.) then the test case applies.
- The tester should have sudor access.
- The tester should have a removable media which is executable for the OS.

Note: To make the removable device executable it will depend on the OS on which DUT is running.

7. **Test Objective:** To verify that the network product does not launch any applications automatically when a removable media device is connected.

8. Test Plan:

8.1 Test Bed Diagram:

8.2 Tools Required:

- Removable media disk
- Command line

8.3 **Test execution steps:**

- Check all the preconditions are met.
- Check if the media device is inserted into DUT.
 - Use the command: **lsusb**
- To check if autoplay of media is disabled:
 - Open terminal
 - Type the command “gsettings get org.gnome.desktop.media-**handling** autorun-never”
 - This command should show true as output.
 - After this when the removable media is inserted, it will not autorun automatically.
- To check if the media device is not mounted automatically
 - Use the command: **lsblk**
- The tester should insert the removable media into the system.
- There should not be any auto run of the contents of the media and the removable media is not mounted automatically.

9. **Expected Results for Pass:**

When the removable media is inserted, there shall not be automatic launch of contents of that media.

10. **Expected Format of Evidence:** Screenshots checking if auto play option is disabled.

11. **Test Execution:**

➤ **Test case Number: - 01**

a. **Test case name:**

TC1_EVALUATION_OF_NO_AUTOMATIC_LAUNCH_OF_REMOVABLE_MEDIA

b. **Test case description:** - the test case should ensure that if the removable device is inserted to the DUT, DUT should not automatically launch its content.

c. **Execution steps: -**

• **Positive test case:**

a. **Check if the removable device is inserted:**

Using command: **lsusb**

```
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 002: ID 0461:0010 Primax Electronics, Ltd HP PR1101U / Primax PMX-KPR1101U Keyboard
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 007: ID 03f0:1985 HP, Inc USB Flash Drive
Bus 001 Device 002: ID 046d:c077 Logitech, Inc. M105 Optical Mouse
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

- b. Open terminal and type the command:
gsettings get org.gnome.desktop.media-handling autorun-never

```
vm2@vm2: $ gsettings get org.gnome.desktop.media-handling autorun-never
true
```

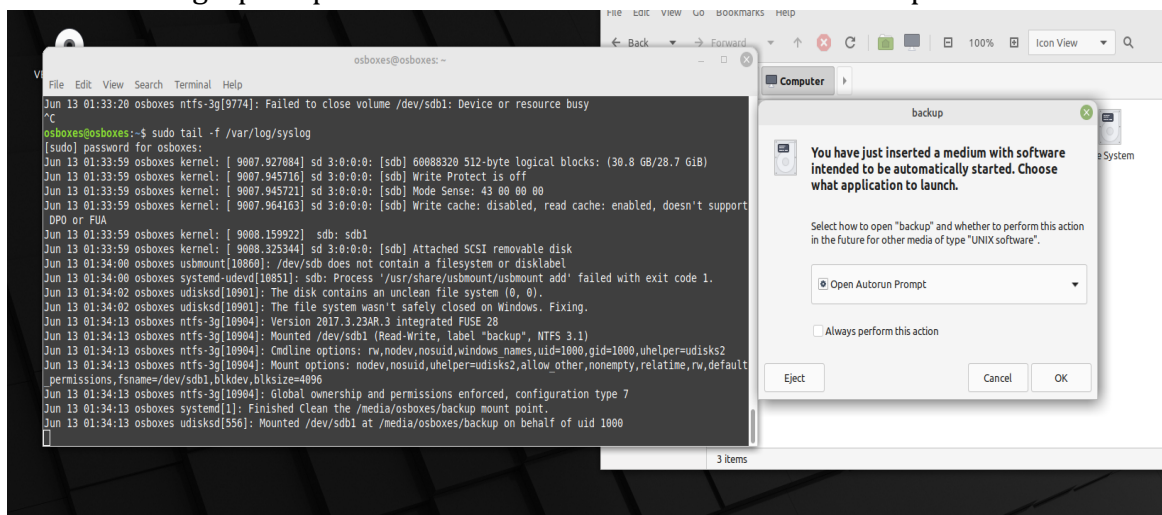
• **Negative Test Case:**

- a. For checking autoplay option

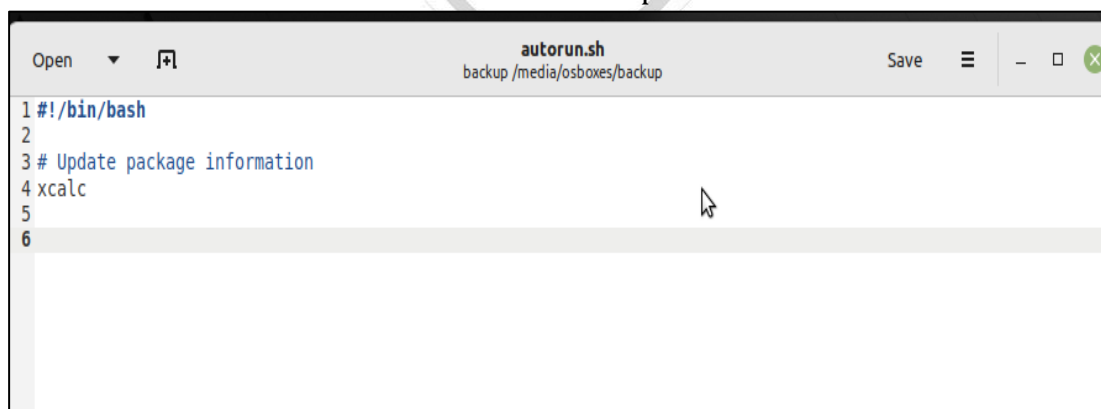
We can see false here it means autorun is true

```
vm2@vm2:~$ gsettings get org.gnome.desktop.media-handling autorun-never
false
```

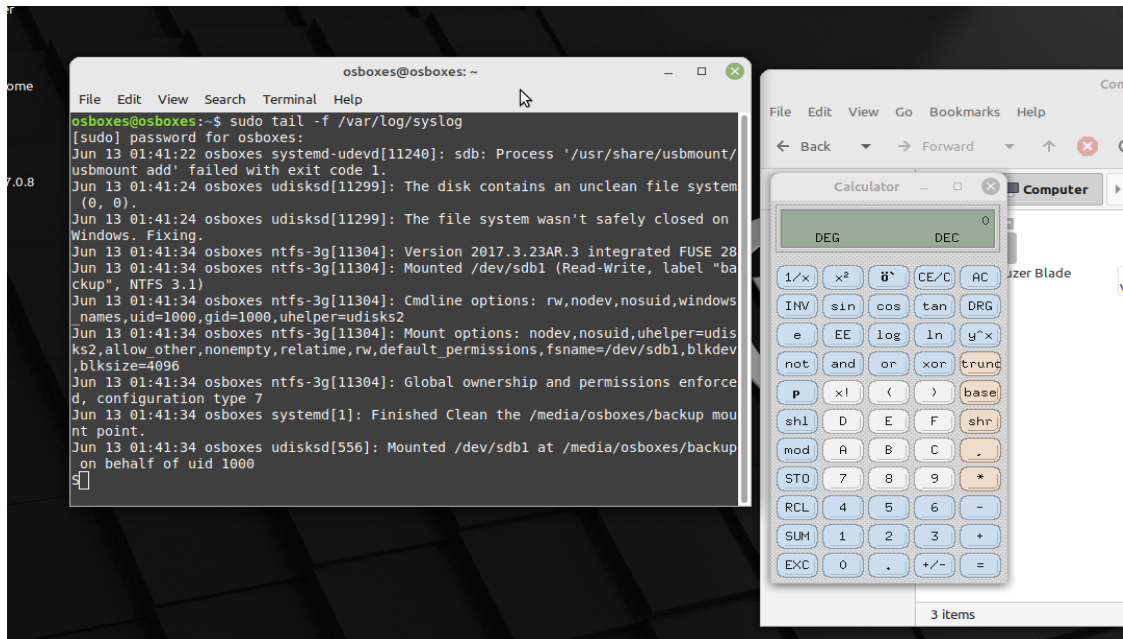
When autorun option is enabled and we insert a removable device with one autorun script. Here we are seeing a prompt that tells the device contains autorun script.



This is the **autorun.sh** file which will open a calculator.



Here we can see this shell script automatically opens a calculator.



d. Test observation:

- DUT meets test case 1 if the automatic launch of media is disabled and it is not mounted automatically.
- DUT does not meet test case 1 if the automatic launch of media is not disabled and it is mounted automatically when removable media is inserted.

Test Results of the Test requirement 2.10.6: PASS

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_EVALUATION_OF_NO_AUTOMATIC_LAUNCH_OF_REMOVABLE_MEDIA	PASS	

2.10.7 TSTP Report for Evaluation of Protection from buffer overflows under 5G ITSAR

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.7
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<TSTP Name:> TC_PROTECTION_FROM_BUFFER_OVERFLOWS

<Applicant Name:> Eg: XYZ

<Application Number>

<DUT Details: > Ex: Router

<TSTP Document ID:>

<DUT Make:> Ex: XYZ

< DUT Model no: > Ex: XYZ1234

<DUT Serial Number:>

<DUT Software Version:>

<Applicable ITSAR: >

< Version No:>

<OEM Supplied Document list: >

<Hyperlink to Portal uploaded documents>

1. **ITSAR Section No & Name:** Section 10: Operating System

2. **Security Requirement No & Name:** 2.10.7 Protection from buffer overflows

3. **Requirement Description:** SMF shall support mechanisms for buffer overflow protection.

Documentation which describes these buffer overflow mechanisms and also how to check that they have been enabled and/or implemented shall be provided by OEM.

[Reference: TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0 Section - 4.3.3.1.5]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use the command: **lscpu**


```

priyansha@priyansha:~$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:             Little Endian
Address sizes:          46 bits physical, 48 bits virtual
CPU(s):                12
On-line CPU(s) list:    0-11
Thread(s) per core:     2
Core(s) per socket:     6
Socket(s):              1
NUMA node(s):          1
Vendor ID:              GenuineIntel
CPU family:             6
Model:                 85
Model name:             Intel(R) Xeon(R) W-2133 CPU @ 3.60GHz
Stepping:              4
CPU MHz:               1800.000
CPU max MHz:           3900.0000
CPU min MHz:           1200.0000
BogoMIPS:              7200.00
Virtualization:         VT-x
L1d cache:             192 KiB
L1i cache:             192 KiB
L2 cache:              8 MiB
L3 cache:              8.0 MiB
NUMA node0 CPU(s):     0-11
Vulnerability Itlb multihit: KVM: Mitigation: VMX disabled
Vulnerability L1tf:       Mitigation: PTE Inversion; VMX conditional cache flushes, SMT vulnerable
Vulnerability Mds:        Mitigation: Clear CPU buffers; SMT vulnerable
Vulnerability Meltdown:   Mitigation: PTI
Vulnerability Spec store bypass: Mitigation: Speculative Store Bypass disabled via prctl and seccomp
Vulnerability Spectre v1:  Mitigation: usercopy/swapgs barriers and __user pointer sanitization
Vulnerability Spectre v2:  Mitigation: Retpolines, IBPB conditional, IBRS_FW, STIBP conditional, RSB filling
Vulnerability Srbds:      Not affected
Vulnerability Tsx async abort: Mitigation: Clear CPU buffers; SMT vulnerable
Flags:                   fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch
                          _permon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid dca sse4_1 s
                          set_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb cat_l3 cdp_l3 invpcid_single pti intel_ppin ssbd mba lbrs
                          lbpb stibp tpr_shadow vmx flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm cqm mpx rdt_a avx512f avx512dq rdseed adx snap clfl
                          shopt clwb intel_pt avx512cd avx512bw avx512vl xsaveopt xsavec xgetbv1 xsaves cqm_llc cqm_occup_llc cqm_mbm_total cqm_mbm_local dtherm ida arat pln pts md_clear flush_l1d

```

5. DUT Configuration:

Configuration of DUT:

Command line

6. Test Plan:

a. **Test Case Description:** To ensure that the system supports mechanisms that protect against buffer overflow.

b. **Pre-Conditions:**

- A document which provides a detailed technical description of the system's buffer overflow protection mechanisms. If a standard buffer overflow mechanism from a 3rd party vendor is used then a reference to the standard feature in the 3rd party vendors documentation should be provided.
- Test results from a test execution phase of buffer overflow protection mechanism testing

7. Test Execution:

a. **Test Case Number:** 1

b. **Test Case Name:** TC1_PROTECTION_FROM_BUFFER_OVERFLOWS

c. **Test Case Description:** To ensure that the system supports mechanisms that protect against buffer overflow.

d. **Tools Used:**

Common line

e. **Execution Steps:**

The accredited evaluator's test lab is required to execute the following steps:

- ≠ Check that all pre-conditions are met.
- ≠ The tester verifies that there is:
 - A technical description of the buffer overflow protection mechanisms that have been implemented on the system.
 - Details of whether the buffer overflow protection mechanisms are implemented by default or if additional actions (e.g., scripts or commands manually executed) are required.
 - If manually executed actions are required, then detailed instructions should be included in the technical description.
- ≠ The tester verifies that the test results:
 - Describe test procedures used to verify the buffer overflow protection mechanisms,
 - Contain data which demonstrates/indicates that the buffer overflow protection mechanisms described in the technical description document has been implemented.
 - Contains details of the test set-up for the testing of the buffer overflow protection mechanisms. Where simulators and/or scripts are used to artificially create the conditions to trigger the buffer overflow protection mechanism then details of these should also be included.

f. Test Observations:

Documentation provided needs to be checked

8. Test Case Result:

SL.No	OUTCOME OF RUNNING THE SCRIPT (CASE 1/CASE2)	PASS/FAIL (Please see Note Below)
1	TC1_PROTECTION_FROM_BUFFER_OVERFLOW	

NOTE: CRITERIA FOR PASS/FAIL

- If Case 1 of the Test Observation occurs: FAIL
- If Case 2 of the Test Observation occurs: PASS

2.10.8 TSTP for Evaluation of External file system mount restrictions

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.8
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **ITSAR Section No & Name:** Section 10: Operating System

2. **Security Requirement No & Name:** 2.10.8 External file system mount restrictions

3. **Requirement Description:** If normal users are allowed to mount external file systems (attached locally or via the network), OS-level restrictions shall be set properly in SMF in order to prevent privilege escalation or extended access permissions due to the contents of the mounted file systems. OS-level restrictions shall apply to normal users against mount / use of removable media devices (e.g., USB drive, CD ROM etc.) for data transfer.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section - 4.3.3.1.6]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5. **DUT Configuration:** We can check the OS level restrictions are there or not in the file `/etc/fstab`

`sudo gedit /etc/fstab`

The options `nodev` and `nosuid` needs to be set.

The "nodev" option prevents the execution of device files (such as `/dev/sda`) from the filesystem, and the "nosuid" option prevents the execution of setuid programs (programs with the "s" bit set, which temporarily elevates privileges when run).

```

1 # /etc/fstab: static file system information.
2 #
3 # Use 'blkid' to print the universally unique identifier for a
4 # device; this may be used with UUID= as a more robust way to name devices
5 # that works even if disks are added and removed. See fstab(5).
6 #
7 # <file system> <mount point> <type> <options>          <dump> <pass>
8 # / was on /dev/vda3 during installation
9 UUID=9ce3b548-d81f-49ef-b549-5672081afc5d /             ext4      errors=remount-ro 0      1
10 # /boot/efi was on /dev/vda2 during installation
11 UUID=4A44-334B /boot/efi      vfat      umask=0077      0      1
12 /swapfile                                 none      swap          0      0
13 /dev/sda1 /mnt/data     ext4      defaults,nodev,nosuid 0      2
14

```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2dea1d08af1844429d30  AMF_Config.conf

```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

6. Preconditions:

- ✗ Tester has admin access to check and configure the external filesystem mount permissions in the OS.
- ✗ Tester has the username and password of a user in the network product that has external filesystem mount privileges.

7. **Test Objective:** To check whether the OS level restrictions are set properly when a user tries to run some privileged escalation method through the mounted file.

8. Test Plan:

8.1 Test Setup Diagram:



8.2 **Tools Used:** Command line

8.3 **Test Execution Steps**

Note: - the configuration of OS level parameters will change according to the OS we will use, so the execution steps will change accordingly.

- Check that all pre-conditions are met.
- The tester checks the OS level permissions.
- This is one example which will be visible when nodev and nosuid are set “/dev/sda1 /mnt/data ext4 defaults,user,nodev,nosuid 0 0”
- The tester tries to mount the files on the system using the command with some different machine: -

```
sudo sshfs -o nonempty unnati@192.168.101.144:/home/unnati/Desktop/test /mnt/test/
```

- Here the tester without root privileges will not be allowed to access or run any files. (This functionality is there in linux by default)

9. **Expected results for Pass:** The OS-level restrictions are set properly in order to prevent privilege escalation or extended access permissions due to the contents of the mounted file systems. Any privilege escalation method used by the tester should be blocked.

10. **Expected Format of Evidence:** Screenshot containing the configuration file showing that OS-level restrictions are set properly for users that are allowed to mount external file systems

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC1_EXTERNAL_FILE_SYSTEM_MOUNT_RESTRICTIONS

b. **Test Case Description:** To check whether the OS level restrictions are set properly when a user tries to run some privileged escalation method through the mounted file.

c. **Execution Steps:** Success Case

- The tester mounts the files to the system.

```
unnati@unnati:~$ sudo sshfs -o nonempty unnati@192.168.101.144:/home/unnati/Desktop/test /mnt/test/
[sudo] password for unnati:
unnati@192.168.101.144's password:
unnati@unnati:~$
```

- The tester goes to the /mnt folder and tries to access the test folder.
- Here we can see OS level restrictions are already set and the tester cannot access the mounted folder.


```
unnati@unnati:~$ cd ../../
unnati@unnati:/$ cd mnt
unnati@unnati:/mnt$ cd test
bash: cd: test: Permission denied
unnati@unnati:/mnt$
```

- Only the root has the access, and the root can go to the folder and run these scripts.

```
unnati@unnati:/mnt$ sudo su
root@unnati:/mnt# cd test
root@unnati:/mnt/test# ls
test test1 test1.c test.sh
root@unnati:/mnt/test#
```

d. **Test Observations:** The testcase will pass when the OS level restrictions are set for any normal user trying to mount.

12. Test Case Result:

SL. No	OUTCOME OF RUNNING THE SCRIPT (CASE 1/CASE2)	PASS/FAIL	Remarks
1	TC1_EXTERNAL_FILE_SYSTEM_MOUNT_RESTRICTIONS		

NCCS
Securing Networks

2.10.9 TSTP for Evaluation of File-system Authorization Privileges

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.9
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 10 - Operating System
2. **<Security Requirement No & Name >** 2.10.9 File-system Authorization Privileges.
3. **<Requirement Description: >**SMF shall be designed to ensure that only users that are authorized to modify files, data, directories or file systems have the necessary privileges to do so.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section - 4.3.2.7]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:** A document describing the access control policy configured for the file systems for each user in the network product shall be provided by the vendor.
7. **Test Objective:** Verify that only users that are authorized to modify files, data, directories or file systems have the necessary privileges to do so.

8. **Test Plan:**

8.1 **Test Setup Diagram:**



8.2 **Tools Used:** NULL

8.3 **Test Execution Steps:**

1. The tester checks that OS-level permissions are configured correctly for users that are authorized to modify files, data, directories, or file systems on the system.
2. The tester tries to modify the files and directories for which the user has the necessary privileges. (Case 1)
3. The tester tries to modify the files and directories for which the user doesn't have the necessary privileges (Case 2)

9. **Expected results for Pass:**

- The modification to files and directories is allowed (Case 1)
- The modification to files and directories is denied (case 2)

10. **Expected Format of Evidence:** Screenshot of terminal/GUI containing the outcome of the performed operations on the files and directories.

11. **Test Execution:**

a. **Test Case Number:** 1

b. **Test Case Name:** TC_FILESYSTEM_AUTH

c. **Execution Steps:**

NOTE: The commands in the following description are considering Ubuntu 20.04 as the Operating System. The commands may vary for different Operating Systems.

1. The tester checks that OS-level permissions are configured correctly for users that are authorized to modify files, data, directories or file systems on the system.

We considered two files '*test1.txt*' & '*test2.txt*' and the user account 'tom'.

'tom' is authorized to access as well as modify (i.e. read + write) *test1.txt* but authorized only to read *test2.txt*.

File access permissions for *test1.txt*:

```
$ ls -lrt test1.txt
-rwxrw-r-- 1 tom tom 19 Jul 24 17:02 test1.txt
```

File access permissions for *test2.txt*:

```
$ ls -lrt test2.txt
-rwxrw-r-- 1 siddhesh siddhesh 19 May 31 2022 test2.txt
```

We considered a directory *test_dir1* for which the user 'tom' has read as well as write privileges, and a directory *test_dir2* for which the user 'tom' has no access privileges.

Access permissions set for directory *test_dir1*:

```
$ ls -lrt | grep test_dir1
drwxr-xr-x 2 tom tom 4096 Jul 24 00:38 test_dir1
```

Access permissions set for directory *test_dir2*:

```
$ ls -lrt | grep test_dir2
drwxrwx--- 2 siddhesh siddhesh 4096 Jul 24 17:06 test_dir2
```

Case for permission for deletion of files/directories:

Here, we again use the linux command 'chmod' to set permissions to a file/directory such that only the owner of the file/directory can delete it and not the other user.

We consider a directory *test_dir2* (owned by other user) for which the above permission is set as follows :

`sudo chmod +t test_dir2`

screenshot of file access permissions after running above command:

```
siddhesh@stark99:~$ ls -lrt | grep test_dir2
drwxrwx--T 2 siddhesh siddhesh 4096 Jul 24 17:06 test_dir2
```

2. The tester tries to modify the files and directories for which the user has the necessary privileges.

User (tom) trying to read *test1.txt* (operation successful):

```
siddhesh@stark99:~$ su tom
Password:
$ cd ../tom
$ cat test1.txt
this is file test1
```

User(tom) trying to modify *test1.txt* (operation successful):

```
siddhesh@stark99:~$ su tom
Password:
$ cd ../tom
$ cat test1.txt
this is file test1
$ nano test1.txt
$ cat test1.txt
this is file test1 (modified by tom)
```

Performing access(read), create & modify operations on *test_dir1* (operations successful):

```
siddhesh@stark99:~$ su tom
Password:
$ cd ../tom
$ cd test_dir1
$ ls
$ touch file.txt
$ ls
file.txt
```

Deleting the file as owner (operation successful):

```
siddhesh@stark99:~$ rmdir test_dir2
siddhesh@stark99:~$ ls | grep test_dir2
```

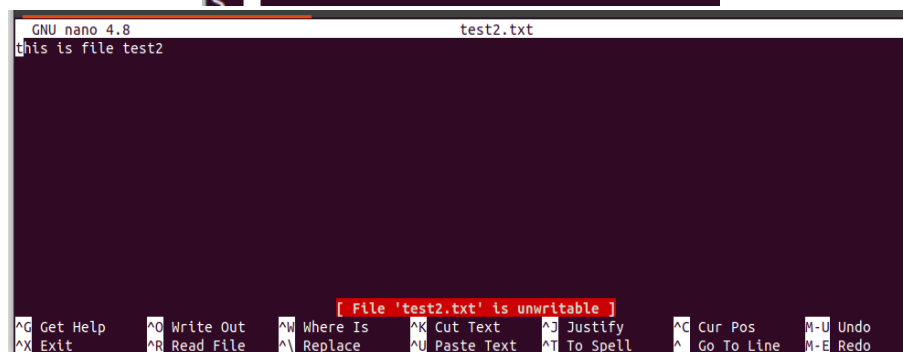
3. The tester tries to modify the files and directories for which the user doesn't have the necessary privileges

User(tom) trying to read *test2.txt* (Operation successful):

```
siddhesh@stark99:~$ su tom
Password:
$ cat test2.txt
this is file test2
```

User(tom) trying to modify *test2.txt* (Operation denied):

```
siddhesh@stark99:~$ su tom
Password:
$ cat test2.txt
this is file test2
$ nano test2.txt
$ cat test2.txt
this is file test2
$
```



Performing access operation on *test_dir2* (operation denied):


```
siddhesh@stark99:~$ su tom
Password:
$ cd test_dir2
sh: 1: cd: can't cd to test_dir2
```

Deleting file as another user (operation denied):

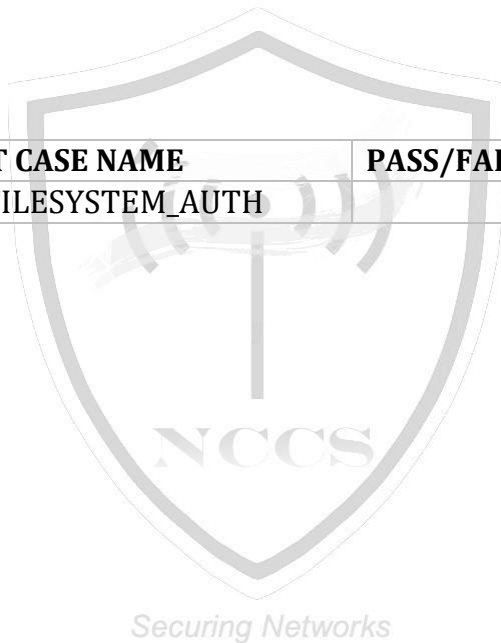
```
siddhesh@stark99:~$ su tom
Password:
$ rmdir test_dir2
rmdir: failed to remove 'test_dir2': Permission denied
```

d. Test Observations:

- **(Case 1)** Tester verifies that the permitted operations on the files and directories are allowed.
- **(Case 2)** Tester verifies that the non-permitted/denied operations on the files & directories are denied.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_FILESYSTEM_AUTH		



2.10.10. TSTP for Syn Flood Prevention

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.10
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 10: Operating System
2. **<Security Requirement No & Name >** 2.10.10 Syn Flood Prevention
3. **<Requirement Description: >**SMF shall support a mechanism to prevent Syn Flood attacks.
This feature shall be enabled by default.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section - 4.3.3.1.4]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions**

- The Network Product is listening on a TCP port one of its interfaces.
- A network traffic analyser on the network product (e.g., TCPDUMP) or an external traffic analyser directly connected to the network product is available.
- A host is connected to the Network Product interface, and it is equipped with a tool able to reproduce a Syn Flood attack (e.g., Nmap or hping)

7. **Test Objective:**

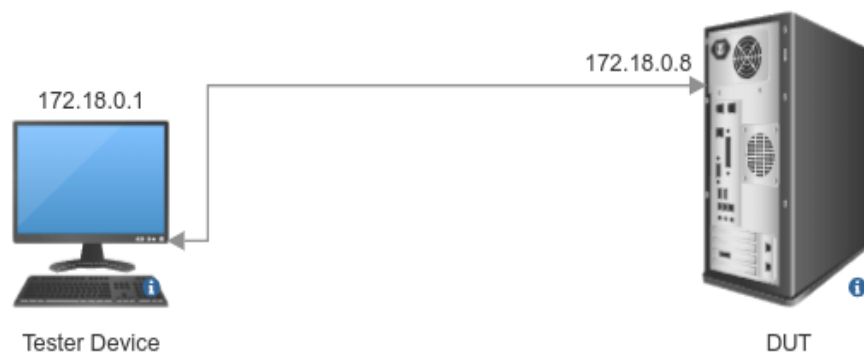
- To verify that the network product provides externally reachable services which are robust against unexpected input.
- The target of this test are the protocol stacks (e.g., diameter stack) rather than the applications (e.g., web app).

8. **Test Plan:**

8.1 **Number of Test Scenarios:**

- 8.1.1 Test Scenario for SYN flooding attack

8.2 **Test Setup Diagram:**



Securing Networks

8.3 **Tools Used:** Wireshark and Command Line Interface of DUT and any tool to simulate a huge number of SYN packets.

8.4 **Test Execution Steps**

- The tester configures the tool to send a huge number of TCP Syn packets against the Network Product.
- The Network Product is still up and running normally, its services are still available and reachable, the memory is not exhausted, there is no crash.

9. **Expected Results for Pass:** The Network Product does not become inoperative.

10. **Expected Format of Evidence:** A Pass/Fail result provided by the tester.

11. Test Execution:

➤ **Test Case Number:** 01

a. **Test Case Name:** TC_SYN_FLOOD_ATTACK

b. **Test Case Description:** Tests need to be conducted to check the effects of SYN Flood attack on the running setup

c. **Execution Steps:**

1. Note the system resources (RAM and CPU usage) of the DUT
2. The tester configures the tool to send a huge amount of TCP Syn packets against the Network Product (e.g. here we have simulated the attack using linux command hping3 -i <waiting time between each packet> -S -p <TCP port> -c <Number of packets> <SMF IP>)
3. Verify if there is unexpected **increased CPU usage** and **memory consumption**.

d. **Test Observations:**

- **Case 1:** Check whether Network Product experiences unexpected hoarding of resources in the presence of SYN Flood attack.

Use command **cat /proc/meminfo** to know the resources of the DUT

```
MemTotal:      8055556 kB
MemFree:       321696 kB
MemAvailable:  2791068 kB
Buffers:       448412 kB
Cached:        2881632 kB
SwapCached:    2216 kB
Active:        2090560 kB
Inactive:      4774616 kB
Active(anon):  81780 kB
Inactive(anon): 4237096 kB
Active(file):  2008780 kB
Inactive(file): 537520 kB
Unevictable:   253412 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB
```

In the testcase, we should get an external entity connected to the network product which produces huge amounts of SYN traffic in order to cause DOS at the intended Network product.

```
sudo hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 10.61.3.66
```

Figure 2: Above figure shows the command that is used to flood the network product with SYN packets.

Usage of the above command should have no unexpected effect on the operation of the network product

```

MemTotal:      8055556 kB
MemFree:       243252 kB
MemAvailable:  2508652 kB
Buffers:       409428 kB
Cached:        2647128 kB
SwapCached:    480 kB
Active:        1925888 kB
Inactive:      5156488 kB
Active(anon):   90536 kB
Inactive(anon): 4636152 kB
Active(file):   1835352 kB
Inactive(file): 520336 kB
Unevictable:   111576 kB
Mlocked:       32 kB
SwapTotal:     1158220 kB

```

```

1234 63.350433744 10.61.3.97 10.61.3.66 176 TCP 1458 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1377 64.347910921 10.61.3.97 10.61.3.66 176 TCP 1459 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1390 65.348088869 10.61.3.97 10.61.3.66 176 TCP 1460 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1411 66.348305233 10.61.3.97 10.61.3.66 176 TCP 1461 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1426 67.348522420 10.61.3.97 10.61.3.66 176 TCP 1462 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1441 68.348606805 10.61.3.97 10.61.3.66 176 TCP 1463 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1466 69.348761465 10.61.3.97 10.61.3.66 176 TCP 1464 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1478 70.348975229 10.61.3.97 10.61.3.66 176 TCP 1465 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1489 71.349547120 10.61.3.97 10.61.3.66 176 TCP 1466 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1504 72.350213079 10.61.3.97 10.61.3.66 176 TCP 1467 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1510 73.350498314 10.61.3.97 10.61.3.66 176 TCP 1468 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1515 74.351132908 10.61.3.97 10.61.3.66 176 TCP 1469 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1521 75.351244990 10.61.3.97 10.61.3.66 176 TCP 1470 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse
1529 76.351430325 10.61.3.97 10.61.3.66 176 TCP 1471 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reasse

```

Above TCP trace shows that numerous TCP SYN messages were sent from 10.61.3.97 (Tester) to DUT (10.61.3.66)

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_SYN_FLOOD_PREVENTION		

2.10.11 TSTP for Handling of IP options and extensions

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.11
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) **<ITSAR Section No & Name>** Section 10: Operating System
- 2) **<Security Requirement No & Name >** 2.10.11 Handling of IP options and extensions
- 3) **<Requirement Description: >** IP packets with unnecessary options or extension headers shall not be processed. IP options and extension headers (e.g., source routing) are only required in exceptional cases. So, all packets with enabled IP options or extension headers shall be filtered.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section - 4.2.4.1.1.3]

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of configfile)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
```

ubuntu	20.04	88bd68917189	sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu	latest	fb52e22af1b0	sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

6) **Preconditions**

- The manufacturer declares in the documentation accompanying the network product at least the following information:
 - The support of filtering capability for IP packets with unnecessary options or extensions headers.
 - The actions performed by the network product when an IP packet with unnecessary options or extensions headers is received (e.g., the packet is dropped, the options or extensions are ignored and the packet is treated as if it has no IP options, etc.).
 - Guidelines on how to enable and configure this filtering capability.
- The network product has at least one physical interface named if1 supporting both IPv4 and IPv6. If the network product does not support IPv6 then IPv6 related steps and checks may be skipped.
- A network traffic analyser on the network product (e.g., TCPDUMP, Wireshark) or an external traffic analyser directly connected to the network product is available.
- The tester has administrative privileges.
- A tester machine is available with a tool able to send IPv4 packets with the IP Options and IPv6 packets (if supported by the network product) with Extension Header set (e.g., Scapy).
- Vendor Must provide documentation detailing extension set and IP options to be enabled and it should work only for those headers and extensions.

7) **Test Objective:** To verify that the network product provides functionality to filter out IP packets with unnecessary options or extension headers.

8) **Test Plan:**

8.1 **Test Setup Diagram:**



8.2 **Tools Used:** Wireshark or TCPDUMP, ScaPY (or any other packet manipulation/forgery tool)

8.3 **Test Execution Steps**

- The tester logs in the network product.

- The tester configures on the network product a filtering rule to drop all IP packets containing an IP Option set
 - o The tester establishes an O&M session on if1 interface
 - o Using the tool (e.g. Scapy) the tester sends from the tester machine an IPv4 TCP SYN packet with an appropriate destination port to if1 interface without setting any IP Options
 - o Using the network traffic analyser, the tester verifies that the IP packet is received by the network product and the tester verifies that the corresponding ACK message is sent back.
 - o Using the tool (e.g. Scapy) the tester sends an IPv4 TCP SYN packet with an appropriate destination port and an IP Option set to the if1 interface
 - o Using the network traffic analyser, the tester verifies that the IP packet is received by the network product but no ACK message is sent back. This confirms the packet is dropped as expected from the filtering rule.
 - The tester configures on the network product a filtering rule to drop all incoming packets based on specific Extension Header Types, e.g. packets with the Routing Header extension. This step may be skipped if the network product does not support IPv6.
 - o Using the tool (e.g. Scapy) the tester sends from the tester machine an IPv6 TCP SYN packet with an appropriate destination port to if1 interface without setting any extension header
 - o Using the network traffic analyser, the tester verifies that the IP packet is received by the network product and the tester verifies that the corresponding ACK message is sent back.
 - o Using the tool (e.g. Scapy) the tester sends an IPv6 TCP SYN packet with an appropriate destination port and an extension header set to the if1 interface
 - o Using the network traffic analyser, the tester verifies that the IP packet is received by the network product but no ACK message is sent back. This confirms the packet is dropped as expected from the filtering rule.
- 9) **Expected Results for Pass:** The network product discards IPv4 packets with unnecessary options or IPv6 packets (assuming the network product supports IPv6) with extension header.

10) **Expected Format of Evidence:**

- A testing report provided by the testing agency which will consist of the following information:
- Used tools and their configurations
- Settings and configurations used
- Pcap trace
- Screenshot

- Test result (Passed or not)

11) Test Execution:

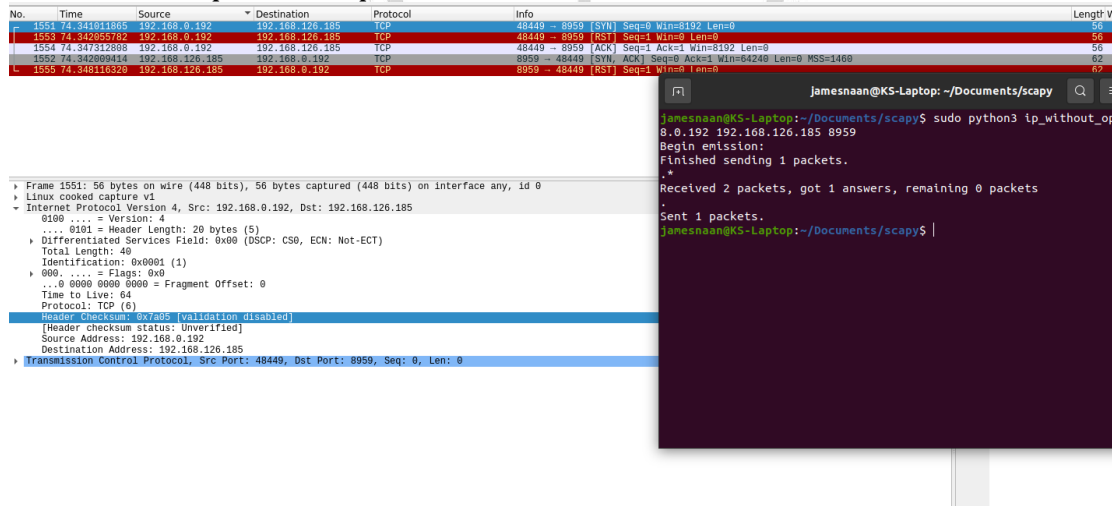
➤ **Test Case Number:** 01

a. **Test Case Name:** TC_HANDLING-IP-OPTIONS

b. **Test Case Description:** To verify that the network product discards IPv4 packets with unnecessary options.

c. **Execution Steps:**

1. The tester shall log into the DUT and check if the filtering rule has been enabled to drop all IP packets containing an IP Option set. **The tester shall check vendor documentation to understand how it is being done in the network product and where to check.**
2. The tester shall forge a TCP SYN packet using a tool such as ScaPY, **without** IP Options **set**. A sample python script (**ip_without_options.py**) has been attached with this document.
3. The tester opens Wireshark and uses appropriate display filter to observe packets going to and arriving from DUT. Here the filter is: **ip.addr == <DUT ip>**
4. The tester runs the script from the tester device to send the SYN packet. Here the sample script is run as follows: **sudo python3 ip_without_options.py <Tester ip> <DUT ip> <DUT port>**
5. The tester captures the packets.

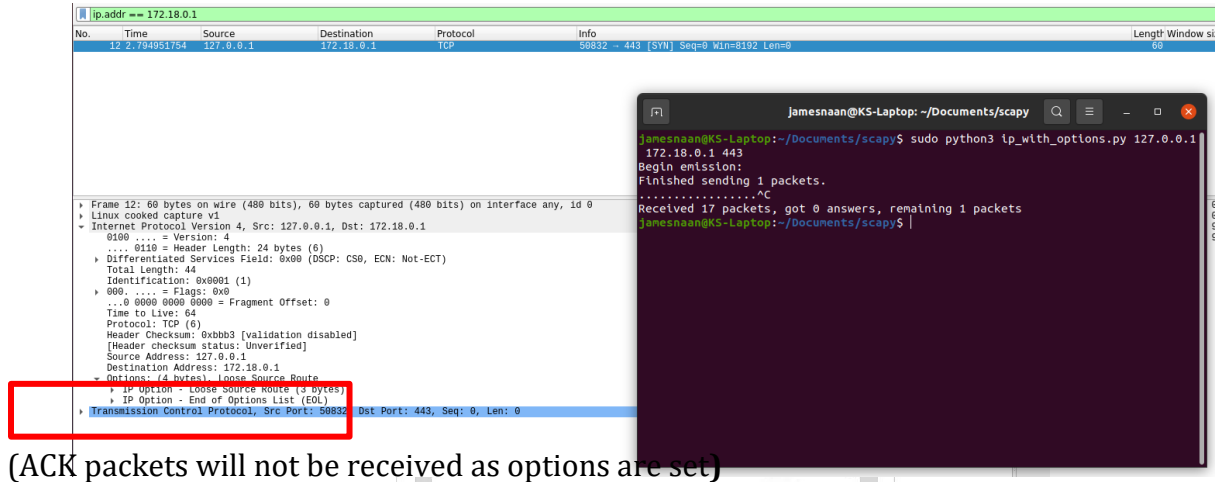


(ACK packets will be received as options are not set)

Note: Ignore the RST packets. This is generated because since we are using a forging tool to create SYN packets, the SYNACK is recognized as forged as the Linux kernel did not register the initial SYN packet, so it drops the SYNACK and resends an RST Packet, which is shown here, even in this case it is the tester device doing so. DUT is working properly.

6. The tester shall forge a TCP SYN packet using a tool such as ScaPY, with IP Options **set**. A sample python script (**ip_with_options.py**) has been attached with this document.

7. The tester runs the script from the tester device to send the SYN packet. Here the sample script is run as follows: **sudo python3 ip_with_options.py <Tester ip> <DUT ip> <DUT port>**
8. The tester captures the packets



➤ **Test Case Number:** 02 (To be ignored if IPv6 not supported by DUT)

a. **Test Case Name:** TC_HANDLING-IP-EXTENSIONS

b. **Test Case Description:** To verify that the network product provides functionality to filter out IP packets with unnecessary options or extension headers.

c. **Execution Steps:**

9. The tester shall log into the DUT and check if the filtering rule has been enabled to drop all IP packets containing IPv6 extensions set. **The tester shall check vendor documentation to understand how it is being done in the network product and where to check.**
10. The tester shall forge a TCP SYN packet using a tool such as ScaPY, **without** IPv6 Extensions set.

Securing Networks

50.450710791	::1	::1	TCP	74444 → 80 [SYN] Seq=0 Win=8192 Le
60.651381902	127.0.0.1	127.0.0.53	DNS	100 Standard query 0x923f A word-edi
Internet Protocol Version 6, Src: ::1, Dst: ::1				
0110 = Version: 6				
.... 0000 0000 = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)				
.... 0000 0000 0000 0000 = Flow Label: 0x000000				
Payload Length: 20				
Next Header: TCP (6)				
Hop Limit: 64				
Source: ::1				
Destination: ::1				
Transmission Control Protocol, Src Port: 444, Dst Port: 80, Seq: 0, Len: 0				
0000	ff ff ff ff ff ff 00 00	00 00 00 00 86 dd 60 00	
0010	00 00 00 14 06 40 00 00	00 00 00 00 00 00 00 00	...@.....	
0020	00 00 00 00 00 01 00 00	00 00 00 00 00 00 00 00	
0030	00 00 00 00 00 01 01 bc	00 50 00 00 03 e8 00 00P.....	
0040	00 00 50 02 20 00 89 ed	00 00	..P.....	

(Here we can see that the IPv6 Hop-by-Hop Options are set)

11. The tester opens Wireshark and uses appropriate display filter to observe packets going to and arriving from DUT. Here the filter is: **ipv6.addr == <DUT ip>**
12. The tester runs the script from the tester device to send the SYN packet. The tester captures the packets.
13. The tester shall forge a TCP SYN packet using a tool such as ScaPY, with IPv6 Extensions set.

```

No.    Time           Source            Destination       Protocol  Length  Info
-----
1 0.000000000    ::1              ::1              TCP        82 444 → 80 [SYN] Seq=0 Win=8192 Len=0

Frame 1: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 6, Src: ::1, Dst: ::1
  0110 .... = Version: 6
  .... 0000 0000 .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
  .... 0000 0000 0000 0000 0000 0000 = Flow Label: 0x000000
Payload Length: 28
Next Header: IPv6 Hop-by-Hop Option (0)
Hop Limit: 64
Source: ::1
Destination: ::1
IPv6 Hop-by-Hop Option
0000  ff ff ff ff ff ff 00 00 00 00 00 00 86 dd 60 00  ....@.....
0010  00 00 00 1c 00 40 00 00 00 00 00 00 00 00 00 00  .....@.....
0020  00 00 00 00 00 01 00 00 00 00 00 00 00 00 00 00  .....@.....
0030  00 00 00 00 00 01 06 00 01 04 01 04 01 00 01 bc  .....@.....
0040  00 50 00 00 03 e8 00 00 00 00 50 02 20 00 89 ed  .....@.....
0050  00 00
  
```

14. The tester runs the script from the tester device to send the SYN packet. The tester captures the packets.

d. Test Observations:

- It should be ensured that the network product discards IPv4 packets with unnecessary options or IPv6 packets (assuming the network product supports IPv6) with unnecessary extensions set.
- Vendor Must provide documentation detailing extension set and IP options to be enabled and it should work only for those headers and extensions.

12)Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_HANDLING-IP-OPTIONS		
2	TC_HANDLING-IP-EXTENSIONS		

2.10.12 TSTP for Evaluation of Restriction on Running Scripts/Batch Processes

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.12
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 10 - Operating System
2. **<Security Requirement No & Name >** 2.10.12 Restriction on running Scripts/Batch processes.
3. **<Requirement Description: >**
 - a. Scheduled tasks for carrying out the activities such as taking the backups, monitoring disk space and system maintenance activities shall be executed by the privileged user such as administrator only.
 - b. Similarly, SMF shall have feature to restrict Scripts / Batch-processes / Macros usage among various users.
 - c. It shall be possible to administratively configure scheduled tasks usage i.e Cron-Job usage (permit / deny) among various users like Normal users, privileged users.

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:**

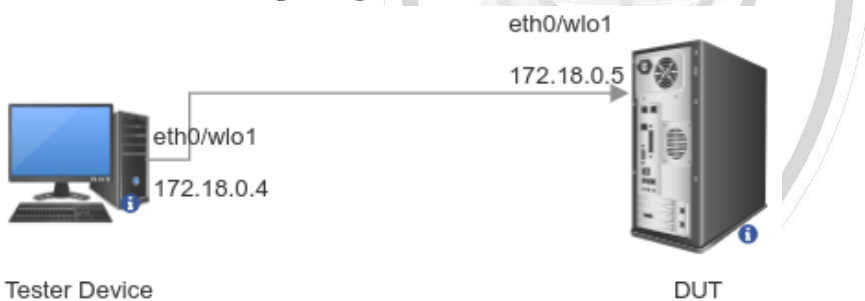
- The tester must be provided with access credentials for the Normal & privileged user accounts that exist on the DUT.
- The tester must be provided access to the administrative account for configuring the scheduled job/Script/Batch Processes/macros usage among different users (Normal/privileged users) within the DUT.
- The Vendor must provide the documentation specifying
 - List of scheduled tasks available in the Network Product.
 - Steps to enable/disable scheduled tasks within the Network Product.
 - the procedure to configure usage of scripts/processes among the users.

7. **Test Objective:** To verify that:

- The scheduled tasks shall be executed by privileged users only.
- It shall be possible to administratively configure scripts/batch processes/scheduled tasks usage among the various users.

8. **Test Plan:**

8.1 **Test Setup Diagram:**



8.2 **Tools used:** DUT CLI

8.3 **Test Execution Steps:**

1. The tester shall login as a Privileged User and attempt to enable the scheduled tasks listed in the network product documentation.
2. The tester shall login as a Normal User and attempt to enable the scheduled tasks listed in the network product documentation.
3. The tester logs into the administrative account and modifies the usage privileges for a particular script/batch process/scheduled task.
4. The tester verifies if the modification made to the usage privileges for a particular script/batch process/scheduled task are in place.

9. **Expected Results for Pass:** The scheduled task(s) shall execute only when attempted from a privileged user. The tester shall be able to successfully configure usage privileges for scheduled tasks/scripts/batch processes.

10. **Expected format of Evidence:** Screenshot/logs containing:

- Attempt made to run a scheduled task/script/batch process along with user details and outcome(allowed/denied).
- Changes made to the usage privileges for the scheduled task/script/batch process.

11. **Test Execution:**

➤ **Test case Number:** 1

a. **Test Case Name:** TC_RESTRICT_SCRIPT_EXECUTION

b. **Test Case Description:**

To verify that:

- The scheduled tasks shall be executed by privileged users only.
- It shall be possible to administratively configure scripts/batch processes/scheduled tasks usage among the various users.

c. **Execution Steps:**

1. The tester shall login as a Privileged User and attempt to enable the scheduled tasks listed in the network product documentation.

In Ubuntu OS, we can create, enable and disable scheduled tasks by using the *cron* tool.

This user can run scheduled tasks by setting them up in the */etc/crontab* file.

Only the Privileged User should have access to modify */etc/crontab* file to set up Cron Jobs/Scheduled tasks.

Permissions for the */etc/crontab* file:

```
root@stark99:/home/siddhesh# ls -lrt /etc/crontab
-rw-r--r-- 1 root root 1042 Feb 14 2020 /etc/crontab
root@stark99:/home/siddhesh#
```

Here, Only the *root* user is allowed to write to the file.

Root User allowed to setup cron jobs in */etc/crontab*:



```
root@stark99:/home/siddhesh
GNU nano 4.8 /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab`
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

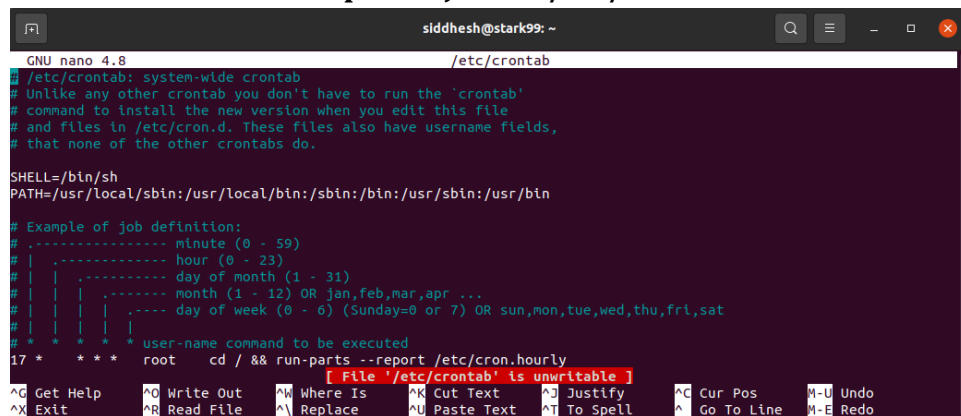
SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# ..... minute (0 - 59)
# | ..... hour (0 - 23)
# | | ..... day of month (1 - 31)
# | | | ..... month (1 - 12) OR jan,feb,mar,apr ...
# | | | | ..... day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# * * * * * user-name command to be executed
17 * * * * root cd / && run-parts --report /etc/cron.hourly

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos ^U Undo
^X Exit ^R Read File ^E Replace ^V Paste Text ^T To Spell ^G Go To Line ^_ Redo
```

- The tester shall login as a Normal User and attempt to enable the scheduled tasks listed in the network product documentation.

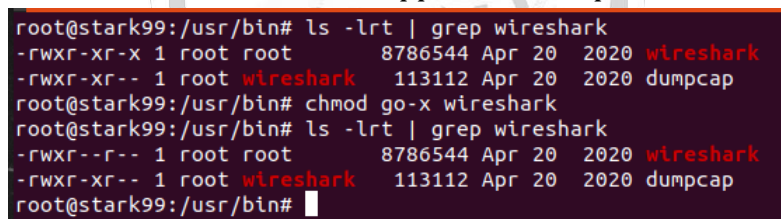
Normal User denied to setup cron jobs in /etc/crontab:



```
siddhesh@stark99: ~  
GNU nano 4.8 /etc/crontab  
# /etc/crontab: system-wide crontab  
# Unlike any other crontab you don't have to run the `crontab`  
# command to install the new version when you edit this file  
# and files in /etc/cron.d. These files also have username fields,  
# that none of the other crontabs do.  
  
SHELL=/bin/sh  
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin  
  
# Example of job definition:  
#----- minute (0 - 59)  
#----- hour (0 - 23)  
#----- day of month (1 - 31)  
#----- month (1 - 12) OR jan,feb,mar,apr ...  
#----- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat  
# user-name command to be executed  
17 * * * * root cd / && run-parts --report /etc/cron.hourly  
[File '/etc/crontab' is unwritable]  
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos  
^X Exit ^R Read File ^N Replace ^U Paste Text ^T To Spell ^G Go To Line ^M Undo  
^E Redo
```

- The tester logs into the administrative account and modifies the usage privileges for a particular script/batch process/scheduled task.

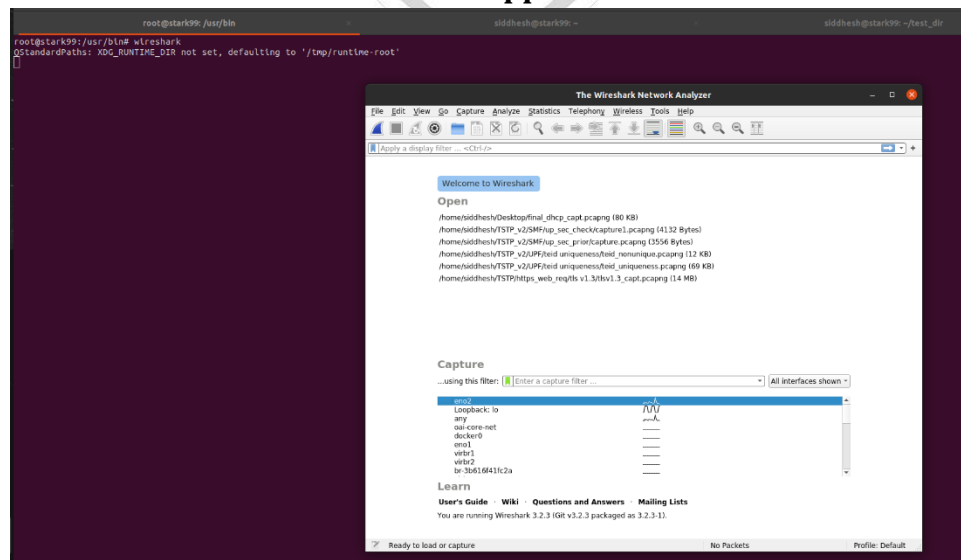
Here we modified the execution/usage permissions for the *wireshark* application, such that only the root user can execute the application script.



```
root@stark99:/usr/bin# ls -lrt | grep wireshark  
-rwxr-xr-x 1 root root 8786544 Apr 20 2020 wireshark  
-rwxr-xr-x 1 root wireshark 113112 Apr 20 2020 dumpcap  
root@stark99:/usr/bin# chmod go-x wireshark  
root@stark99:/usr/bin# ls -lrt | grep wireshark  
-rwxr--r-- 1 root root 8786544 Apr 20 2020 wireshark  
-rwxr-xr-x 1 root wireshark 113112 Apr 20 2020 dumpcap  
root@stark99:/usr/bin#
```

- The tester verifies if the modification made to the usage privileges for a particular script/batch process/scheduled task are in place.

Root is allowed to run the wireshark application:



Other users are not allowed to run the wireshark application:


```
siddhesh@stark99:~$ wireshark
bash: /usr/bin/wireshark: Permission denied
siddhesh@stark99:~$
```

d. Test Observations: It should be ensured that

- The scheduled tasks are allowed to be executed only by privileged users.
- It is possible to administratively configure usage privileges for scheduled tasks/scripts/batch processes.

12. **Test Case result:**

Sl. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_RESTRICT_SCRIPT_EXECUTION		



2.10.13 TSTP for Evaluation of Restrictions on Soft-Restart

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.10.13
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 10: Operating System
2. **<Security Requirement No & Name>** 2.10.13 Restrictions on Soft-Restart
3. **<Requirement Description>** SMF shall restrict software-based system restart options usage among various users. The software reset / restart either through command or use of key-combinations like CTRL+ALT+DEL is not available to normal users for prevention of unintended / malicious trigger of system reset / restart.

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration: Check if the reboot option is enabled

```
iith@vm2:~$ reboot
-bash: /usr/sbin/reboot: Permission denied
```

- We can also check that the shutdown and reboot option is disabled for this user in the configuration file.

Command – `sudo visudo`

```
#Defaults:%sudo env_keep += "GPG_AGENT_INFO"

# Host alias specification

# User alias specification

# Cmnd alias specification
Cmnd_Alias SHUTDOWN = /sbin/shutdown,/sbin/reboot,/sbin/halt,/sbin/poweroff

# User privilege specification
iith ALL=(ALL:ALL) ALL, !SHUTDOWN
```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. Preconditions

- The tester should have root access.
- The tester should create one new account on DUT.
- The tester should login with the new account.
- The document should be provided by OEM specifying any external command for soft restart.

7. **Test Objective:-** To verify that the software reset / restart either through command or use of key-combinations like CTRL+ALT+DEL is not available to normal users.

8. Test Plan:

8.1 Test Setup Diagram:

8.2 Tools used: Command line

8.3 Test Execution Steps

- All the preconditions should meet.
- The tester should login as the new account created.
- Open terminal.
- To check the key combination the tester can use
ls -l /etc/systemd/system/ctrl-alt-del.target
- The tester should try to press the key combination CTRL+ALT+DEL.
- The system should not be rebooted.
- The user tries to reboot with command - "**reboot**"
- It shouldn't work.
- The tester also needs to check for options provided by OEM for soft restart.

Note: - the similar approach will be followed for other software like Apache.

9. **Expected Results for Pass:** When a normal user tries to reboot the system, it should not be allowed.

10. **Expected Format of Evidence:** Screenshots denying the permission for reboot for normal user.

11. **Test Execution:**

➤ **Test Case Number: 1**

a. **Test Case Name:** TC1_RESTRICTIONS_ON_SOFT_RESTART

b. **Test Case Description:** To ensure that the system should not be rebooted after pressing the key combination CTRL+ALT+DEL or any other command.

c. **Execution Steps:**

• **Positive case**

iith is a normal user created by tester, type “**reboot**” command and check if system reboots

```
iith@vm2:~$ reboot
-bash: /usr/sbin/reboot: Permission denied
```

To check if CTRL+ALT+DEL is disabled, check with command:

“**ls -l /etc/systemd/system/ctrl-alt-del.target**”

If it is set to null it means that this functionality is disabled.

```
vm2@vm2:~$ ls -l /etc/systemd/system/ctrl-alt-del.target
lrwxrwxrwx 1 root root 9 Jun 13 12:17 /etc/systemd/system/ctrl-alt-del.target -> /dev/null
```

d. **Test Observations:**

- DUT meets test case 1 when the normal user tries to restart with a command or key option it should not be allowed to do so.
- DUT does not meet test case 1 when the reboot option is enabled for normal user.

12. **Test Case Result:**

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_RESTRICTIONS_ON_SOFT_RESTART	PASS	

2.11.1 TSTP for Evaluation of HTTPS

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.1
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF.

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 11: Web Server
2. **<Security Requirement No & Name >** 2.11.1 HTTP
3. **<Requirement Description: >** The communication between Web client and Web server shall be protected strictly using the Secure cryptographic controls prescribed in Table 1 of the latest document "Cryptographic Controls For Indian Telecom Security Assurance Requirements (ITSAR)" only.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.2.5.1]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: **nmap -p- 172.18.0.8 -sC -sV** (IP address of DUT)

```

[ashwini@ashwini-news-lab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_   Not valid before: 2023-05-17T08:39:01
|_   Not valid after: 2024-05-16T08:39:01
|_   tls-alpn:
|_     http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

a. For Apache httpd:

- Command used: **httpd -v**
(To get version information)

```

amf@localhost $ httpd -v
Server version: Apache/2.4.57 (Unix)
Server built:   May 11 2023 19:37:07

```

- Command used: **apachectl -M | grep ssl**
(To check if SSL module is loaded or not)

```

amf@localhost $ apachectl -M | grep ssl
ssl_module (shared)

```

- Command used: **apachectl -t -D DUMP_INCLUDES**
(To check if SSL configuration is loaded or not)

```

amf@localhost $ apachectl -t -D DUMP_INCLUDES
Included configuration files:
(*) /usr/local/apache2/conf/httpd.conf
(541) /usr/local/apache2/conf/extra/httpd-ssl.conf

```

- Command used: **openssl ciphers -s -tls1_3**
(To get supported cipher suite for TLS 1.3)

The tester should verify that the Symmetric Key encryption and decryption algorithm available in the DUT must match the Secure cryptographic controls prescribed in Table 1

of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)”

```
amf@localhost $ openssl ciphers -s -tls1_3
TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256
```

- Command used: **openssl ciphers -s -tls1_2**

(To get supported cipher suite for TLS 1.2)

The tester should verify that the Symmetric Key encryption and decryption algorithm available in the DUT must match the Secure cryptographic controls prescribed in Table 1 of the latest document “Cryptographic Controls For Indian Telecom Security Assurance Requirements (ITSAR)”

```
amf@localhost $ openssl ciphers -s -tls1_2
ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES128-SHA:DHE-RSA-AES128-SHA:AES256-GCM-SHA384:AES128-GCM-SHA256:AES256-SHA256:AES128-SHA256:AES256-SHA:AES128-SHA
```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

Securing Networks

6. Preconditions

- Network product documentation containing information about supported Web Servers and Protocol to communicate with the Web Server in the DUT is provided by the vendor.
- A peer implementing the security protocol same as the one configured by the vendor on DUT for Web Server, (e.g., HTTPS client) shall be available.
- Network product documentation stating which security protocols for protection of data in transit are implemented and which profiles in TS 33.310 [9] and TS 33.210 [X] are applicable is provided by the vendor.
- For TLS, the tester shall base the tests on the profile defined by 3GPP in TS 33.310 [9].
- Test environment with a Web Browser.

- All ciphers supported by the server need to be checked that they are as per Table 1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)”

For example, in Apache httpd running on ubuntu 20.04 the command *OpenSSL ciphers -v* is used to check all supported ciphers. The tester must ensure there are no weak ciphers and if so, those ciphers must be removed from configuration before conducting the rest of the test. If this fails

- TEST FAILS

```
naan30@naa30-pc:~$ openssl ciphers -v
TLS_AES_256_GCM_SHA384 TLSv1.3 Kx=any Au=any Enc=AESGCM(256) Mac=AEAD
TLS_CHACHA20_POLY1305_SHA256 TLSv1.3 Kx=any Au=any Enc=CHACHA20/POLY1305(256) Mac=AEAD
TLS_AES_128_GCM_SHA256 TLSv1.3 Kx=any Au=any Enc=AESGCM(128) Mac=AEAD
ECDHE-ECDSA-AES256-GCM-SHA384 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AESGCM(256) Mac=AEAD
ECDHE-RSA-AES256-GCM-SHA384 TLSv1.2 Kx=ECDH Au=RSA Enc=AESGCM(256) Mac=AEAD
DHE-RSA-AES256-GCM-SHA384 TLSv1.2 Kx=DH Au=RSA Enc=AESGCM(256) Mac=AEAD
ECDHE-ECDSA-CHACHA20-POLY1305 TLSv1.2 Kx=ECDH Au=ECDSA Enc=CHACHA20/POLY1305(256) Mac=AEAD
ECDHE-RSA-CHACHA20-POLY1305 TLSv1.2 Kx=ECDH Au=RSA Enc=CHACHA20/POLY1305(256) Mac=AEAD
DHE-RSA-CHACHA20-POLY1305 TLSv1.2 Kx=DH Au=RSA Enc=CHACHA20/POLY1305(256) Mac=AEAD
ECDHE-ECDSA-AES128-GCM-SHA256 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AESGCM(128) Mac=AEAD
ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2 Kx=ECDH Au=RSA Enc=AESGCM(128) Mac=AEAD
DHE-RSA-AES128-GCM-SHA256 TLSv1.2 Kx=DH Au=RSA Enc=AESGCM(128) Mac=AEAD
ECDHE-ECDSA-AES256-SHA384 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AES(256) Mac=SHA384
ECDHE-RSA-AES256-SHA384 TLSv1.2 Kx=ECDH Au=RSA Enc=AES(256) Mac=SHA384
DHE-RSA-AES256-SHA256 TLSv1.2 Kx=DH Au=RSA Enc=AES(256) Mac=SHA256
ECDHE-ECDSA-AES128-SHA256 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AES(128) Mac=SHA256
DHE-RSA-AES128-SHA256 TLSv1.2 Kx=DH Au=RSA Enc=AES(128) Mac=SHA256
ECDHE-ECDSA-AES256-SHA TLSv1 Kx=ECDH Au=ECDSA Enc=AES(256) Mac=SHA1
ECDHE-RSA-AES256-SHA TLSv1 Kx=ECDH Au=RSA Enc=AES(256) Mac=SHA1
DHE-RSA-AES256-SHA SSLv3 Kx=DH Au=RSA Enc=AES(256) Mac=SHA1
ECDHE-ECDSA-AES128-SHA TLSv1 Kx=ECDH Au=ECDSA Enc=AES(128) Mac=SHA1
ECDHE-RSA-AES128-SHA TLSv1 Kx=ECDH Au=RSA Enc=AES(128) Mac=SHA1
DHE-RSA-AES128-SHA SSLv3 Kx=DH Au=RSA Enc=AES(128) Mac=SHA1
RSA-PSK-AES256-GCM-SHA384 TLSv1.2 Kx=RSAPSK Au=RSA Enc=AESGCM(256) Mac=AEAD
DHE-PSK-AES256-GCM-SHA384 TLSv1.2 Kx=DHEPSK Au=PSK Enc=AESGCM(256) Mac=AEAD
RSA-PSK-CHACHA20-POLY1305 TLSv1.2 Kx=RSAPSK Au=RSA Enc=CHACHA20/POLY1305(256) Mac=AEAD
DHE-PSK-CHACHA20-POLY1305 TLSv1.2 Kx=DHEPSK Au=PSK Enc=CHACHA20/POLY1305(256) Mac=AEAD
ECDHE-PSK-CHACHA20-POLY1305 TLSv1.2 Kx=ECDHEPSK Au=PSK Enc=CHACHA20/POLY1305(256) Mac=AEAD
AES256-GCM-SHA384 TLSv1.2 Kx=RSA Au=RSA Enc=AESGCM(256) Mac=AEAD
PSK-AES256-GCM-SHA384 TLSv1.2 Kx=PSK Au=PSK Enc=AESGCM(256) Mac=AEAD
PSK-CHACHA20-POLY1305 TLSv1.2 Kx=PSK Au=PSK Enc=CHACHA20/POLY1305(256) Mac=AEAD
RSA-PSK-AES128-GCM-SHA256 TLSv1.2 Kx=RSAPSK Au=RSA Enc=AESGCM(128) Mac=AEAD
DHE-PSK-AES128-GCM-SHA256 TLSv1.2 Kx=DHEPSK Au=PSK Enc=AESGCM(128) Mac=AEAD
AES128-GCM-SHA256 TLSv1.2 Kx=RSA Au=RSA Enc=AESGCM(128) Mac=AEAD
PSK-AES128-GCM-SHA256 TLSv1.2 Kx=PSK Au=PSK Enc=AESGCM(128) Mac=AEAD
AES256-SHA256 TLSv1.2 Kx=RSA Au=RSA Enc=AES(256) Mac=SHA256
AES128-SHA256 TLSv1.2 Kx=RSA Au=RSA Enc=AES(128) Mac=SHA256
ECDHE-PSK-AES256-CBC-SHA384 TLSv1 Kx=ECDHEPSK Au=PSK Enc=AES(256) Mac=SHA384
ECDHE-PSK-AES256-CBC-SHA TLSv1 Kx=ECDHEPSK Au=PSK Enc=AES(256) Mac=SHA1
SRP-RSA-AES-256-CBC-SHA SSLv3 Kx=SRP Au=RSA Enc=AES(256) Mac=SHA1
SRP-AES-256-CBC-SHA SSLv3 Kx=SRP Au=SRP Enc=AES(256) Mac=SHA1
RSA-PSK-AES256-CBC-SHA384 TLSv1 Kx=RSAPSK Au=RSA Enc=AES(256) Mac=SHA384
DHE-PSK-AES256-CBC-SHA384 TLSv1 Kx=DHEPSK Au=PSK Enc=AES(256) Mac=SHA384
RSA-PSK-AES256-CBC-SHA SSLv3 Kx=RSAPSK Au=RSA Enc=AES(256) Mac=SHA1
DHE-PSK-AES256-CBC-SHA SSLv3 Kx=DHEPSK Au=PSK Enc=AES(256) Mac=SHA1
AES256-SHA SSLv3 Kx=RSA Au=RSA Enc=AES(256) Mac=SHA1
PSK-AES256-CBC-SHA384 TLSv1 Kx=PSK Au=PSK Enc=AES(256) Mac=SHA384
PSK-AES256-CBC-SHA SSLv3 Kx=PSK Au=PSK Enc=AES(256) Mac=SHA1
ECDHE-PSK-AES128-CBC-SHA256 TLSv1 Kx=ECDHEPSK Au=PSK Enc=AES(128) Mac=SHA256
ECDHE-PSK-AES128-CBC-SHA TLSv1 Kx=ECDHEPSK Au=PSK Enc=AES(128) Mac=SHA1
SRP-RSA-AES-128-CBC-SHA SSLv3 Kx=SRP Au=RSA Enc=AES(128) Mac=SHA1
SRP-AES-128-CBC-SHA SSLv3 Kx=SRP Au=SRP Enc=AES(128) Mac=SHA1
RSA-PSK-AES128-CBC-SHA256 TLSv1 Kx=RSAPSK Au=RSA Enc=AES(128) Mac=SHA256
DHE-PSK-AES128-CBC-SHA256 TLSv1 Kx=DHEPSK Au=PSK Enc=AES(128) Mac=SHA256
RSA-PSK-AES128-CBC-SHA SSLv3 Kx=RSAPSK Au=RSA Enc=AES(128) Mac=SHA1
DHE-PSK-AES128-CBC-SHA SSLv3 Kx=DHEPSK Au=PSK Enc=AES(128) Mac=SHA1
AES128-SHA SSLv3 Kx=RSA Au=RSA Enc=AES(128) Mac=SHA1
PSK-AES128-CBC-SHA256 TLSv1 Kx=PSK Au=PSK Enc=AES(128) Mac=SHA256
PSK-AES128-CBC-SHA SSLv3 Kx=PSK Au=PSK Enc=AES(128) Mac=SHA1
```


7. **Test Objective:** The communication between Web client and Web server shall be protected using TLS.

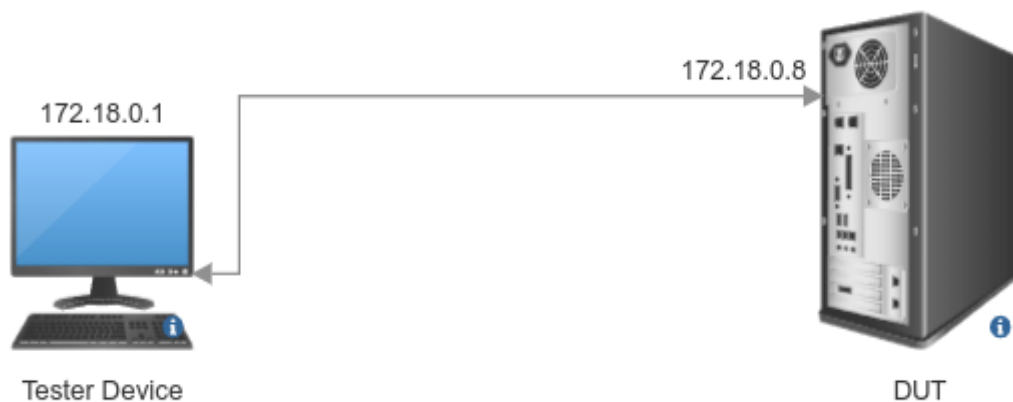
8. **Test Plan:**

8.1 **Number of Test Scenarios:**

8.1.1 **Test Scenario for HTTP**

- Check traffic is properly protected
- Check TLS is being used or not

8.2 **Test Setup Diagram:**



8.3 **Tools Used:** Wireshark in Tester Device

8.4 **Test Execution Steps**

- Launch the Wireshark app on the tester device.
- Power up the testbed
- The tester shall check that compliance with the selected security profile can be inferred from detailed provisions in the product documentation.
- The tester shall establish a secure connection between the network product and the peer and verify that all protocol versions and combinations of cryptographic algorithms that are mandated by the security profile are supported by the network product.

9. **Expected Results for Pass:**

- The traffic is properly protected, and insecure options are not accepted by the Network Product.
- The communication between Web client and Web server shall be protected using TLS.

10. **Expected Format of Evidence:**

- Screenshots of Wireshark and Pcap files capturing the Successful TLS Handshake.

- Screenshots of Web Browser accessing the Web Server

11. Test Execution:

➤ Test Case Number: 01

a. Test Case Name: TC_HTTPS

- b. **Test Case Description:** The communication between Web client and Web server shall be protected using TLS.

c. Execution Steps:

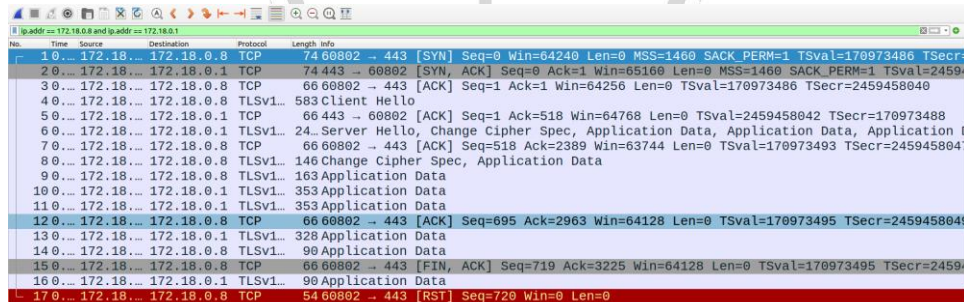
1. The tester shall open any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.

- URL: `https://<IP address of DUT>`



It works!

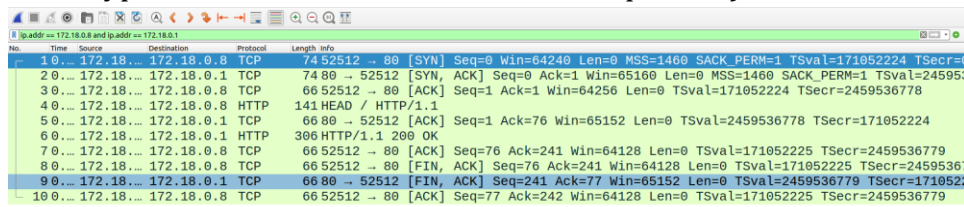
2. While the testbed is running, the tester should put appropriate display filter (IP address of DUT and Tester Device) in Wireshark



3. The tester should try to access the DUT using HTTP in URL

- URL: `http://<IP address of DUT>`

And then using the display filter from step 2 analyse the packet capture in Wireshark. (NO Unencrypted or Plain HTTP Traffic should be present).



4. The tester should verify the cipher suite used for TLS 1.2/1.3 are being used in the DUT in the message "**Server Hello**" is same as the Secure cryptographic controls prescribed

in Table 1 of the latest document “Cryptographic Controls For Indian Telecom Security Assurance Requirements (ITSAR).

```
50... 172.18.0... 172.18... TCP 66443 → 60802 [ACK] Seq=1 Ack=518 Win=64768 Len=0 TSval=24
60... 172.18.0... 172.18... TLS... 2..Server Hello, Change Cipher Spec, Application Data, Appl
70... 172.18.0... 172.18... TCP 6660802 → 443 [ACK] Seq=518 Ack=2389 Win=63744 Len=0 TSval
```

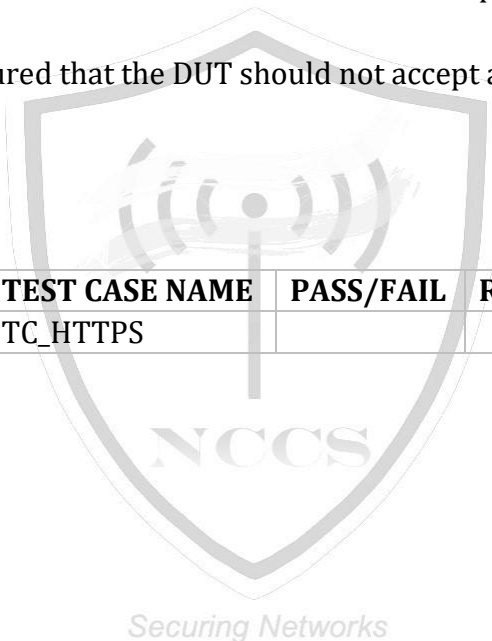
```
-Transport Layer Security
-TLSv1.3 Record Layer: Handshake Protocol: Server Hello
Content Type: Handshake (22)
Version: TLS 1.2 (0x0303)
Length: 122
-Handshake Protocol: Server Hello
Handshake Type: Server Hello (2)
Length: 118
Version: TLS 1.2 (0x0303)
Random: dafdc38dfa4c6b3112d508fd3f6fff1dd360eebbc07f24627fb9cdf750c38bc5
Session ID Length: 32
Session ID: 2f9d682c208eec0705d1248615eba23a35e09425735ebc9d38ec0793f6de95fc
Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
Compression Method: null (0)
Extensions Length: 46
```

d. Test Observations:

- It should be ensured that a successful TLS handshake is performed between the Tester device and the DUT.
- It should be also ensured that the DUT should not accept any request in plain HTTP like in 3rd execution step.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_HTTPS		



2.11.2 TSTP for Evaluation of Webserver logging

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.2
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) **<ITSAR Section No & Name>** Section 11: Web Server
- 2) **<Security Requirement No & Name >** 2.11.2 Webserver logging
- 3) **<Requirement Description: >** Access to the SMF webserver (for both successful as well as failed attempts) shall be logged

by SMF.

The web server log shall contain the following information:

- Access timestamp - Source (IP address)
- Account (if known)
- Attempted login name (if the associated account does not exist)
- Relevant fields in http request. The URL should be included whenever possible.
- Status code of web server response

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.2.5.2]

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: `nmap -p- 172.18.0.8 -sC -sV` (IP address of DUT)

```

[ashwini@ashwini-news1ab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_   Not valid before: 2023-05-17T08:39:01
|_   Not valid after: 2024-05-16T08:39:01
|_   tls-alpn:
|_     http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

a. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built:   2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in : ***"/etc/apache2/apache2.conf"***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- Command used: ***grep -I log /etc/apache2/apache2.conf*** (To check where the Log files are present)
 - Error Log (Contains all the errors and failed attempts)
 - CustomLog (Contains all the access logs)

```
amf@localhost $ grep -I log /etc/apache2/apache2.conf
# configuration, error, and log files are kept.
# ErrorLog: The location of the error log file.
# logged here. If you *do* define an error logfile for a <VirtualHost>
# container, that host's errors will be logged there and not here.
ErrorLog "/var/log/apache2/error.log"
CustomLog "/var/log/apache2/access.log" common
# LogLevel: Control the severity of messages logged to the error_log.
# It is also possible to configure the log level for particular modules, e.g.
```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6) Preconditions

- Network Product documentation which contains information on log file location and procedure to access it.
- Tester has the necessary privileges to access the log files.
- Test environment with a Web Browser.

7) **Test Objective:** Verify that all accesses to the webserver are logged with the required information.

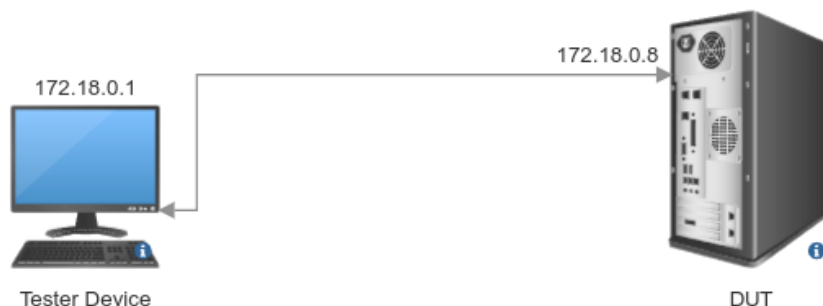
8) Test Plan:

8.1 Number of Test Scenarios:

8.1.1 Test scenario for Web server logging

- Check if proper logging is done
- Required information is logged or not

8.2 Test Setup Diagram:



8.3 **Tools Used:**

- Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 **Test Execution Steps**

- Power up the testbed
- The tester tries to login to the webserver using the correct and incorrect login credentials.
- The tester verifies whether the login attempts were logged correctly with all the required information.

9) **Expected Results for Pass:** All webserver events are logged with all of the required information.

10) **Expected Format of Evidence:** Testing report contains copies of the log file showing the captured information.

11) **Test Execution:**

➤ **Test Case Number:** 01

a) **Test Case Name:** TC_WEBSERVER_LOGGING

b) **Test Case Description:** Verify that all accesses to the webserver are logged with the required information

The web server log shall contain the following information:

- Access timestamp - Source (IP address)
- Account (if known)
- Attempted login name (if the associated account does not exist)
- Relevant fields in http request. The URL should be included whenever possible.
- Status code of web server response

c) **Execution Steps:** The tester shall open any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.

- URL: http://<IP address of DUT>

And enter the right credentials.

Sign in

http://172.18.0.8

Your connection to this site is not private

Username

Password

With the proper privilege in the DUT while accessing the Log file we can see the following log info.

- Command used: **cat /var/log/apache2/access.log**

```
anf@localhost $ cat /var/log/apache2/access.log
172.18.0.1 - - [22/May/2023:11:45:17 +0000] "GET / HTTP/1.1" 401 729
172.18.0.1 - - [22/May/2023:11:45:19 +0000] "GET / HTTP/1.1" 401 728
172.18.0.1 - - [22/May/2023:11:45:19 +0000] "GET /favicon.ico HTTP/1.1" 401 728
172.18.0.1 - - [22/May/2023:11:45:26 +0000] "GET / HTTP/1.1" 401 729
172.18.0.1 - admin [22/May/2023:11:46:55 +0000] "GET / HTTP/1.1" 304 248
172.18.0.1 - admin [22/May/2023:11:46:56 +0000] "GET /favicon.ico HTTP/1.1" 404 488
```

Access IP	Valid User	Access Timestamp	Method & URL	Status Code
172.18.0.1	admin	[22/May/2023:11:46:56 +0000]	"GET /favicon.ico HTTP/1.1"	404 488

Again, the tester opens any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.

- URL: http://<IP address of DUT>

And enter the wrong credentials (wrong password/wrong username)

Sign in

http://172.18.0.8

Your connection to this site is not private

Username

Password

With the proper privilege in the DUT while accessing the Error Log file we can see the following log info.

- Command used: ***cat /var/log/apache2/error.log***

```
anf@localhost $ cat /var/log/apache2/error.log
[Mon May 22 12:04:52.281758 2023] [auth_basic:error] [pid 13:tid 1398091312686784] [client 172.18.0.1:59798] AH01618: user no_admin not found: /
[Mon May 22 12:04:57.117851 2023] [auth_basic:error] [pid 13:tid 1398091170076352] [client 172.18.0.1:59798] AH01617: user admin: authentication failure for "/": Password Mismatch
AH01617: user admin: authentication failure for "/": Password Mismatch
AH01618: user no_admin not found: /
```

d) **Test Observations:**

- It should be ensured that any access/error/events to the DUT Web-Server the logs are filed with the information as per the ITSAR clause.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_WEBSERVER_LOGGING		



2.11.3 TSTP for Evaluation of HTTPS Input Validation

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.3
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) <ITSAR Section No & Name> Section 11: Web Server
- 2) <Security Requirement No & Name > 2.11.3 HTTPS input validation
- 3) <Requirement Description: >

The SMF shall have a mechanism in place to ensure that web application inputs are not vulnerable to command injection or cross-site scripting attacks. SMF shall validate, filter, escape, and encode user-controllable input before it is placed in output that is used as a web page that is served to other users.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.2.5.4]

4) **DUT Confirmation Details:** *Securing Networks*

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6) **Preconditions**

- The Vendor must provide a Certificate stating that the Web Application is tested and free from command injection or cross-site scripting attacks.
- Network Product documentation which contains information on log file location and procedure to access it.
- Tester has the necessary privileges to access the log files.
- Test environment with a Web Browser.
- Knowledge about XSS and Command Injection attacks.

7) **Test Objective:** The DUT shall have a mechanism in place to ensure that web application inputs are not vulnerable to command injection or cross-site scripting attacks

8) **Test Plan:**

8.1 **Number of Test Scenarios:**

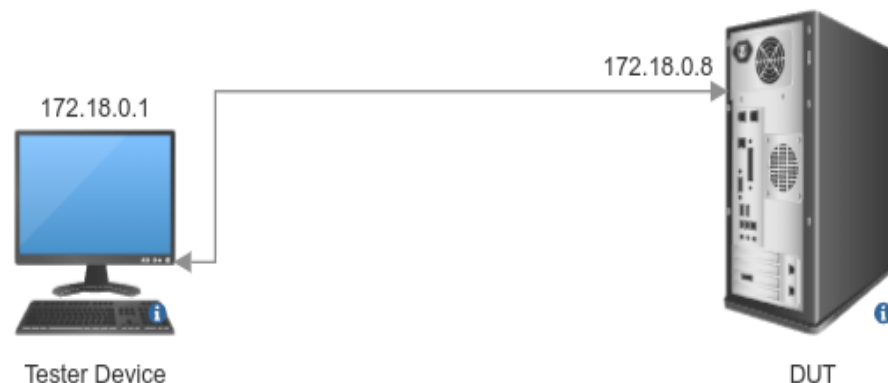
8.1.1 **Test Scenarios for XSS attack**

- Manual check via Cheat Sheet
- Automated check via tool

8.1.2 **Test Scenarios for Command Injection attack**

- Manual check via Cheat Sheet
- Automated check via tool

8.2 **Test Setup Diagram:**



8.3 **Tools Used:**

- Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.
- XSSStrike (<https://github.com/s0md3v/XSSStrike>)
- Commix (<https://github.com/commixproject/commix>)

8.4 **Test Execution Steps**

- Power up the testbed
- The tester manually tries to find the inputs in the Web Application.

- Perform manual test on inputs for XSS and Command Injection vulnerability based on cheat sheets.
- The tester using automated tools like XSSStrike and Commix to find Command Injection vulnerability.

9) **Expected Results for Pass:** The Web Application must be free from XSS and Command Injection vulnerability.

10) **Expected Format of Evidence:** Testing report contains copies of the log file showing the captured information.

11) **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC_XSS_TESTING

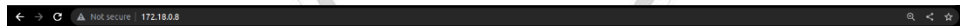
b. **Test Case Description:** Verify that all the input in the WEB Application is free from XSS attack.

- i. Manual approach
- ii. Automated approach

c. **Execution Steps:**

The tester shall open any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.

- URL: `http://<IP address of DUT>`



DUT - WEB SERVER

- Find all the input in the WEB Application/Page Manual Approach
- a. Use the below cheat sheet for XSS and try it in the input and observe the behaviour.
- (https://gist.githubusercontent.com/kurobeats/9a613c9ab68914312cbb415134795b45/raw/c24dd91dd91c324ae5c28b124aa4d379dbcb8e59/xss_vectors.txt)


```
%253Cscript%253Ealert('XSS')%253C%252Fscript%253E
<IMG SRC=x onload="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onafterprint="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onbeforeprint="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onbeforeunload="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onerror="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onhashchange="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onload="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onmessage="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x ononline="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onoffline="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onpagehide="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onpageshow="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onpopstate="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onresize="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onstorage="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onunload="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onblur="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onchange="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x oncontextmenu="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x oninput="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x oninvalid="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onreset="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onsearch="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onselect="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onsubmit="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onkeydown="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onkeypress="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onkeyup="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onclick="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x ondblclick="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onmousedown="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onmousemove="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onmouseout="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onmouseover="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onmouseup="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onmousewheel="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onwheel="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x ondrag="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x ondragend="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x ondragenter="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x ondragleave="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x ondragover="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x ondragstart="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x ondrop="alert(String.fromCharCode(88,83,83))">
<IMG SRC=x onscroll="alert(String.fromCharCode(88,83,83))">
```

- (A snapshot of few XSS payloads)

Automated Approach.

- Command used: **xsstrike -crawl -u <URL of WEB Application>**

```
[ashwini@ashwini-newslab]--[XSStrike]
$ python3 xsstrike.py --crawl -u http://172.18.0.8/

XSStrike v3.1.5

[~] Crawling the target
[!] Progress: 1/1

[ashwini@ashwini-newslab]--[XSStrike]
$
```

- (Here we can see the tool has not detected any vulnerable object)

```

[ashwini@ashwini-news-lab]--[~/XSStrike]
$ python3 xsstrike.py --crawl -u http://172.18.0.8/

XSStrike v3.1.5

[~] Crawling the target
[+] Potentially vulnerable objects found at http://172.18.0.8/
-----
6   output.innerHTML = input.value;
-----
[!] Progress: 1/1

```

- (Here we can see the tool has detected a vulnerable object this must be manually checked and if found vulnerable the test case will fail)

d. **Test Observations:**

- It should be ensured that any input on the WEB Application should not be vulnerable to XSS attacks.

➤ **Test Case Number: 02**

a. **Test Case Name:** TC_CI_TESTING

b. **Test Case Description:** Verify that all the input in the WEB Application is free from Command Injection attack.

- Manual approach
- Automated approach

c. **Execution Steps:**

The tester shall open any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.

- URL: `http://<IP address of DUT>`



DUT - WEB SERVER

- Find all the input in the WEB Application/Page
 - Manual Approach
- Use the below cheat sheet for Command Injection and try it in the input and observe the behaviour.
- (<https://hackersonlineclub.com/command-injection-cheatsheet/>)

```

&lt;!--#exec%20cmd=&quot;/bin/cat%20/etc/passwd&quot;;--&gt;
&lt;!--#exec%20cmd=&quot;/bin/cat%20/etc/shadow&quot;;--&gt;
&lt;!--#exec%20cmd=&quot;/usr/bin/id;--&gt;
&lt;!--#exec%20cmd=&quot;/usr/bin/id;--&gt;
/index.html|id|
;id;
;id
;netstat -a;
;id;
|id
|/usr/bin/id
|id|
|/usr/bin/id|
|/usr/bin/id|
|id;
|/usr/bin/id;
;id|
;|/usr/bin/id|
\n/bin/ls -al\n
\n/usr/bin/id\n
\nid\n
\n/usr/bin/id;
\nid;
\n/usr/bin/id|
\nid|
;/usr/bin/id\n
;id\n
|usr/bin/id\n
\nid\n

```

- (A snapshot of few Command Injection payloads)

Automated Approach.

- Command used: **<Read the utility documentation to get information about the usage>**

```

ashwini@ashwini-news-lab:~/commix$ python3 commix.py -u http://localhost:8888/ci.php
v3.8-dev#42
https://commixproject.com
(commixproject)

Automated All-in-One OS Command Injection Exploitation Tool
Copyright © 2014-2023 Anastasios Stasinopoulos (@ancst)

(!!) Legal disclaimer: Usage of commix for attacking targets without prior mutual consent is illegal. It is the end user's responsibility
to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage cau
sed by this program.

[16:40:23] [info] Testing connection to the target URL.
[16:40:23] [info] Performing identification checks to the target URL.
[16:40:23] [critical] No parameter(s) found for testing on the provided target URL. You are advised to rerun with '--crawl=2'.

```

- (Here we can see the tool has not detected any vulnerable object)

```

ashwini@ashwini-news1ab [~/commix]
$ python3 commix.py --url="http://192.168.17.2/vulnerabilities/exec/#" --data="ip=127.0.0.1&Submit=submit" --cookie="security=low; PHPSESSID=3pmaechpniqjjb56qgf4udc62"

v3.8-dev#42
https://commixproject.com
@commixproject

Automated All-in-One OS Command Injection Exploitation Tool
Copyright © 2014-2023 Anastasios Stasinopoulos (@ancst)

(!) Legal disclaimer: Usage of commix for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program.

[16:36:57] [Info] Testing connection to the target URL.
Got a redirect to 'http://192.168.17.2/vulnerabilities/exec/'. Do you want to follow? [Y/n] >
[16:37:15] [Info] Following redirection to 'http://192.168.17.2/vulnerabilities/exec/'.
[16:37:15] [Info] Performing identification checks to the target URL.
[16:37:18] [Warning] Target's estimated response time is 3 seconds. That may cause serious delays during the data extraction procedure and/or possible corruptions over the extracted data.
[16:37:18] [Info] Setting POST parameter 'ip' for tests.
[16:37:27] [Info] Heuristic (basic) tests shows that POST parameter 'ip' might be injectable (possible OS: 'Unix-like').
[16:37:38] [Info] Testing the (results-based) classic command injection technique.
[16:37:38] [Info] POST parameter 'ip' appears to be injectable via (results-based) classic command injection technique.
_ 127.0.0.1;echo CJZATJ$((56+49))$(echo CJZATJ)CJZATJ
POST parameter 'ip' is vulnerable. Do you want to prompt for a pseudo-terminal shell? [Y/n] > Y
Pseudo-Terminal Shell (type '?' for available options)
commix(os_shell) > ls
help index.php source

```

- (Here we can see the tool has detected a vulnerable object this must be manually checked and if found vulnerable the test case will fail)

d. Test Observations:

- It should be ensured that any input on the WEB Application should not be vulnerable to Command Injection attacks.

12) Test Case Result:

SL. No	TESTCASE NAME	PASS/FAIL	Remarks
1	TC_XSS_TESTING		
2	TC_CI_TESTING		

2.11.4 TSTP for No System Privileges

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.4
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 11: Web Server
2. **<Security Requirement No & Name >** 2.11.4 No system privileges
3. **<Requirement Description: >**

No DUT web server processes shall run with system privileges.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.2]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
root@os-controller $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.17.2 netmask 255.255.255.0 broadcast 192.168.17.255
    ether 02:42:c0:a8:11:02 txqueuelen 0 (Ethernet)
    RX packets 6400 bytes 27172981 (27.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3590 bytes 301838 (301.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Command used: **openstack -version** (To find version information of OpenStack)

```
root@os-controller $ openstack --version
openstack 4.0.2
```

Command used: `cat /etc/os-release` (To get OS information)

```
root@os-controller $ cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.1 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.1 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=jammy
```

5. DUT Configuration:

To get the hash of OS image is using Virtual Machine

Command used: `sha256sum <path to OS image>` (To get hash/digest of OS Image)

```
root@os-controller $ sha256sum ubuntu-22.04.1-amd64.iso
10f19c5b2b8d6db711582e0e27f5116296c34fe4b313ba45f9b201a5007056cb  ubuntu-22.04.1-amd64.iso
```

To get the hash of OS if using Docker/Kubernetes

Command used: `docker images <image name> --digests` (To get hash/digest of Container Image)

```
root@os-controller $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 14be0685b768 sha256:c9820a44b950956a790c354700c1166a7ec648bc0d215fa438d3a339812f1d01
ubuntu focal 14be0685b768 sha256:c9820a44b950956a790c354700c1166a7ec648bc0d215fa438d3a339812f1d01
ubuntu 18.04 5d2df19066ac sha256:a3765b4d74747b5e9bdd03205b3fbc4fa19a02781c185f97f24c8f4f84ed7bbf
ubuntu bionic 5d2df19066ac sha256:a3765b4d74747b5e9bdd03205b3fbc4fa19a02781c185f97f24c8f4f84ed7bbf
ubuntu latest 58db3edaf2be sha256:9a0bdde4188b896a372804be2384015e90e3f84906b750c1a53539b585fbbe7f
ubuntu <none> a8780b506fa4 sha256:4b1d0c4a2d2aaf63b3711f34eb9fa89fa1bf53dd6e4ca954d47caebca4005c2
```

Securing Networks

6. Preconditions

- Network product documentation containing information about web server configuration and management tool which maybe command line, GUI or any other as specified by Vendor.
- The tester needs to have administrative privileges over DUT.

7. Test Objective:

Verify that the Web server is not run under system privileges.

8. Test Plan:

8.1 Number of Test Scenarios:

8.1.1 Test Scenario for System Privileges for Web Server Process

8.2 Test Setup Diagram:



8.3 **Tools Used:** Default DUT configuration and management tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 **Test Execution Steps**

- Power up the testbed.
- The tester shall access the list of running processes in the DUT with details of the user under which process is running
- The tester shall ensure the web server process is not running under user with system privileges or Root User.
- The tester shall also check the default user under which web server process can be run i.e the default user under which web server runs does not have system privileges or is not root user.

9. **Expected Results for Pass:**

- None of the web server processes running are running with system privileges or root user.
- System settings have been found correctly set to ensure that no web server processes will run with system privileges.

Securing Networks

10. **Expected Format of Evidence:**

- Screenshots of Running processes and user under which web server has started.
- Screenshots of default user under which web server shall run every time.

11. **Test Execution:**

➤ **Test Case Number:** 01

- a. **Test Case Name:** TC_NO_SYSTEM_PRIVILEGES_WEB_SERVER_PROCESSES
- b. **Test Case Description:** None of the processes running are running with system privileges or root user.
- c. **Execution Steps:**
 1. The tester shall use the following command or corresponding command in their server or GUI settings to obtain the list of running processes in the DUT and ensure that the web

server process is not running as any user with system privileges or as root user - “**ps aux | grep apache**” (In case Apache httpd is used)

```
amf@localhost $ ps aux | grep apache
www-data    13  0.0  0.0 1997948 4276 ?        Sl   May22   0:00 /usr/sbin/apache2 -DFOREGROUND -k start
```

2. In the above image user www-data is known default user used by Apache to run the web server. Please check with vendor in your case what shall be the default user group and ensure it doesn't have system privileges.

3. If the user is found to be root user or user with system privileges, test fails.

```
amf@localhost $ ps aux | grep apache
root        11  0.0  0.0   6744   5428 ?        S    May22   0:02 /usr/sbin/apache2 -DFOREGROUND -k start
root        833  0.0  0.0   3460   1484 pts/0    S+   05:05   0:00 grep --color=auto apache
```

In the above image we see that user is 'root'. In this case test fails

- d. **Test Observations:** It is ensured that none of the web server processes running are not running with system privileges or root user.

➤ **Test Case Number:** 02

- a. **Test Case Name:** TC_NO_SYSTEM_PRIVILEGES_WEB_SERVER_DEFAULT
- b. **Test Case Description:** To ensure system settings are correctly set to ensure that no web server processes will run with system privileges.
- c. **Execution Steps:**
 1. The tester shall use the following command to obtain the default user under which web server in DUT is started and run - “**cat /etc/apache2/envvars | grep -I user**” (In case Apache httpd is used in command line). Corresponding command or GUI option according to vendor shall be used to check the same.

```
amf@localhost $ cat /etc/apache2/envvars | grep -i user
export APACHE_RUN_USER=www-data
```

In the above image we can see that www-data (user with no system privileges) is the default user under which server starts (To check if the above user does not have root privileges following file can be checked in Linux to make sure www-data does not have root privileges: \$ cat /etc/sudoers)

- d. **Test Observations:**

It is ensured that system settings have been found correctly set to ensure that no web server processes will run with system privileges.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_SYSTEM_PRIVILEGES_WEB_SERVER_PROCESSES		
2	TC_NO_SYSTEM_PRIVILEGES_WEB_SERVER_DEFAULT		



2.11.5 TSTP for No unused HTTPS methods

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.5
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1) <ITSAR Section No & Name> Section 11: Web Server

2) <Security Requirement No & Name > 2.11.5 No unused HTTPS methods

3) <Requirement Description: >HTTPS methods that are not required for SMF operation shall be deactivated.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.3]

4) DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: **nmap -p- 172.18.0.8 -sC -sV** (IP address of DUT)

```

[ashwini@ashwini-newslab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_ Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_ Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_ Not valid before: 2023-05-17T08:39:01
|_ Not valid after: 2024-05-16T08:39:01
|_ tls-alpn:
|_ http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

b. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built:   2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in: ***"/etc/apache2/apache2.conf"***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- Command used: ***grep -B 2 -A 2 -I "AllowMethods" <path to the apache2 config>*** (To check Apache allows which Http methods)


```
amf@localhost $ grep -B 2 -A 2 -I "AllowMethods" /etc/apache2/apache2.conf
<Location "/">
  AllowMethods GET POST OPTIONS
</Location>
```

(Above result states that **AllowMethods** set to enable GET POST and OPTIONS methods)

AllowMethods Directive

Description:	Restrict access to the listed HTTP methods
Syntax:	AllowMethods <i>reset HTTP-method [HTTP-method]...</i>
Default:	AllowMethods reset
Context:	directory
Status:	Experimental
Module:	mod_allowmethods

- Command used: **grep -B 2 -A 2 -I "Limit" <path to the apache2 config>** (To check Apache allows which Http methods)

```
amf@localhost $ grep -A 1 -I "Limit" /etc/apache2/apache2.conf
<Limit GET HEAD POST PUT DELETE OPTIONS>
  Require all granted
</Limit>
</Directory>
```

(Above result states that **Limit** set to enable http methods)

<Limit> Directive

Description:	Restrict enclosed access controls to only certain HTTP methods
Syntax:	<Limit <i>method [method] ...</i> > ... </Limit>
Context:	directory, .htaccess
Override:	AuthConfig, Limit
Status:	Core
Module:	core

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6) **Preconditions**

- The tester has needed administrative privileges.
- A tester machine is available.
- Test environment with a provision to access the DUT.
- Vendor list of Enabled and necessary methods in the DUT Web-Server.

7) **Test Objective:** Verify that the Web server has deactivated all HTTP methods that are not required.

8) **Test Plan:**

8.1 **Number of Test Scenarios:**

8.1.1 **Test Scenario if OPTIONS is enabled**

8.1.2 **Test Scenario if OPTIONS is disabled:-** Custom Script to be used to check for enabled Methods

8.2 **Test Setup Diagram:**



8.3 **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 **Test Execution Steps**

- Power up the testbed
- The tester tries to check the available HTTP Methods in webserver.
- The tester verifies whether the only required HTTP Methods are enabled or not.
- If the OPTIONS method is not enabled in WEB-SERVER, the manual checking via script is to be used.

9) **Expected Results for Pass:** System settings and configurations have been found adequately set, in all Web components of the system, to ensure that unneeded HTTP methods are deactivated.

10) Expected Format of Evidence: Testing report contains copies of the log file showing the captured information.

11) **Test Execution:**

➤ **Test Case Number:** 01

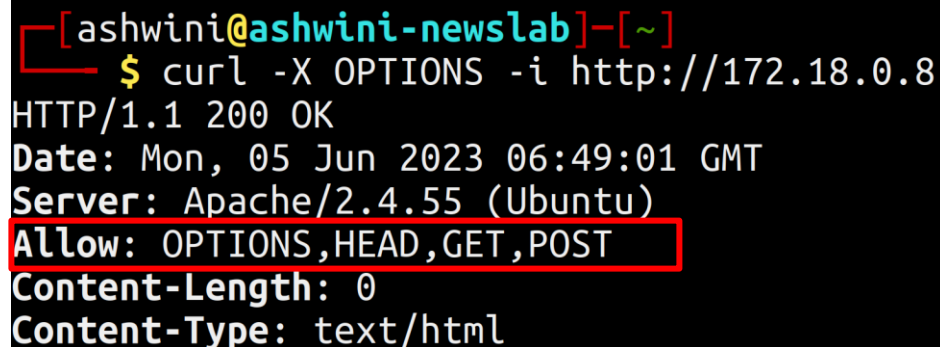
a) **Test Case Name:** TC_NO_UNUSED_HTTP_METHODS

b) **Test Case Description:** Verify that the Web server has deactivated all HTTP methods that are not required.

c) **Execution Steps:**

The tester shall open any terminal or any tool to access DUT shell

- Command used: ***curl -X OPTIONS -i <IP address of DUT>***



```
[ashwini@ashwini-news-lab]~$ curl -X OPTIONS -i http://172.18.0.8
HTTP/1.1 200 OK
Date: Mon, 05 Jun 2023 06:49:01 GMT
Server: Apache/2.4.55 (Ubuntu)
Allow: OPTIONS, HEAD, GET, POST
Content-Length: 0
Content-Type: text/html
```

- The tester will compare the vendor list of enabled and necessary methods.
 - If any method other than the vendor list is present the test case will fail.
 - Here we can see allowed HTTP methods are **HEAD, GET, OPTIONS, POST**
 - All this must not be present here only allowed and necessary methods must be allowed.
- d) **Test Observations:** It should be ensured that HTTPS methods that are not required for DUT operation shall be deactivated.

➤ **Test Case Number:** 02

a. **Test Case Name:** TC_NO_UNUSED_HTTP_METHODS_OPTIONS_DISABLED

b. **Test Case Description:** Verify that the Web server has deactivated all HTTP methods that are not required if OPTIONS method is not enabled.

c. **Execution Steps:**

1. The tester shall open any terminal or any tool to access DUT shell

- Command used: ***python3 WebServer-Methods.py <IP address of DUT>***

```
[ashwini@ashwini-news-lab]~$ python3 WebServer-Methods.py http://172.18.0.8
[GET]: Enabled (200)
[POST]: Enabled (200)
[PUT]: Disabled (405)
[DELETE]: Disabled (405)
[PATCH]: Disabled (405)
[HEAD]: Enabled (200)
[OPTIONS]: Enabled (200)
[TRACE]: Disabled (405)
[CONNECT]: Enabled (400)
```

- The tester will compare the vendor list of enabled and necessary methods.
- If any method other than the vendor list is present the test case will fail.

d. **Test Observations:** It should be ensured that HTTPS methods that are not required for DUT operation shall be deactivated.

12) **Test** Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_UNUSED_HTTP_METHODS		
2	TC_NO_UNUSED_HTTP_METHODS_OPTIONS_DISABLED		

Securing Networks

2.11.6 TSTP for No unused add-ons

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.6
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) <ITSAR Section No & Name> Section 11: Web Server
- 2) <Security Requirement No & Name > 2.11.6 No unused add-ons
- 3) <Requirement Description: >

All optional add-ons and components of the web server shall be deactivated if they are not required for SYSTEM operation. In particular, CGI or other scripting components, Server Side Includes (SSI), and WebDAV shall be deactivated if they are not required.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.4]

4) DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
root@os-controller $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.17.2 netmask 255.255.255.0 broadcast 192.168.17.255
    ether 02:42:c0:a8:11:02 txqueuelen 0 (Ethernet)
    RX packets 6400 bytes 27172981 (27.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3590 bytes 301838 (301.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Command used: **openstack --version** (To find version information of OpenStack)

```
root@os-controller $ openstack --version
openstack 4.0.2
```

Command used: `cat /etc/os-release` (To get OS information)

```
root@os-controller $ cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.1 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.1 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=jammy
```

5) DUT Configuration:

To get the hash of OS image is using Virtual Machine

Command used: `sha256sum <path to OS image>` (To get hash/digest of OS Image)

```
root@os-controller $ sha256sum ubuntu-22.04.1-amd64.iso
18f19c5b2b8d6db711582e0e27f5116296c34fe4b313ba45f9b201a5007056cb  ubuntu-22.04.1-amd64.iso
```

To get the hash of OS if using Docker/Kubernetes

Command used: `docker images <image name> --digests` (To get hash/digest of Container Image)

```
root@os-controller $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 14be0685b768 sha256:c9820a44b950956a790c354700c1166a7ec648bc0d215fa438d3a339812f1d01
ubuntu focal 14be0685b768 sha256:c9820a44b950956a790c354700c1166a7ec648bc0d215fa438d3a339812f1d01
ubuntu 18.04 5d2df19066ac sha256:a3765b4d74747b5e9bdd03205b3fbc4fa19a02781c185f97f24c8f4f84ed7bbf
ubuntu bionic 5d2df19066ac sha256:a3765b4d74747b5e9bdd03205b3fbc4fa19a02781c185f97f24c8f4f84ed7bbf
ubuntu latest 58db3edaf2be sha256:9a0bdd4e4188b096a372804be2384015e90e3f84906b750c1a53539b585fbbe7f
ubuntu <none> a8780b506fa4 sha256:4b1d0c4a2d2aaf63b37111f34eb9fa89fa1bf53dd6e4ca954d47caebca4005c2
```

Securing Networks

6) Preconditions

- The vendor has supplied a list of add-ons or scripting tools for Web server components needed for system operation, and that therefore need to be exempted from the test investigation.
- The tester has administrative privileges.
- A tester machine is available.

7) Test Objective: To verify that the Web server has deactivated unneeded add-ons and unneeded scripting components.

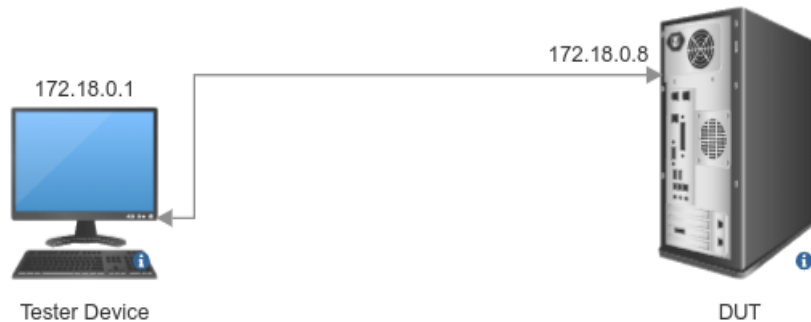
8) Test Plan:

8.1 Number of Test Scenarios:

8.1.1 Test Scenario for Unused Add-ons

This test scenario will check for the add-ons using DUT command.

8.2 Test Setup Diagram:



8.3 **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 Test Execution Steps

- Power up the testbed
- Check that the web server is only running and listening on known ports (e.g., tcp port 80 and/or 443). Check that
- CGI or other scripting components, Server Side Includes (SSI), and WebDAV are deactivated if they are not required.
- The tester verifies that nothing else has been installed than the web server.
- The tester verifies that relevant system settings and configurations are correct to ensure fulfilment of the requirement.
- The tester scans the Webserver using a suitable tool (e.g., Nikto) and verifies the tool report for availability of any default content.

9) **Expected Results for Pass:** System settings and configurations have been found adequately set, in all Web components of the system, to ensure that all unneeded add-ons or script components are deactivated.

10) **Expected Format of Evidence:** Testing report contains copies of the log file showing the captured information.

11) Test Execution:

➤ **Test Case Number:** 01

a) **Test Case Name:** TC_NO_UNUSED_ADD-ONS

b) **Test Case Description:** To verify that the Web server has deactivated unneeded

add-ons and unneeded scripting components.

c) **Execution Steps:**

The tester shall open any terminal or any tool to access DUT shell

- Command used: ***apache2ctl -M***

```
amf@localhost $ apache2ctl -M
Loaded Modules:
  core_module (static)
  so_module (static)
  watchdog_module (static)
  http_module (static)
  log_config_module (static)
  logio_module (static)
  version_module (static)
  unixd_module (static)
  access_compat_module (shared)
  alias_module (shared)
  auth_basic_module (shared)
  authn_core_module (shared)
  authn_file_module (shared)
  authz_core_module (shared)
  authz_host_module (shared)
  authz_user_module (shared)
  autoindex_module (shared)
  deflate_module (shared)
  dir_module (shared)
  env_module (shared)
  filter_module (shared)
  mime_module (shared)
  mpm_event_module (shared)
  negotiation_module (shared)
  reqtimeout_module (shared)
  setenvif_module (shared)
  status_module (shared)
  allowmethods module (shared)
```

- The tester manually checks for the vendor-supplied list of add-ons or scripting tools for Web server components needed for system operation other than those that must not be present.

The tester opens any terminal or any tool to access DUT shell

- Command used: ***apache2ctl -M***

```

amf@localhost $ apache2ctl -M
Loaded Modules:
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
reqtimeout_module (shared)
setenvif_module (shared)
status_module (shared)
allowmethods_module (shared)
mod_actions (shared)
mod_authnz_fcgi (shared)
mod_cgi (shared)
mod_cgid (shared)
mod_suexec (shared)
mod_dav (shared)
mod_dav_fs (shared)
mod_dav_lock (shared)
mod_include (shared)

```

- Here we can see allowed enabled modules that should not be there because they are WEBDAV, CGI, SSI (mod_actions, mod_authnz_fcgi, mod_cgi, mod_cgid, mod_suexec, mod_dav, mod_dav_fs, mod_dav_lock, mod_include)

(Clickable link so for more info)

- All this must not be present here only allowed and necessary add-ons must be allowed.

d) **Test Observations:** System settings and configurations have been found adequately set, in all Web components of the system, to ensure that all unneeded add-ons or script components are deactivated.

12) **Test** Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_UNUSED_ADD-ONS		

2.11.7 TSTP for No compiler, interpreter, or shell via CGI or other server-side scripting

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.7
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) **<ITSAR Section No & Name>** Section 11: Web Server
- 2) **<Security Requirement No & Name >** 2.11.7 No compiler, interpreter, or shell via CGI or other server-side scripting
- 3) **<Requirement Description: >**If CGI (Common Gateway Interface) or other scripting technology is used, the CGI directory or other corresponding scripting directory shall not include compilers or interpreters.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.5]

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: `nmap -p- 172.18.0.8 -sC -sV` (IP address of DUT)

```

[ashwini@ashwini-newslab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_   Not valid before: 2023-05-17T08:39:01
|_   Not valid after: 2024-05-16T08:39:01
|_   tls-alpn:
|_     http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

c. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built:   2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in: ***/etc/apache2/apache2.conf***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- Command used: ***grep -I "ExecCGI" /etc/apache2/apache2.conf*** (To check CGI is enabled or not)


```
amf@localhost $ grep -I "ExecCGI" /etc/apache2/apache2.conf
Options +ExecCGI
```

(Above result states that CGI is enabled)

- Command used: **grep -C2 -I "ExecCGI" /etc/apache2/apache2.conf** (To check for which directory CGI is enabled)

```
amf@localhost $ grep -C2 -I "ExecCGI" /etc/apache2/apache2.conf

<Directory "/var/www/html/cgi-enabled">
  Options +ExecCGI
  AddHandler cgi-script .cgi .pl .py .rb
</Directory>
```

- Same has to be checked in for Vhost files

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6) Preconditions

- The tester has administrative privileges
- A tester machine is available.
- Test environment with a Terminal.

- 7) **Test Objective:** To verify that there are no compilers, interpreters, or shell accessible via CGI or other scripting components.

8) Test Plan:

8.1 Number of Test Scenarios:

8.1.1 Test Scenario for CGI

This test scenario is regarding CGI and check for compiler/interpreter or shell in CGI

8.2 Test Setup Diagram:



8.3 **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 Test Execution Steps

- Power up the testbed
- The tester tries to access the Shell of Web Server.
- Consult the web server configuration to identify all directories used for CGI or other scripting components.
- In case the DUT is using other than CGI they must give the information about different technology used and further tests should be performed on that technology. (This document only deals with CGI Scripts)
- The tester manually checks that there are no compilers or interpreters (e.g., PERL® interpreter, PHP interpreter/compiler, Tcl interpreter/compiler or operating system shells) in the directory/directories used for CGI or for other scripting tools (including PERL®, PHP, and others).
- The tester will run some web scanning tools like Nikto, to get the detailed information and verify that there are no compilers or interpreters (e.g., PERL® interpreter, PHP interpreter/compiler, Tcl interpreter/compiler or operating system shells) in the directory/directories used for CGI or for other scripting tools (including PERL®, PHP, and others).

9) **Expected Results for Pass:** There are no compilers, interpreters or shells in directories accessible via CGI or other scripting components.

10) **Expected Format of Evidence:** Log files and screen shots of test executions.

11) **Test Execution:**

➤ **Test Case Number:** 01

a) **Test Case Name:** TC_NO_COMPILER_FOR_CGI

b) **Test Case Description:** There are no compilers, interpreters, or shells in directories accessible via CGI or other scripting components.

c) **Execution Steps:**

The tester shall open any terminal emulator to access the DUT shell and use the following command.

- Command used: ***ls <path of the CGI-BIN directory>***

```
amf@localhost $ ls /var/www/html/cgi-enabled/  
test.cgi test.py
```

- We can see that the directory does not contain any compiler/interpreter, shell.

The tester shall open any terminal emulator and use the following command.

- Command used: ***ls <path of the CGI-BIN directory>***

```
amf@localhost $ ls  
bash python3 test.cgi test.py
```

- We can see that the directory does contain bash shell, python3 interpreter.

- (This should not be present)

d) **Test Observations:** There are no compilers, interpreters or shells in directories accessible via CGI or other scripting components.

12)Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_COMPILER_FOR_CGI		

2.11.8 TSTP for No CGI or other scripting for uploads

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.8
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) **<ITSAR Section No & Name>** Section 11: Web Server
- 2) **<Security Requirement No & Name >** 2.11.8 No CGI or other scripting for uploads
- 3) **<Requirement Description: >** If CGI or other scripting technology is used, the associated CGI/script directory shall not be used for uploads.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.6]

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) **DUT Configuration:**

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: `nmap -p- 172.18.0.8 -sC -sV` (IP address of DUT)

```

[ashwini@ashwini-newslab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_   Not valid before: 2023-05-17T08:39:01
|_   Not valid after: 2024-05-16T08:39:01
|_   tls-alpn:
|_     http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

d. For Apache httpd/CGI:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built:   2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in: ***/etc/apache2/apache2.conf***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- Command used: ***grep -I "ExecCGI" /etc/apache2/apache2.conf*** (To check CGI is enabled or not)


```
amf@localhost $ grep -I "ExecCGI" /etc/apache2/apache2.conf
Options +ExecCGI
```

(Above result states that CGI is enabled)

- Command used: **grep -C2 -I "ExecCGI" /etc/apache2/apache2.conf** (To check for which directory CGI is enabled)

```
amf@localhost $ grep -C2 -I "ExecCGI" /etc/apache2/apache2.conf

<Directory "/var/www/html/cgi-enabled">
  Options +ExecCGI
  AddHandler cgi-script .cgi .pl .py .rb
</Directory>
```

- Same has to be checked in for Vhost files

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6) Preconditions

- The tester has administrative privileges
- A tester machine is available.
- Test environment with a Terminal.
- Test Environment with a Browser
- If the web server is configured with CGI/Scripting on, this test applies
- The tester checks the vendor documentation for the "upload directory"

7) **Test Objective:** To test whether the upload directory is equal to the CGI/Scripting directory.

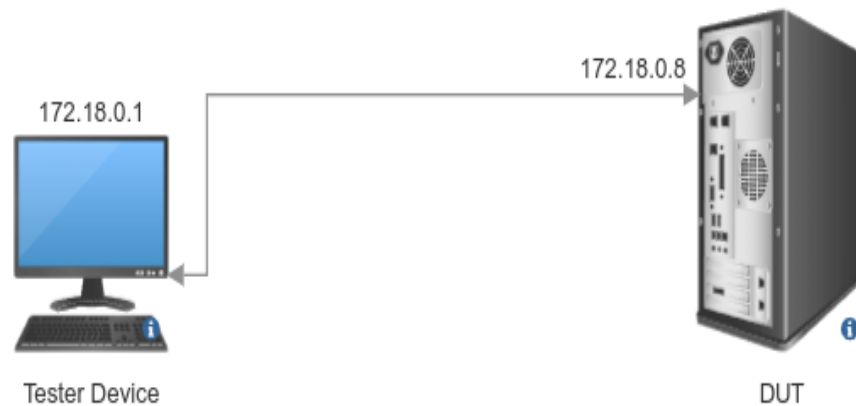
8) Test Plan:

8.1 Number of Test Scenarios:

8.1.1 Test Scenario for CGI

This test scenario is regarding CGI

8.2 Test Setup Diagram:



8.3 **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 **Test Execution Steps**

- Power up the testbed
- The tester tries to access the Shell of Web Server.
- Consult the web server configuration to identify all directories used for CGI or other scripting components.
- In case the DUT is using other than CGI they must give the information about different technology used and further tests should be performed on that technology. (This document only deals with CGI Scripts)
- Consult the vendor documentation to identify all directories used for upload.
- The tester manually checks that the CGI and upload directory are not different and verify it in the DUT.
- The tester will run some web scanning tools like Nikto, to get the detailed information and verify that the CGI and upload directory are different.

9) **Expected Results for Pass:**

- The configured upload directory is different from the CGI/Scripting directory.
- Additional evidence might be provided that shows that the web server has no write rights for the CGI/Scripting directory.

10) **Expected Format of Evidence:** A part of the configuration file / screenshot of the configuration showing that the web server is properly configured.

11) Test Execution:

➤ **Test Case Number:** 01

a) **Test Case Name:** TC_NO_CGI_OR_SCRIPTING_FOR_UPLOADS

b) **Test Case Description:** To test whether the upload directory is equal to the CGI/Scripting directory.

c) **Execution Steps:**

The tester compares the *<path of the CGI-BIN directory>* and *<path of the upload directory>* and confirms that they are not same, if same the test case will fail.

The tester shall open any terminal emulator to access the DUT shell and use the following command.

- Command used: *ls -la <path of the CGI-BIN directory>*

```
amf@localhost $ ls -la
total 20
drwxr-xr-x 1 www-data www-data 4096 Jun 19 12:07 .
drwxr-xr-x 1 www-data www-data 4096 Apr 26 21:58 ..
dr--r--r-- 2 www-data www-data 4096 Jun 26 10:15 cgi-enabled
-rw-rw-r-- 1 www-data www-data  51 May 26 11:37 index.html
```

- (Here we can see that the *<path of the CGI-BIN directory>* does not have write access)

The tester shall open any terminal emulator and use the following command.

- Command used: *ls -la <path of the CGI-BIN directory>*

```
amf@localhost $ ls -la
total 20
drwxr-xr-x 1 www-data www-data 4096 Jun 19 12:07 .
drwxr-xr-x 1 www-data www-data 4096 Apr 26 21:58 ..
drw-r--r-- 2 www-data www-data 4096 Jun 26 10:15 cgi-enabled
-rw-rw-r-- 1 www-data www-data  51 May 26 11:37 index.html
```

- (Here we can see that the *<path of the CGI-BIN directory>* have write access)

- (This must be not done)

d) **Test Observations:** The CGI and upload directory are different.

12) Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_CGI_OR_SCRIPTING_FOR_UPLOADS		

2.11.9 TSTP for No execution of system commands with SSI

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.9
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) <ITSAR Section No & Name> Section 11: Web Server
- 2) <Security Requirement No & Name > 2.11.9 No execution of system commands with SSI
- 3) <Requirement Description: > If Server Side Includes (SSI) is active, the execution of system commands shall be deactivated.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.7]

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

5) **DUT Configuration:**

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: `nmap -p- 172.18.0.8 -sC -sV` (IP address of DUT)

```

[ashwini@ashwini-newslab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_   Not valid before: 2023-05-17T08:39:01
|_   Not valid after: 2024-05-16T08:39:01
|_   tls-alpn:
|_     http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

e. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built:   2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in: ***"/etc/apache2/apache2.conf"***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- Command used: ***grep -I "IncludesNOEXEC" <path to configuration file>*** (To check SSI is enabled or not with no exec directive)


```
amf@localhost $ grep -I "IncludesNOEXEC" /etc/apache2/apache2.conf
Options +IncludesNOEXEC
```

(Above result states that SSI is enabled with No Exec Directive)

- Command used: **grep -C2 -I "IncludesNOEXEC" <path to configuration file>** (To check for which directory SSI is enabled)

```
amf@localhost $ grep -C 2 -I "IncludesNOEXEC" /etc/apache2/apache2.conf
<Directory "/var/www/html/ssi-enabled">
    Options +IncludesNOEXEC
    AddOutputFilter INCLUDES .html
    AddType text/html .html
```

- Same has to be checked in for Vhost files.

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6) Preconditions

- The tester has administrative privileges.
- If the web server is configured with SSI active, this test applies.
- A tester machine is available.
- Test environment with a Terminal.

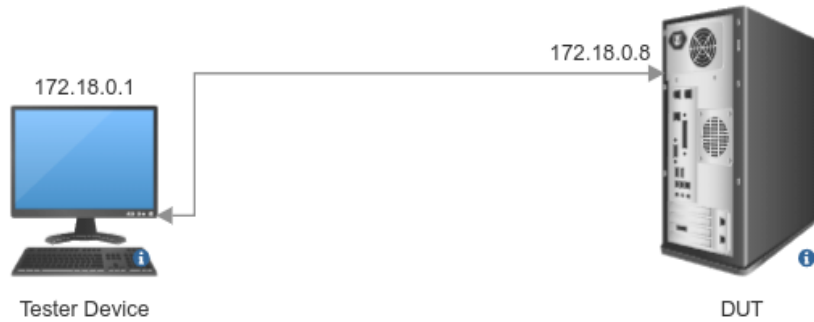
- 7) **Test Objective:** To test whether it is possible to use the exec directive and if so, whether it can be used for system commands.

8) Test Plan:

8.1 Number of Test Scenarios:

- 8.1.1 **Test Scenario for SSI:-** This test scenario is regarding SSI.

8.2 Test Setup Diagram:



8.3 **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 **Test Execution Steps**

- Power up the testbed
- The tester tries to access the Shell of Web Server.
- The tester checks whether execution of system commands is disabled in the web server configuration.
- The tester actually attempts to use the exec directive in an SSI file with and without system commands.
- Some web scanning tools like Nikto should be used to scan the webserver for the SSI exec.

9) **Expected Results for Pass:**

- The execution of system commands via SSIs exec directive is disabled in the web server configuration.
- It is impossible to execute system commands via SSIs exec directives.

10) **Expected Format of Evidence:** A part of the configuration file / screenshot of the configuration showing that the web server is properly configured. For example, a configuration file that shows that the IncludesNOEXEC (Apache HTTP Server®) or ssiExecDisable (Microsoft® IIS) is set.

- Web server log.

11) **Test Execution:**

➤ **Test Case Number:** 01

a) **Test Case Name:** TC_NO_EXECUTION_OF_SYSTEM_COMMANDS

b) **Test Case Description:** ----- CHANGE -----

c) **Execution Steps:**

The tester shall open any browser to access the DUT webserver using the following URL.

- URL: ***http://<ip/hostname of DUT>/<path of the SSI directory>***



[an error occurred while processing this directive]

- (Here we can see that we got the error when using the exec directive)
- We can see that the output is an error while using EXEC directive because we have used **Options IncludeNOEXEC**.
- With proper authentication the tester shall open any terminal emulator and use the following command to see the error log in the DUT.

Command used: **cat <path to the apache error log file>**

```
amf@localhost $ cat error.log
[Fri Jun 30 08:43:25.214042 2023] [cgid:error] [pid 387:tid 140349780350656] [client 172.18.0.1:36286] AH01271: exec used but not allowed in /var/www/html/ssi-enabled/index.html
```

- We can see that the error stating **exec used but not allowed in <path of the file>**
The tester shall open any browser to access the DUT webserver using the following URL.
- URL: **http://<ip/hostname of DUT>/<path of the SSI directory>**



This executed using is a shell command

- We can see that a shell command is being run using EXEC directive and it run successfully.
- (This should not be executed)

d) **Test Observations:** When Server Side Includes (SSI) is active, the execution of system commands is deactivated using Includes NOEXEC Options.

12) **Test Case Result:**

Securing Networks

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_EXECUTION_OF_SYSTEM_COMMANDS		

2.11.10 TSTP for Access rights for web server configuration

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.10
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) **<ITSAR Section No & Name>** Section 11: Web Server
- 2) **<Security Requirement No & Name >** 2.11.10 Access rights for web server configuration
- 3) **<Requirement Description: >**Access rights for SMF web server configuration files shall only be granted to the owner of the web server process or to a user with system privileges.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.8]

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: **cat /etc/os-release** (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: **nmap -p- 172.18.0.8 -sC -sV** (IP address of DUT)

```

[ashwini@ashwini-newslab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_   Not valid before: 2023-05-17T08:39:01
|_   Not valid after: 2024-05-16T08:39:01
|_   tls-alpn:
|_     http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

f. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built:   2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in: ***/etc/apache2/apache2.conf***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- Command used: ***apache2ctl -t -D DUMP_VHOSTS*** (To list what VirtualHost Configurations are running)
 - Using the comand we got the output and the config file
- (***/etc/apache2/sites-enabled/000-default.conf***)


```
amf@localhost $ apache2ctl -t -D DUMP_VHOSTS
VirtualHost configuration:
*:80                  172.18.0.8 (/etc/apache2/sites-enabled/000-default.conf:1)
```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6) Preconditions

- The tester has administrative privileges
- A tester machine is available.

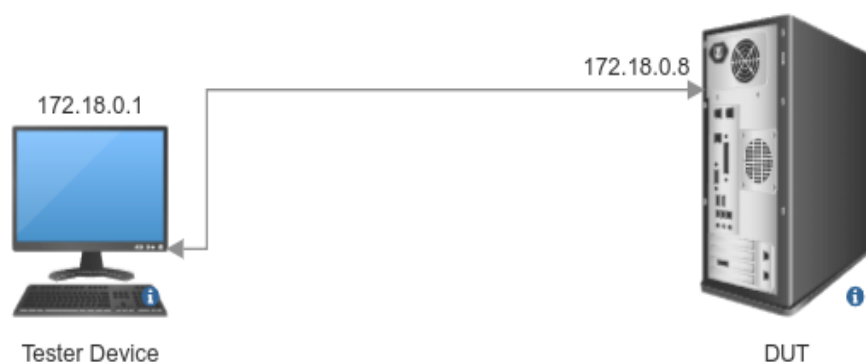
7) **Test Objective:** To verify that the access rights for Web server configuration files are correctly set.

8) Test Plan:

8.1 Number of Test Scenarios:

8.1.1 **Test Scenario for Apache:-** This test scenario is regarding Apache Web Server

8.2 Test Setup Diagram:



8.3 **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 Test Execution Steps

- Power up the testbed
- The tester tries to access the Web Server in the browser.
- Check the access rights settings for Web server system configuration files.
- Check that relevant system settings and configurations are correct to ensure fulfilment of the requirement.
- The tester also scans the Web Server using a suitable tool (e.g., Nikto) to verify that the relevant system settings and configurations are correct to ensure fulfilment of the requirement.

9) **Expected Results for Pass:** Access rights for system configuration files are adequately set.

10) **Expected Format of Evidence:** Log files and screen shots of test executions.

11) **Test Execution:**

➤ **Test Case Number:** 01

a) **Test Case Name:** TC_ACCESS_RIGHTS_WEB_SERVER_FILES

b) **Test Case Description:** To verify that the access rights for Web server configuration files are correctly set.

c) **Execution Steps:**

With the proper privilege in the DUT, we can see the permission on the configuration file.

- Command used: *ls -la <path to the configuration file>*

```
amf@localhost $ ls -la apache2.conf
-rw-r----- 1 www-data www-data 7567 Jun 28 05:03 apache2.conf
```

- Here we can see that

- Owner - RW
- Group - R
- Other - <None>

- Only granted to the owner of the web server process or to a user with system privileges.

With the proper privilege in the DUT, we can see the permission on the configuration file.

- Command used: *ls -la <path to the configuration file>*

```
amf@localhost $ ls -la apache2.conf
-rw-rw-r-- 1 www-data www-data 7567 Jun 28 05:03 apache2.conf
```

- Here we can see that

- Owner - RW
- Group - RW

- Other - R
- Read and Write permission should not be given to the **Other** and to the **Group** no write permission.

With the proper privilege in the DUT, we can see the permission on the configuration file.

- Command used: ***ls -la <path to the VHOST configuration file>***

```
amf@localhost $ ls -la 000-default.conf
-rw-r----- 1 ubuntu ubuntu 1402 Jun 26 11:10 000-default.conf
```

- Here we can see that

- Owner - RW
- Group - R
- Other - <None>

- Only granted to the owner of the web server process or to a user with system privileges. With the proper privilege in the DUT, we can see the permission on the configuration file.

- Command used: ***ls -la <path to the VHOST configuration file>***

```
amf@localhost $ ls -la 000-default.conf
-rw-rw-r-- 1 ubuntu ubuntu 1402 Jun 26 11:10 000-default.conf
```

- Here we can see that

- Owner - RW
- Group - RW
- Other - R

- Read and Write permission should not be given to the **Other** and to the **Group** no write permission.

The Above step for Virtual Host should be done for every Vhosts file.

- d) **Test Observations:** It should be ensured and verify that the relevant system settings and configurations are correct to ensure fulfilment of the requirement.

12) **Test** Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_ACCESS_RIGHTS_WEB_SERVER_FILES		

2.11.11 TSTP for Evaluation of No default content

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.11
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. <ITSAR Section No & Name> Section 11: Web Server

2. <Security Requirement No & Name > 2.11.11 No default content

3. <Requirement Description: >

Default content that is provided with the standard installation of the SMF web server shall be removed.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.9]

4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: **nmap -p- 172.18.0.8 -sC -sV** (IP address of DUT)

```

[ashwini@ashwini-news-lab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_   Not valid before: 2023-05-17T08:39:01
|_   Not valid after: 2024-05-16T08:39:01
|_   tls-alpn:
|_     http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

a. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built: 2023-03-08T16:32:34

```

To get the hash of configuration file if the file is a ASCII text file

Command used: ***sha256sum SMF_config.conf*** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf

```

To get the hash of OS if using docker

Command used: ***docker images --digests*** (To get hash/digest of config file)

```

amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f

```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6.Preconditions

- The tester has administrative privileges
- A tester machine is available.
- Test environment with a Web Browser.

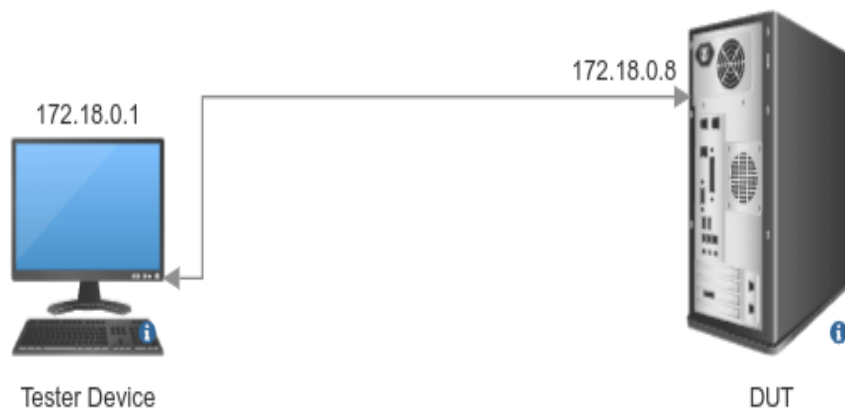
7. **Test Objective:** To verify that there is no default content on the web server, that is not needed for web server operation, since such default content can be useful for an attacker.

8. **Test Plan:**

8.1 **Number of Test Scenarios:**

8.1.1 **Test Scenario for Apache:-** This test scenario is regarding Apache Web Server

8.2 **Test Setup Diagram:**



8.3 **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 **Test Execution Steps**

- Power up the testbed
- The tester tries to access the Web Server in the browser.
- The tester should verify the Web-Server documentation to know the default contents which gets installed along with the Web-Server installation.
- The tester manually verifies that the no default content is available.
- The tester scans the Webserver using a suitable tool (e.g. Nikto) and verifies the tool report for availability of any default content.

9. **Expected Results for Pass:** No default content (examples, help files, documentation, aliases, un-needed directories or manuals) has been found to remain on any Web server component.

10. **Expected Format of Evidence:** Log files and screen shots of test executions.

11. Test Execution:

➤ Test Case Number: 01

a. Test Case Name: TC_NO_DEFAULT_CONTENT

b. **Test Case Description:** To verify that there is no default content on the web server, that is not needed for web server operation, since such default content can be useful for an attacker.

c. Execution Steps:

1. The tester shall open any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.

- URL: *http://<IP address of DUT>*

The index webpage must show the DUT Content not the server default content.



DUT - WEB SERVER

2. The tester opens any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.

- URL: *http://<IP address of DUT>*

- (NO Default content should be visible)



NOTE: Tester should check the root directory and all accessible directories by the web server and ensure there are no examples, help files, documentation, or aliases. Here there only the root directory is present, so we are only checking that. Test is not complete without a thorough check of all directories present by default.

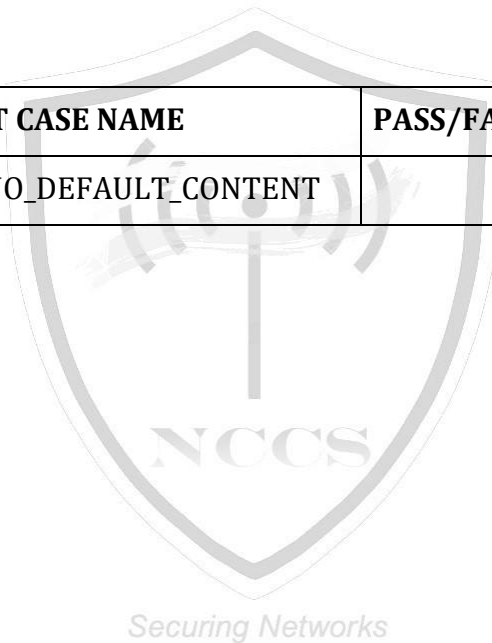
For example, the default director is /var/www/html in httpd. As we can see only index.html is present here

```
naan30@naa30-pc:/var/www/html$ ls
index.html
naan30@naa30-pc:/var/www/html$
```

- d. **Test Observations:** It should be ensured that Web Server Component should not serve default content.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_DEFAULT_CONTENT		



2.11.12 TSTP for Evaluation of No directory listings

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.12
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 11: Web Server
2. **<Security Requirement No & Name >** 2.11.12 No directory listings
3. **<Requirement Description: >**Directory listings (indexing) / "Directory browsing" shall be deactivated.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.10]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: `cat /etc/os-release` (To get OS information)

```

amf@localhost $ cat /etc/os-release

```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: `nmap -p- 172.18.0.8 -sC -sV` (IP address of DUT)

```

[ashwini@ashwini-news-lab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_ Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_ Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_ Not valid before: 2023-05-17T08:39:01
|_ Not valid after: 2024-05-16T08:39:01
|_ tls-alpn:
|_ http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

a. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built: 2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in: ***"/etc/apache2/apache2.conf"***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- Command used: ***grep -I "Indexes" /etc/apache2/apache2.conf*** (To check if the Directory indexing is disabled or not)
 - ***Options -Indexes*** (Directory listing is disabled)
 - ***Options +Indexes*** (Directory listing is enabled)


```
amf@localhost $ grep -I "Indexes" /etc/apache2/apache2.conf
Options -Indexes
```

(Above result states that directory listing is disabled)

```
amf@localhost $ grep -I "Indexes" /etc/apache2/apache2.conf
Options Indexes FollowSymLinks
```

(Above result states that directory listing is enabled)

- Command used: **apache2ctl -t -D DUMP_VHOSTS** (To list what VirtualHost Configurations are running)
 - Using the command we got the output and the config file
- (/etc/apache2/sites-enabled/000-default.conf)
 - Above step should be applied here also to check if directory listing is enabled or not in the VirtualHost Configuration file.

```
amf@localhost $ apache2ctl -t -D DUMP_VHOSTS
VirtualHost configuration:
*:80                  172.18.0.8 (/etc/apache2/sites-enabled/000-default.conf:1)
```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6. Preconditions

- The tester has administrative privileges
- A tester machine is available.
- Test environment with a Web Browser.

7. **Test Objective:** To verify that Directory listings / Directory browsing has been deactivated in all Web server components.

8. Test Plan:

8.1 Number of Test Scenarios:

8.1.1 Test Scenario for Apache:-This test scenario is regarding Apache Web Server

8.2 **Test Setup Diagram:**



8.3 **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 **Test Execution Steps**

- Power up the testbed
- The tester tries to access the Web Server in the browser.
- The tester verifies that the Directory Listing/Indexing is disabled.
- The tester also scans the Web Server using a suitable tool (e.g. Nikto) to verify that the Directory listing is not present.

9. **Expected Results for Pass:** Evidence that Directory listing / Directory browsing has been deactivated in all Web server components.

10. **Expected Format of Evidence:** Log files and screen shots of test executions.

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC_NO_DIRECTORY_LISTINGS

b. **Test Case Description:** To verify that Directory listings / Directory browsing has been deactivated in all Web server components.

c. **Execution Steps:**

1. The tester shall open any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.
 - URL: ***http://<IP address of DUT>***
 - **The Webpage must show Forbidden with a HTTP status code 403.**



Forbidden

You don't have permission to access this resource.

Apache/2.4.55 (Ubuntu) Server at 172.18.0.8 Port 80

2. With the proper privilege in the DUT while accessing the Log file we can see the following log info.

- Command used: **cat /var/log/apache2/error.log**






```
amf@localhost $ cat /var/log/apache2/error.log
[Thu May 25 06:16:02.813553 2023] [autoindex:error] [pid 12:tid 139864527857344] [client 172.18.0.1:52082] AH01276: Can not serve directory /var/www/html/: No matching DirectoryIndex (index.html,index.cgi,index.pl,index.php,index.xhtml,index.htm) found, and server-generated directory index forbidden by Options directive
```

server-generated directory index forbidden by Options directive

3. The tester opens any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.
 - URL: **http://<IP address of DUT>**
 - (NO Directory listing/indexing should not be visible)



Index of /

	<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
	conf/	2023-05-25 05:51	-	
	configuration.conf	2023-05-25 05:50	15	
	image.png	2023-05-25 05:49	15	
	main.html	2023-04-26 21:58	10K	
	main.py	2023-05-25 05:49	15	

Apache/2.4.55 (Ubuntu) Server at 172.18.0.8 Port 80

4. The tester repeats the above steps to verify that directory listing is disabled for all directories and subdirectories accessible by the web server.

d. Test Observations:

It should be ensured that any path in the Web Server or any Component of the Web Server should not allow Directory Listing/Indexing.

12. Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_DIRECTORY_LISTINGS		



2.11.13 TSTP for Evaluation of Web server information in HTTPS headers

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.13
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

<DUT Details: > Ex: Router

<DUT Software Version:>

<Digest Hash of OS>

<Digest Hash of Configuration>

<Applicable ITSAR: >

<ITSAR Version No:>

<OEM Supplied Document list: >

- 1) <ITSAR Section No & Name> Section 11: Web Server
- 2) <Security Requirement No & Name > 2.11.13 Web server information in HTTPS headers
- 3) <Requirement Description: >The HTTPS header shall not include information on the version of the SMF web server and the modules/add-ons used.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.11]

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```

amf@localhost $ ./amf.out --version
IIT_amf 6.9

```

Command used: **cat /etc/os-release** (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1  NAME="Ubuntu"
2  VERSION="20.04.5 LTS (Focal Fossa)"
3  ID=ubuntu
4  ID_LIKE=debian
5  PRETTY_NAME="Ubuntu 20.04.5 LTS"
6  VERSION_ID="20.04"
7  HOME_URL="https://www.ubuntu.com/"
8  SUPPORT_URL="https://help.ubuntu.com/"
9  BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10  PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11  VERSION_CODENAME=focal
12  UBUNTU_CODENAME=focal

```

5) DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: **nmap -p- 172.18.0.8 -sC -sV** (IP address of DUT)


```

[ashwini@ashwini-newslab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_ Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_ Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_ Not valid before: 2023-05-17T08:39:01
|_ Not valid after: 2024-05-16T08:39:01
|_ tls-alpn:
|_ http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

g. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built:   2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in: ***"/etc/apache2/apache2.conf"***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- Command used: ***grep -I "Server Tokens" /etc/apache2/apache2.conf*** (To check Apache will send back in response headers)
 - Server Tokens decides what Apache will send back in response headers. It takes the following values

- Server Tokens Full (or not specified)
 - Response to clients: Server: Apache/2.4.2 (Unix) PHP/4.2.2
- Server Tokens Prod[uctOnly]
 - Response to clients: Server: Apache
- Server Tokens Major
 - Response to clients: Server: Apache/2
- Server Tokens Minor
 - Response to clients: Server: Apache/2.4
- Server Tokens Min[imal]
 - Response to clients: Server: Apache/2.4.2
- Server Tokens OS
 - Response to clients: Server: Apache/2.4.2 (Unix)

```
amf@localhost $ grep -I "ServerTokens" /etc/apache2/apache2.conf
ServerTokens Prod
```

(Above result states that ServerTokens should be set to Prod not anything else)

- Command used: **grep -I "SecServerSignature" /etc/apache2/apache2.conf** (To check Apache will send back null in response headers)

```
amf@localhost $ grep -I "SecServerSignature" /etc/apache2/apache2.conf
SecServerSignature " "
```

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6) Preconditions

- The tester has administrative privileges
- A tester machine is available.
- Test environment with a Terminal.

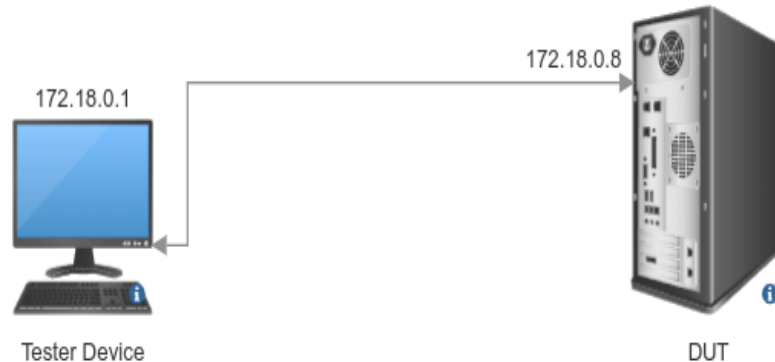
7) **Test Objective:** To verify that HTTP headers do not include information on the version of the web server and the modules/add-ons used.

8) **Test Plan:**

8.1 **Number of Test Scenarios:**

8.1.1 **Test Scenario for Apache:-** This test scenario is regarding Apache Web Server

8.2 **Test Setup Diagram:**



8.3 **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4 **Test Execution Steps**

- Power up the testbed
- The tester tries to access the Header of Web Server.
- The tester manually verifies that HTTP headers do not include information on the version of the web server.
- The tester will run some web scanning tools like Nikto, to get the detailed information and verify that the HTTP headers do not include information on the version of the web server.

9) **Expected Results for Pass:** Evidence that HTTP headers do not include information on the version of the web server and the modules/addons used.

10) **Expected Format of Evidence:** Log files and screen shots of test executions.

11) **Test Execution:**

➤ **Test Case Number:** 01

a) **Test Case Name:** TC_NO_WEB_SERVER_HEADER_INFORMATION

b) **Test Case Description:** To verify that HTTP headers do not include information on the version of the web server and the modules/add-ons used.

c) **Execution Steps:**

The tester shall open any terminal emulator and use the following command.

- Command used: ***curl -I http://< IP address of DUT >***

```
[ashwini@ashwini-news-lab]~$ curl -I http://172.18.0.8
HTTP/1.1 200 OK
Date: Mon, 26 Jun 2023 11:10:13 GMT
Server:
Last-Modified: Fri, 26 May 2023 11:37:56 GMT
ETag: "33-5fc972b14ac6b"
Accept-Ranges: bytes
Content-Length: 51
Content-Type: text/html
```

- We can see that the HTTP headers do not include information on the version of the web server and the modules/addons used.

The tester shall open any terminal emulator and use the following command.

- Command used: ***curl -I http://< IP address of DUT >***

```
[ashwini@ashwini-news-lab]~$ curl -I http://172.18.0.8
HTTP/1.1 200 OK
Date: Fri, 26 May 2023 11:45:35 GMT
Server: Apache/2.4.55 (Ubuntu)
Last-Modified: Fri, 26 May 2023 11:37:56 GMT
ETag: "33-5fc972b14ac6b"
Accept-Ranges: bytes
Content-Length: 51
Content-Type: text/html
```

- We can see that the HTTP headers do include information on the version of the web server and the modules/addons used.

- (This should not be present)

d) **Test Observations:** It should be ensured that the HTTP headers do not include information on the version of the web server and the modules/addons used.

12) Test Case Result:

SL. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_WEB_SERVER_HEADE R_INFORMATION		

2.11.14 TSTP for Evaluation of Web server information in error pages

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.14
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. <ITSAR Section No & Name> Section 11: Web Server
2. <Security Requirement No & Name > 2.11.13 Web server information in HTTPS headers
3. <Requirement Description: >User-defined error pages and Error messages shall not include version information and other internal information about the SMF web server and the modules/add-ons used.

Default error pages of the SMF web server shall be replaced by error pages defined by the OEM.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.12]

4. DUT Confirmation Details:

Securing Networks

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No

- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)
- **Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)**

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

- Command used : **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: **nmap -p- 172.18.0.8 -sC -sV** (IP address of DUT)


```

[ashwini@ashwini-news-lab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_   http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_   http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_   http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_   http-title: Site doesn't have a title (text/html).
|_   ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_   Not valid before: 2023-05-17T08:39:01
|_   Not valid after: 2024-05-16T08:39:01
|_   tls-alpn:
|_     http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

a. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built:   2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in : ***"/etc/apache2/apache2.conf"***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- Command used: ***grep -I "ServerSignatures" /etc/apache2/apache2.conf*** (To check Apache will suppresses the footer line which contain Server information in error pages)

- The ServerSignature directive allows the configuration of a trailing footer line under server-generated documents (error messages, mod_proxy ftp directory listings, mod_info output, ...).
- The **Off setting**, which is the default, suppresses the footer line. The **On setting** simply adds a line with the server version number and ServerName of the serving virtual host, and the **Email setting** additionally creates a "mailto:" reference to the ServerAdmin of the referenced document.

```
amf@localhost $ grep -I "ServerSignature" /etc/apache2/apache2.conf
ServerSignature Off
```

(Above result states that ServerSignatures should be set to off not anything else)

- To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf
```

- To get the hash of OS if using docker
- Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6. Preconditions

- The tester has administrative privileges
- A tester machine is available.
- Test environment with a Terminal.

7. **Test Objective:** To verify that error pages and error messages do not include information about the web server.

8. Test Plan:

8.1. Number of Test Scenarios:

8.1.1. Test Scenario for Apache:- This test scenario is regarding Apache Web Server

8.2. Test Setup Diagram:



8.3. **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4. Test Execution Steps

- Power up the testbed
 - The tester tries to access the Header of Web Server.
 - The tester verifies that generated error pages and error messages do not include information about the web server.
 - Tester checks for any other Error case like 5XX, 4XX errors.
 - Tester checks for one more error scenario randomly.
 - In case of tester does not have access to the web server configuration files on the DUT, then test is to be performed for all the Http error scenarios (5XX, 4XX).
9. **Expected Results for Pass:** Evidence that generated error pages and error messages do not include information about the web server

10. **Expected Format of Evidence:** Log files and screen shots of test executions.

11. Test Execution:

➤ **Test Case Number:** 01

a. **Test Case Name:** TC_NO_WEB_SERVER_ERROR_PAGES_INFORMATION

b. **Test Case Description:** To verify that error pages and error messages do not include information about the web server.

c. **Execution Steps:**

1. The tester shall open any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.
- URL: http://<IP address of DUT>/<random text>
 - (Random text is used so that we can get the Http 404 error)



Not Found

The requested URL was not found on this server.

- We can see that the error page do not include information on the version of the web server and the modules/addons used.

2. The tester opens any Web Browser (Mozilla Firefox, Google Chrome) and go to the web address/IP of the DUT.

- URL: http://<IP address of DUT>/ <random text>
- (Random text is used so that we can get the Http 404 error)

(NO Directory listing/indexing should not be visible)



Not Found

The requested URL was not found on this server.

Apache/2.4.55 (Ubuntu) Server at 172.18.0.8 Port 80

- We can see that the error page do include information on the version of the web server and the modules/addons used.
- (This should not be present)
- Tester checks for any other Error case like 5XX, 4XX errors.
- Tester checks for one more error scenario randomly.
- Incase of tester does not have access to the web server configuration files on the DUT, then test is to be performed for all the Http error scenarios (5XX, 4XX).

d. **Test Observations:**

- It should be ensured that that the error page do not include information on the version of the web server and the modules/addons used.

12. **Test Case Result:**

Sl. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_WEB_SERVER_ERROR_PAGES_INFORMATION		



2.11.15 TSTP for Minimized file type mappings

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.15
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
 - <DUT Software Version:>
 - <Digest Hash of OS>
 - <Digest Hash of Configuration>
 - <Applicable ITSAR: >
 - <ITSAR Version No:>
 - <OEM Supplied Document list: >
- 1) <ITSAR Section No & Name> Section 11: Web Server
 - 2) <Security Requirement No & Name > 2.11.15 Minimized file type mappings
 - 3) <Requirement Description: >File type or script-mappings that are not required for SMF operation shall be deleted.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.13]

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
 - Use command to get Application No/Version
 - Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)
- **Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)**

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

- Command used : **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used : **nmap -p- 172.18.0.8 -sC -sV** (IP address of DUT)

```

[ashwini@ashwini-newslab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_ Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_ Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_ Not valid before: 2023-05-17T08:39:01
|_ Not valid after: 2024-05-16T08:39:01
|_ tls-alpn:
|_ http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

h. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built: 2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in : ***"/etc/apache2/apache2.conf"***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- To get the hash of configuration file if the file is a ASCII text file
Command used: ***sha256sum SMF_config.conf*** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

- To get the hash of OS if using docker

Command used: docker images --digests (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6) Preconditions

- The tester has administrative privileges
- The web server is configured according to the manual
- The vendor must provide the list of necessary file types.
- A tester machine is available.

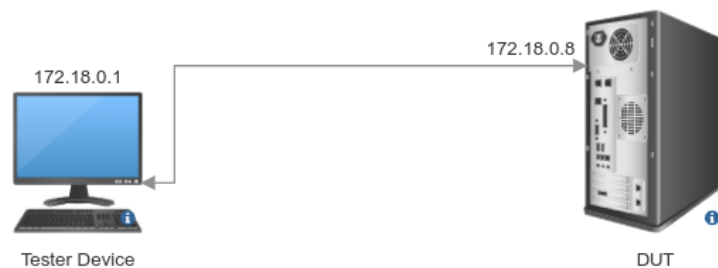
7) **Test Objective:** To verify that file type- or script-mappings that are not required have been deleted.

8) Test Plan:

8.1. Number of Test Scenarios:

8.1.1. Test Scenario for Apache:- This test scenario is regarding Apache Web Server

8.2. Test Setup Diagram:



8.3. **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4. Test Execution Steps

- Power up the testbed

- The tester manually verifies that the file type- or script-mappings that are not required have been deleted.

9) **Expected Results for Pass:** Evidence that all file type- or script-mappings, that are not required, have been deleted.

10) **Expected Format of Evidence:** Log files and screen shots of test executions.

11) **Test Execution:**

➤ **Test Case Number:** 01

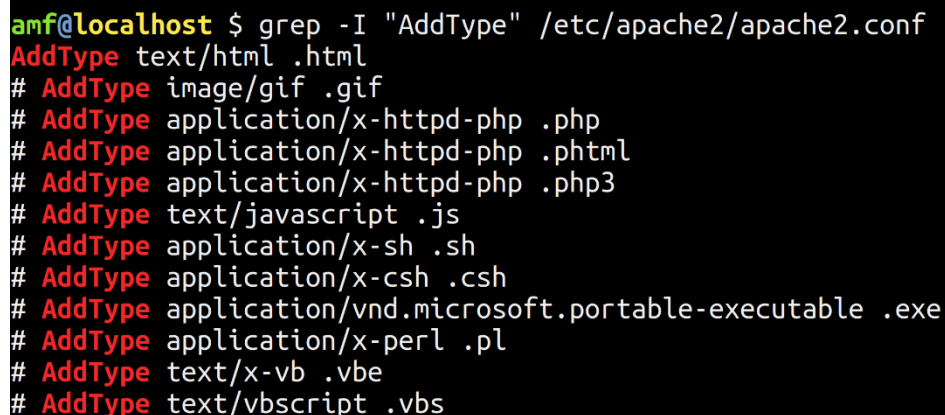
a) **Test Case Name:** TC_NO_WEB_SERVER_FILE_TYPE MAPPINGS

b) **Test Case Description:** To verify that file type- or script-mappings that are not required have been deleted.

c) **Execution Steps:**

With the proper privilege in the DUT access the shell to see the file types that are commented out.

Command used: ***grep -I "AddType" <path of configuration file>***



```
amf@localhost $ grep -I "AddType" /etc/apache2/apache2.conf
AddType text/html .html
# AddType image/gif .gif
# AddType application/x-httpd-php .php
# AddType application/x-httpd-php .phtml
# AddType application/x-httpd-php .php3
# AddType text/javascript .js
# AddType application/x-sh .sh
# AddType application/x-csh .csh
# AddType application/vnd.microsoft.portable-executable .exe
# AddType application/x-perl .pl
# AddType text/x-vb .vbe
# AddType text/vbscript .vbs
```

- Here we can see that the File type that has been added in the server using AddType Directive.
- The File Types have been commented out using “#”
- The tester will compare the list by the Vendor and the output of the command any filetype left the test case will fail.

With the proper privilege in the DUT access the shell to see the which File types are been removed.

- Command used: ***grep -I "RemoveType" <path of configuration file>***

```
amf@localhost $ grep -I "RemoveType" /etc/apache2/apache2.conf
RemoveType .gif
RemoveType .php
RemoveType .phtml
RemoveType .php3
RemoveType .js
RemoveType .sh
RemoveType .csh
RemoveType .exe
RemoveType .pl
RemoveType .vbe
RemoveType .vbs
```

(Here we can see that the unnecessary file types are removed using ***RemoveType*** Directive)

d) **Test Observations:-** Hence, we verified that the unnecessary file types are removed in the web server.

12) **Test Case Result:**

Sl. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_NO_WEB_SERVER_FILE_TYPE MAPPINGS		

NCCS
Securing Networks

2.11.16 TSTP for Evaluation of Restricted file access

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.16
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 11: Web Server
2. **<Security Requirement No & Name >** 2.11.16 Restricted file access
3. **<Requirement Description: >** Restrictive access rights shall be assigned to all files which are directly or indirectly reside in the SMF web server's document directory. In particular, the SMF web server shall not be able to access files which are not meant to be delivered.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.14]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
 - Command used: **ifconfig -a** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: `./SMF.out` (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)
- Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

- Command used: `cat /etc/os-release` (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: `nmap -p- 172.18.0.8 -sC -sV` (IP address of DUT)

```

[ashwini@ashwini-newslab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp   open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_   Not valid before: 2023-05-17T08:39:01
|_   Not valid after: 2024-05-16T08:39:01
|_   tls-alpn:
|_     http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

a. For Apache httpd:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built:   2023-03-08T16:32:34

```

- Command used: ***find / -name apache2.service*** (To get the systemd service file of apache2)

```

amf@localhost $ find / -name apache2.service 2> /dev/null
/etc/systemd/system/multi-user.target.wants/apache2.service

```

- Command used: ***cat <path of the Apache service file>*** (To know if it is running in a jailed chrooted environment or not)

```
amf@localhost $ cat /etc/systemd/system/multi-user.target.wants/apache2.service
```

```
File: /etc/systemd/system/multi-user.target.wants/apache2.service

1 [Unit]
2 Description=The Apache HTTP Server
3 After=network.target remote-fs.target nss-lookup.target
4 Documentation=https://httpd.apache.org/docs/2.4/
5
6 [Service]
7 Type=forking
8 Environment=APACHE_STARTED_BY_SYSTEMD=true
9 ExecStart=/sbin/chroot /chroot/httpd /usr/sbin/apachectl start
10 ExecStop=/sbin/chroot /chroot/httpd /usr/sbin/apachectl graceful-stop
11 ExecReload=/sbin/chroot /chroot/httpd /usr/sbin/apachectl graceful
12 KillMode=process
13 PrivateTmp=true
14 Restart=on-abort
15
16 [Install]
17 WantedBy=multi-user.target
```

(We can see every command starts with `/sbin/chroot/` this shows that the Apache server is running in chrooted environment.)

- To get the hash of configuration file if the file is a ASCII text file

Command used: ***sha256sum SMF_config.conf*** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30  AMF_Config.conf
```

- To get the hash of OS if using docker
- Command used: `docker images --digests` (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6. Preconditions

- The tester has administrative privileges
- The web server is configured according to the manual
- A tester machine is available.
- Test environment with a Web Browser.

7. **Test Objective:** To test whether the restrictive access rights are assigned to all files which are directly or indirectly in the web server's document directory and to verify whether path traversal is made improbable.

8. Test Plan:

8.1. Number of Test Scenarios:

8.1.1. Test Scenario for Apache

- This test scenario is regarding Apache Web Server

8.2. **Test Setup Diagram:**



8.3. **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4. **Test Execution Steps**

- Power up the testbed
- The tester verifies that access rights on the servable content (meaning directories and files) is set to the following:
 - o The files are owned by the user that runs the web server;
 - o The files are not writable to others, except the web server's account;
- The tester verifies that the user running the web server is an unprivileged account;(Refer the TSTP for clause 2.11.4 - No system privileges)
- For Operating Systems that have chrooted environments, the tester verifies that the web server runs inside a jail or chrooted environment.

9. **Expected Results for Pass:**

Securing Networks

- Name of user running the web server with the privileges of the account;
- Access rights of files and directories that the web server serves;
- Configuration that shows that the web server is in a chrooted environment

10. **Expected Format of Evidence:** Log files and screen shots of test executions.

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC_RESTRICTED_FILE_ACCESS

b. **Test Case Description:** To test whether the restrictive access rights are assigned to all files which are directly or indirectly in the web server's document directory and to verify whether path traversal is made improbable.

c. **Execution Steps:**

1. With the proper privilege in the DUT access the shell to see the owner and group of the files and directories.
- Command used: ***ls -la <path of the directory where the files are hosted>***

```
amf@localhost $ ls -la
total 20
drwxr-xr-x 1 www-data www-data 4096 Jun 19 12:07 .
drwxr-xr-x 1 www-data www-data 4096 Apr 26 21:58 ..
drwxr-xr-x 2 www-data www-data 4096 Jun 19 12:08 cgi-enabled
-rw-rw-r-- 1 www-data www-data 51 May 26 11:37 index.html
```

(Here we can see that both the owner and group of the files are of the www-data that is a special User and Group under Apache)

2. With the proper privilege in the DUT access the shell to see the owner and group of the files and directories.

- Command used: ***id***

```
amf@localhost $ id
uid=1001(amf) gid=1001(amf) groups=1001(amf)
```

(Here we can see the USERID is **1001** and the USERNAME is **SMF**)

- Command used: ***echo " " >> <path of the directory where the files are hosted>/<any available file>***

```
amf@localhost $ echo " " >> /var/www/html/index.html
bash: /var/www/html/index.html: Permission denied
```

(Here we can see that the permission denied cause the user is not the same user as the web server user.)

3. With the proper privilege in the DUT access the shell to see the owner and group of the files and directories.

- Command used: ***ps aux | grep apache2***

```
amf@localhost $ ps aux | grep apache2
root      181560  0.0  0.0  6836  5560 ?        Ss   13:50   0:00 /sbin/chroot /chroot/httpd usr/sbin/apache2 -DFOREGROUND -k start
www-data  181561  0.0  0.0  6496  2964 ?        Ss   13:50   0:00 /sbin/chroot /chroot/httpd usr/sbin/apache2 -DFOREGROUND -k start
www-data  181562  0.0  0.0  1997952 4108 ?       SL   13:50   0:00 /sbin/chroot /chroot/httpd usr/sbin/apache2 -DFOREGROUND -k start
www-data  181563  0.0  0.0  1997952 4132 ?       SL   13:50   0:00 /sbin/chroot /chroot/httpd usr/sbin/apache2 -DFOREGROUND -k start
root      184505  0.0  0.0   6768  4740 ?        Ss   14:03   0:00 /sbin/chroot /chroot/httpd usr/sbin/apache2 -k start
www-data  184506  0.0  0.0  1998248 5468 ?       SL   14:03   0:00 /sbin/chroot /chroot/httpd usr/sbin/apache2 -k start
www-data  184507  0.0  0.0  1998176 4520 ?       SL   14:03   0:00 /sbin/chroot /chroot/httpd usr/sbin/apache2 -k start
amf       188296  0.0  0.0  17864  2528 pts/4    S+   14:20   0:00 grep --colour=auto apache2
```

(We can see every process starts with */sbin/chroot/* this shows that the Apache server is running in chrooted environment.)

(Anything other than that the testcase will fail)

4. Additionally, Tester can set DocumentRoot (root folder of web server as a new folder For Example- *var/www/secure* with its own html file - Tester will create this directory and *index.html* inside it)

1. Then tester will ensure access rights are such that from port 80 externally only *index.html* of *var/www/secure* is accessible. Nor is *var/www/html* accessible not is any other file in *var/www/secure* accessible. The settings are change in *vi /etc/apache2/sites-available/000-default.conf*<VirtualHost *:80>

*# The ServerName directive sets the request scheme, hostname and port that
the server uses to identify itself. This is used when creating
redirection URLs. In the context of virtual hosts, the ServerName
specifies what hostname must appear in the request's Host: header to
match this virtual host. For the default virtual host (this file) this
value is not decisive as it is used as a last resort host regardless.
However, you must set it for any further virtual host explicitly.
#ServerName www.example.com*

ServerAdmin webmaster@localhost

DocumentRoot /var/www/secure

Adding another directoryAlias /secure /var/www/html<Directory /var/www/html>

Options Indexes FollowSymLinks AllowOverride All Require all granted</Directory>

Available loglevels: trace8, ..., trace1, debug, info, notice, warn,

error, crit, alert, emerg.

It is also possible to configure the loglevel for particular

modules, e.g.

#LogLevel info ssl:warn

ErrorLog \${APACHE_LOG_DIR}/error.log

CustomLog \${APACHE_LOG_DIR}/access.log combined

For most configuration files from conf-available/, which are

enabled or disabled at a global level, it is possible to

include a line for only one particular virtual host. For example the

following line enables the CGI configuration for this host only

after it has been globally disabled with "a2disconf".

#Include conf-available/serve-cgi-bin.conf

</VirtualHost>


```
<Directory />  
  Order Deny,Allow  
  Deny from all  
  Options None  
  AllowOverride None  
</Directory>
```

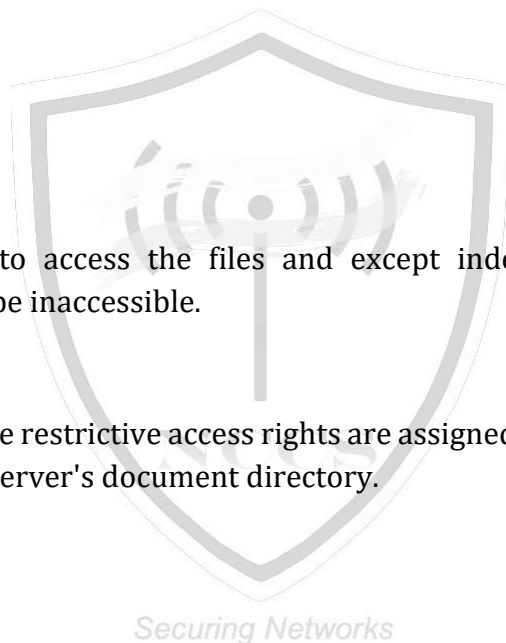
```
<Directory /var/www/secure>  
  # Deny first, then allow  
  Order deny,allow  
  # Deny everyone from everything  
  Deny from all
```

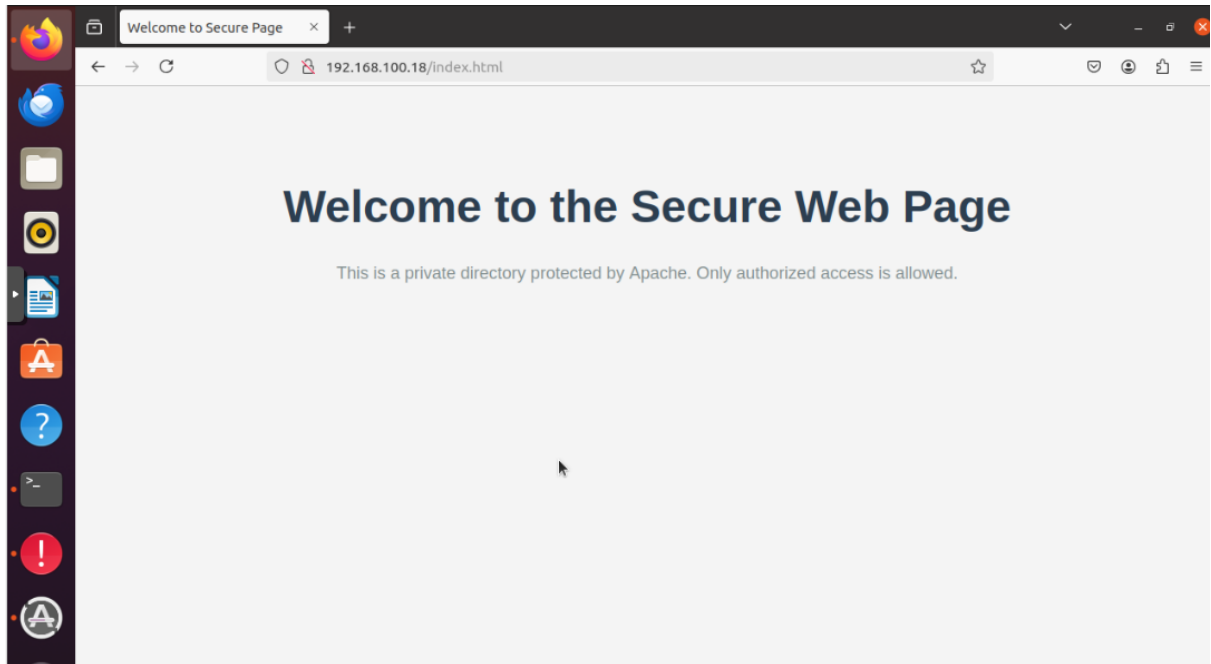
```
<FilesMatch index\.html>  
  # but allow index.html  
  Allow from all  
</FilesMatch>  
</Directory>
```

2. Then tester tries to access the files and except index.html of var/www/secure everything should be inaccessible.

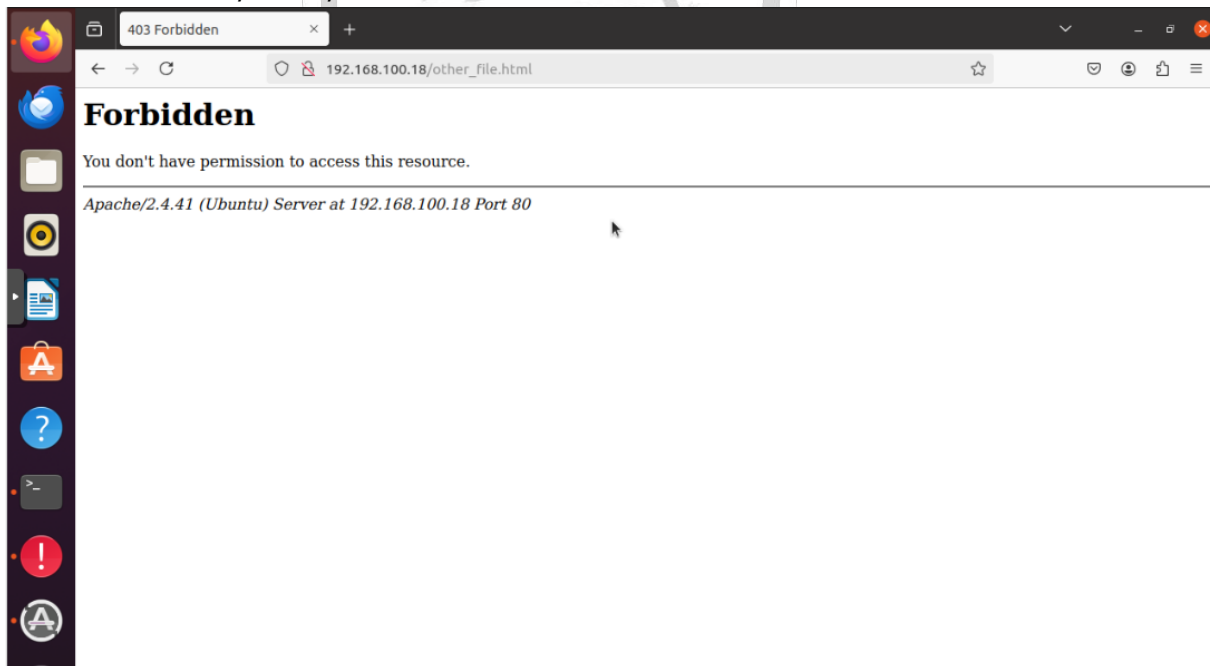
d. **Test Observations:**

- Hence we verified that the restrictive access rights are assigned to all files which are directly or indirectly in the web server's document directory.

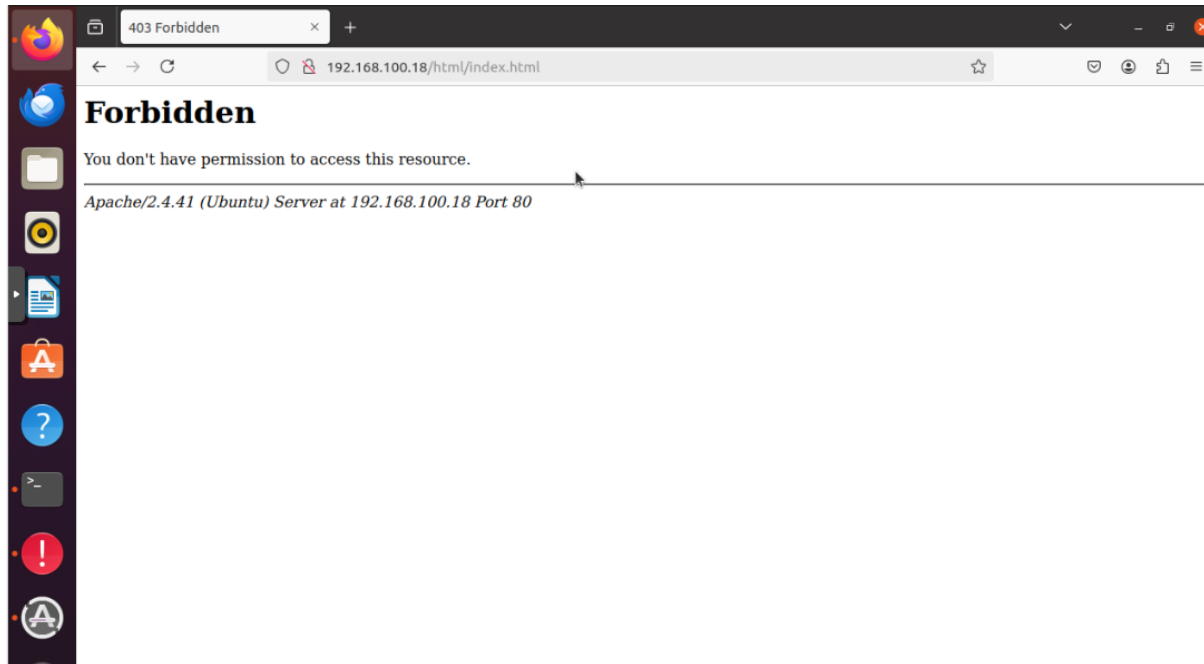




- index.html of var/www/secure is accessible



- other_file in var/www/secure is inaccessible



index.html of var/www/html also not accessible

12. **Test Case Result:**

Sl. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_RESTRICTED_FILE_ACCESS		

Securing Networks

2.11.17 TSTP for Execute rights exclusive for CGI/Scripting directory

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.17
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
 - <DUT Software Version:>
 - <Digest Hash of OS>
 - <Digest Hash of Configuration>
 - <Applicable ITSAR: >
 - <ITSAR Version No:>
 - <OEM Supplied Document list: >
1. <ITSAR Section No & Name> Section 11: Web Server
 2. <Security Requirement No & Name > 2.11.17 Execute rights exclusive for CGI/Scripting directory
 3. <Requirement Description: >If CGI or other scripting technology is used, only the CGI/Scripting directory is configured with execute rights. Other directories used or meant for web content do not have execute rights.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.3.4.15]

4. DUT Confirmation Details:

Securing Networks

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
 - Use command to get Application No/Version
 - Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)
- **Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)**

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

- Command used : **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

- Check OEM documentation to find all which Web Server and version is running in the DUT.
- Run NMAP or any port scanning tool to get running services and verify the version and type of the Web Server:

Command used: **nmap -p- 172.18.0.8 -sC -sV** (IP address of DUT)

```

[ashwini@ashwini-news-lab]~$ nmap -p- 172.18.0.8 -sC -sV
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-17 15:41 IST
Nmap scan report for 172.18.0.8
Host is up (0.00012s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
443/tcp    open  ssl/http  Apache httpd 2.4.57 ((Unix) OpenSSL/1.1.1n)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.4.57 (Unix) OpenSSL/1.1.1n
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=172.18.0.8/organizationName=Example Inc/stateOrProvinceName=Example/countryName=US
|_   Not valid before: 2023-05-17T08:39:01
|_   Not valid after: 2024-05-16T08:39:01
|_   tls-alpn:
|_     http/1.1

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.23 seconds

```

Example list of Web Server may be present as per (NMAP and OEM Documentation)

- Apache httpd
- Nginx
- Lighttpd
- Microsoft IIS and more

Note: We have made this TSTP for Apache httpd Web Server the tester must verify for the respective Web Server as mentioned in the OEM documentation.

a. For Apache httpd/CGI:

- Command used: ***apachectl -v*** (To get version information)

```

amf@localhost $ apachectl -v
Server version: Apache/2.4.55 (Ubuntu)
Server built:   2023-03-08T16:32:34

```

- Command used:
 - ***apachectl -V | grep HTTPD_ROOT*** (To get the Root directory of apache2)
 - ***apachectl -V | grep SERVER_CONFIG_FILE*** (To get the config file name)
 - Using that we found the Apache config file is in: ***" /etc/apache2/apache2.conf"***

```

amf@localhost $ apachectl -V | grep HTTPD_ROOT
-D HTTPD_ROOT="/etc/apache2"

amf@localhost $ apachectl -V | grep SERVER_CONFIG_FILE
-D SERVER_CONFIG_FILE="apache2.conf"

```

- Command used: ***grep -I "ExecCGI" /etc/apache2/apache2.conf*** (To check CGI is enabled or not)


```
amf@localhost $ grep -I "ExecCGI" /etc/apache2/apache2.conf
Options +ExecCGI
```

(Above result states that CGI is enabled)

- Command used: **grep -C2 -I "ExecCGI" /etc/apache2/apache2.conf** (To check for which directory CGI is enabled)

```
amf@localhost $ grep -C2 -I "ExecCGI" /etc/apache2/apache2.conf
<Directory "/var/www/html/cgi-enabled">
  Options +ExecCGI
  AddHandler cgi-script .cgi .pl .py .rb
</Directory>
```

- Same has to be checked in for Vhost files

- To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30  AMF_Config.conf
```

- To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Note: The tester must consider the respective steps for procuring above mentioned information if some other Web Server apart from Apache httpd is used by DUT.

6. Preconditions

- The tester has administrative privileges
- A tester machine is available.
- Test environment with a Terminal.
- Test Environment with a Browser
- If the web server is configured with CGI/Scripting on, this test applies

7. **Test Objective:** To test whether the web server only has execute permissions on the CGI/Scripting directory.

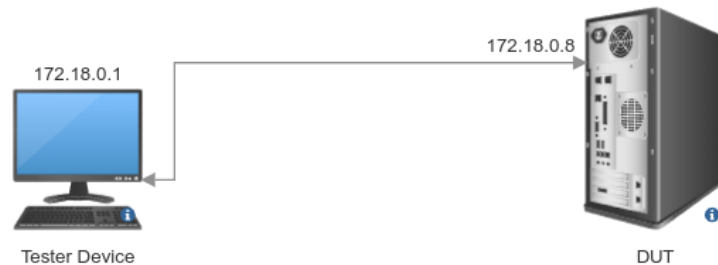
8. Test Plan:

8.1. Number of Test Scenarios:

8.1.1. Test Scenario for Apache

- This test scenario is regarding Apache Web Server

8.2. Test Setup Diagram:



8.3. **Tools Used:** Default DUT configuration tool for Web Server as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4. **Test Execution Steps**

- Power up the testbed
- The tester tries to access the Shell of Web Server.
- Consult the web server configuration to identify all directories used for CGI or other scripting components.
- The tester manually checks that the permission of the CGI directory and verify it by checking it in the browser.
- The tester will run some web scanning tools like Nikto, to get the detailed information and verify that the only executable CGI are working.
- The tester should create two directories: one for CGI scripts and one for static content.
- The tester should set execute permissions on the CGI directory to ensure that CGI scripts can run.
- The tester should ensure that the static content directory does not have execute permissions to prevent it from executing scripts.
- The tester should configure Apache to allow CGI script execution in the CGI directory
- The tester should create a test CGI script in the CGI directory to verify that it executes correctly and also in the static directory to ensure it does not execute.

9. **Expected Results for Pass:**

- The web server is configured such that only the CGI/Scripting directory has execute permissions in the web server
- NOTE: Generally, it is insufficient to set operating system permissions file rights to avoid that the web server executes code from a particular location. If only operating system access

rights are used, additional documentation, e.g. a best practice guide or OS manual, that this is sufficient should be provided.

10. **Expected Format of Evidence:** Log files and screen shots of test executions.

11. **Test Execution:**

Test Case Number: 01

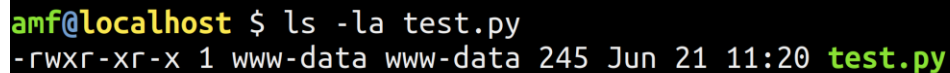
a. **Test Case Name:** TC_EXCLUSIVE_EXECUTE_RIGHTS_FOR_CGI

b. **Test Case Description:** To test whether the web server only has executed permissions on the CGI/Scripting directory.

c. **Execution Steps:**

1. The tester shall open any terminal emulator to access the DUT shell and use the following command.

- Command used: *ls -la <path of the CGI-BIN directory>/<available file name>*

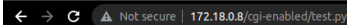


```
amf@localhost $ ls -la test.py
-rwxr-xr-x 1 www-data www-data 245 Jun 21 11:20 test.py
```

(Here we can see that owner of the file is allowed execution rights)

- We will open any browser and verify the following

o URL: *http://<IP address of the DUT>/<CGI-BIN Directory>/<available file name>*

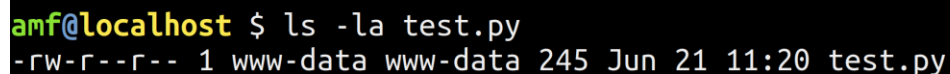


← → 🔒 Not secure | 172.18.0.8/cgi-enabled/test.py

CGI Script Test Page

2. The tester shall open any terminal emulator and use the following command.

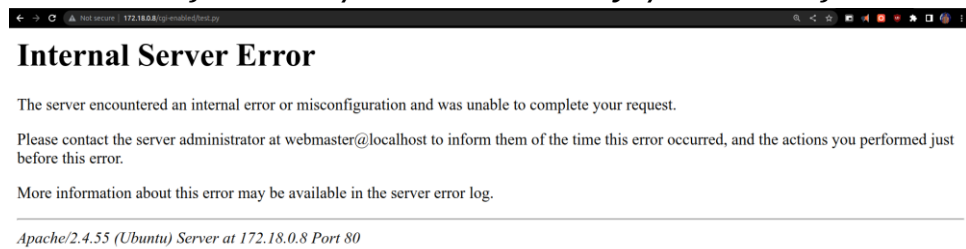
- Command used: *ls -la <path of the CGI-BIN directory>/<available file name>*



```
amf@localhost $ ls -la test.py
-rw-r--r-- 1 www-data www-data 245 Jun 21 11:20 test.py
```

(Here we can see that owner of the file does not have execution rights)

- We will open any browser and verify the following
- o URL: ***http://<IP address of the DUT>/<CGI-BIN Directory>/<available file name>***



- With proper authorization in the DUT we will access the DUT shell to view the logs

```
anf@localhost $ cat /var/log/apache2/error.log
[Wed Jun 21 11:24:27.784194 2023] [cgid:error] [pid 3150:tid 140511222773632] (13)Permission denied: AH01241: exec of '/var/www/html/cgi-enabled/test.py' failed
```

(Here we can see the error permission denied when the executable bit is off)

3. The tester should create two directories: one for CGI scripts and one for static content. For example:

1. CGI directory: `/var/www/cgi-bin`
2. Static content directory: `/var/www/static`

Command: `$sudo mkdir -p /var/www/cgi-bin$sudo mkdir -p /var/www/static`

3. The tester should set execute permissions on the CGI directory to ensure that CGI scripts can run. This can be done with: `sudo chmod 755 /var/www/cgi-bin`
4. The tester should ensure that the static content directory does not have execute permissions to prevent it from executing scripts. This can be done with: `sudo chmod 644 /var/www/static`
5. The tester should configure Apache to allow CGI script execution in the CGI directory.
6. Ensure that the CGI module is enabled: `sudo a2enmod cgi`
7. Modify the Apache configuration file: `sudo vi /etc/apache2/sites-available/000-default.conf`

Following should be in between: <VirtualHost *:80></VirtualHost *:80> tags

```
Alias /cgi-bin /var/www/cgi-bin<Directory "/var/www/cgi-bin">Options +ExecCGI
AddHandler cgi-script .cgiRequire all granted</Directory>Alias /static
/var/www/static
```

4. The tester should create a test CGI script in the CGI directory to verify that it executes correctly. For example:

```
>sudo bash -c 'cat > /var/www/cgi-bin/test-script.cgi <<EOL
>#!/bin/bash
>echo "Content-type: text/html"
```

```
>echo ""
>echo "<html><head><title>Test Script</title></head><body>"
>echo "<h1>CGI Script Executed Successfully</h1>"
>echo "</body></html>"
>EOL'
```

```
>sudo bash -c 'cat > /var/www/static/test-script.cgi <<EOL
>#!/bin/bash
>echo "Content-type: text/html"
>echo ""
>echo "<html><head><title>Test Script</title></head><body>"
>echo "<h1>CGI Script Executed Successfully</h1>"
>echo "</body></html>"
>EOL'
```

5. Make both the test script executable with:

```
sudo chmod 755 /var/www/cgi-bin/test-script.cgi
sudo chmod 755 /var/www/static/test-script.cgi
```

6. The tester should restart Apache to apply the changes: *sudo systemctl restart apache2*

7. The tester should verify the setup by visiting the following URL in a web browser:

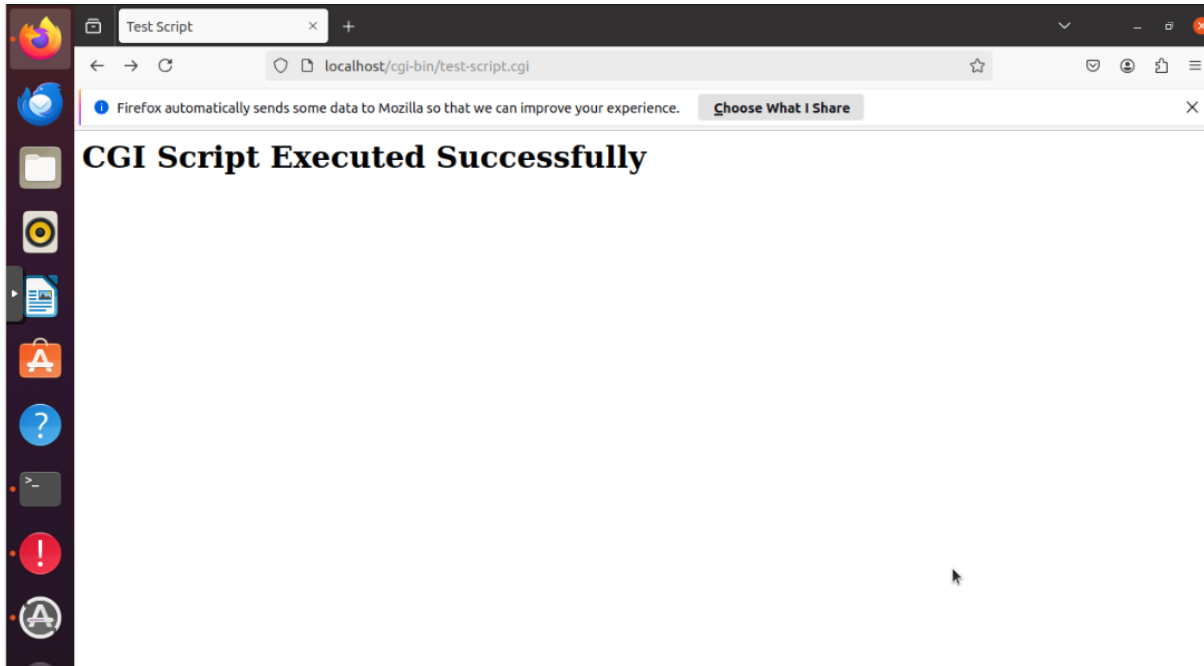
```
http://localhost/cgi-bin/test-script.cgi
http://localhost/static/test-script.cgi
```

d. Test Observations:

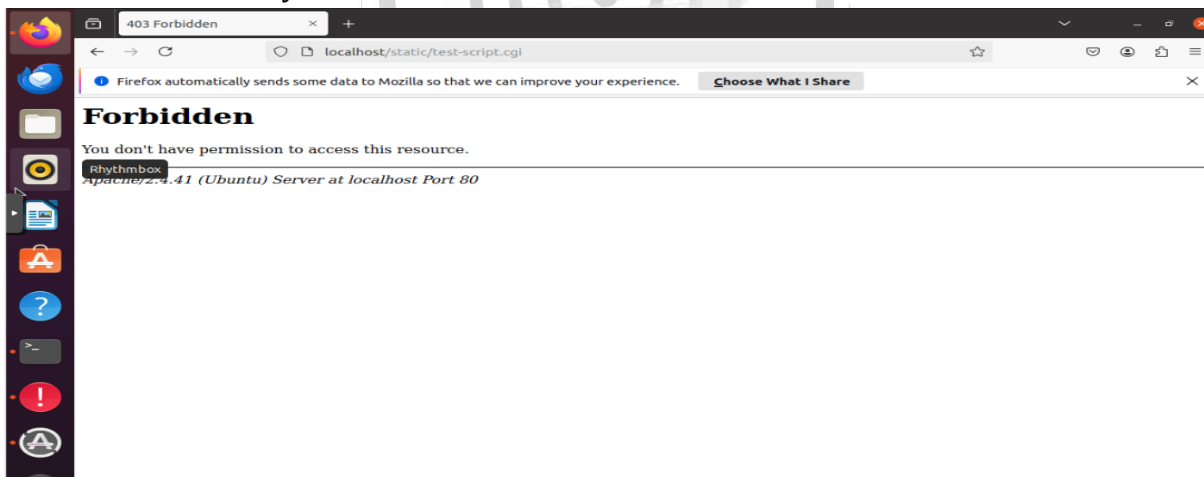
- When the executable bit is on the CGI script run with no error.
- When the executable bit is set to off the CGI script gave a 500 error.
- For step 3 onwards:

Securing Networks

- For CGI-BIN directory:



- For STATIC Directory:



12. Test Case Result:

Sl. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_EXCLUSIVE_EXECUTE_RIGHTS_FOR_CGI		

2.11.18 TSTP for HTTP User session

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.11.18
-----------------------------------	--------------------------	-------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >
 - 1) <ITSAR Section No & Name> Section 11: Web Server
 - 2) <Security Requirement No & Name > 2.11.18 HTTP User session
 - 3) <Requirement Description: >To protect user sessions, SMF shall support the following session ID and session cookie requirements:
 - i) The session ID shall uniquely identify the user and distinguish the session from all other active sessions.
 - ii) The session ID shall be unpredictable.
 - iii) The session ID shall not contain sensitive information in clear text (e.g., account number, social security, etc.).
 - iv) In addition to the Session Idle Timeout, SMF shall automatically terminate sessions after a configurable maximum lifetime. This maximum lifetime defines the maximum session span. When the maximum lifetime expires, the session shall be closed, the session ID shall be deleted and the user shall be forced to (re)authenticate in the web application and to establish a new session. The default value for this maximum lifetime shall be set to 8 hours.
 - v) Session IDs shall be regenerated for each new session (e.g., each time a user logs in).
 - vi) The session ID shall not be reused or renewed in subsequent sessions.
 - vii) The SMF shall not use persistent cookies to manage sessions but only session cookies. This means that neither the "expire" nor the "max-age" attribute shall be set in the cookies.
 - viii) Where session cookies are used the attribute 'HttpOnly' shall be set to true.

- ix) Where session cookies are used the 'domain' attribute shall be set to ensure that the cookie can only be sent to the specified domain.
- x) Where session cookies are used the 'path' attribute shall be set to ensure that the cookie can only be sent to the specified directory or sub-directory.
- xi) The SMF shall not accept session identifiers from GET/POST variables.
- xii) The SMF shall be configured to only accept server generated session ID.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. section 4.2.5.3]

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
 - Use command to get Application No/Version
 - Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)
- **Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)**

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

- Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) DUT Configuration:

```
amf@localhost $ apache2ctl -t -D DUMP_VHOSTS
VirtualHost configuration:
*:80                  172.18.0.8 (/etc/apache2/sites-enabled/000-default.conf:1)
```

- To get the hash of configuration file if the file is a ASCII text file
Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

- To get the hash of OS if using docker
Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6) Preconditions

- The tester has administrative privileges
- A tester machine is available.
- Test environment with a Web Browser.
- The Network Product uses a session ID that is communicated between the client and Network Product to establish and maintain a session.
- Documentation describing how a session is maintained and where the session ID is stored / and how this is communicated and after how long sessions expire.
- The documentation should describe the algorithm used to generate the session IDs

- 7) **Test Objective:** Verify that the above 12 session ID and session cookie requirements have been met.

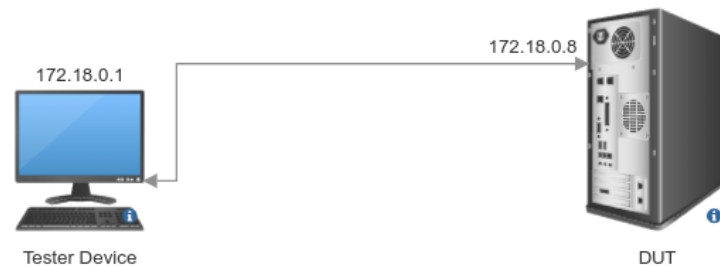
- 8) **Test Plan:**

8.1. **Number of Test Scenarios:**

8.1.1. Test Scenario for Cookie and Session ID

- This test scenario is regarding Cookie and Session ID and its best validation

8.2. **Test Setup Diagram:**



8.3. **Tools Used:** Default DUT tool to access DUT as per vendor. It can be command line, GUI or any other interface as specified in vendor documentation.

8.4. **Test Execution Steps**

- Power up the testbed
- The tester tries to access the Web Server in the browser.
- Access the Cookies and Session IDs and analyse them.
- The tester logs in repeatedly with different user IDs and a number of times with the same user ID in a row and collects the session IDs according to the documentation and the user IDs associated with them. The tester verifies that:
 - a. The session IDs are different between sessions of the same and different users;
 - b. The session IDs seems random based on his/her own experience. The tester may use tests like the bitstream test or the count-the-1s-tests from the diehard test suite. The tester documents how randomness was verified;
 - c. The session IDs are always different between sessions, also when the user ID is the same.
- The tester verifies that when session cookies are used
 - a. neither the "expire" or the "max-age" is set;
 - b. the 'HttpOnly' is set to true;
 - c. the 'domain' attribute is set to the correct domain;
 - d. the 'path' attribute is set to the correct directory or sub-directory.
- The tester verifies that it is impossible to:

- a. access a session by retrieving the session ID and communicating the session ID through a POST or GET variable.
- b. generate a session ID on the client by attempting to login with a custom generated session ID.
- c. keep a session alive for longer than the configured maximum lifetime (by default 8 hours).

9) **Expected Results for Pass:**

- A list of session IDs and user IDs that are different between sessions even when the tester has logged in with the
- same user and that are unpredictable as is confirmed by the entropy calculation.
- A confirmation from the tester that the correct variables are indeed set.
- A denied access to the tester when attempting the login via GET and POST when using and an expired session.

10) **Expected Format of Evidence:**

- Session IDs follow the rules 1-3, 5, 6.
- A session times out after 8 hours or sooner according to the documentation.
- The correct cookie settings are used.
- The network product does not accept customly generated session IDs and that session IDs over GET or POST are ignored.

11) **Test Execution:**

➤ **Test Case Number:** 01

a) **Test Case Name:** TC_SESSIONID_COOKIE_TESTS

b) **Test Case Description:** Verify that the above 12 session ID and session cookie requirements have been met.

c) **Execution Steps:**

The tester shall open any Web Browser (Mozilla Firefox, Google Chrome) and the browser must have an Extension/Addon installed like **Cookie-Editor** or similar must be installed in the tester browser.



Cookie-Editor

cookie-editor.cgagnier.ca Featured

★★★★★ 226 ⓘ | Developer Tools | 1,000,000+ users

The tester shall open any Web Browser (Mozilla Firefox, Google Chrome) and then visit the DUT's web application using the URL.

After login to the web application open the addon and search for the cookies and find the session id (Refer the vendor documentation to find the name of session id), generally SID, SESSIONID is

Cookie Editor ☐ Show Advanced

^ sessionId

Name
sessionId

Value
8cb55158-0fa8-f375-9bd1-8736a94ae607

Domain
teams.microsoft.com

Path
/

Expiration

Same Site

☒ Host Only ☒ Session ☒ Secure ☐ Http Only

▼ skypetoken_asm

+ [trash icon] [copy icon] [paste icon]

being u

- Here we can see that the session id in the cookie under the name ***sessionId***

The tester logs in repeatedly with different user IDs and a number of times with the same user ID in a row and collects the session IDs according to the documentation and the user IDs associated with them. The tester verifies that:

- The session IDs are different between sessions of the same and different users;
- The session IDs seems random based on his/her own experience. The tester may use tests like the bitstream test or the count-the-1s-tests from the diehard test suite. The tester documents how randomness was verified;

The tester verifies that when session cookies are used

- neither the "expire" or the "max-age" is set;
- the 'HttpOnly' is set to true;
- the 'domain' attribute is set to the correct domain;
- the 'path' attribute is set to the correct directory or sub-directory.

Cookie Editor ☐ Show Advanced

^ sessionId

Name
sessionId

Value
8cb55158-0fa8-f375-9bd1-8736a94ae607

Domain
teams.microsoft.com

Path
/

Expiration

Same Site
Default

☒ Host Only ☒ Session ☒ Secure ☒ Http Only

▼ skypetoken_asm

+ [trash] [duplicate] [import]

Open Postman or any other tool to send the Http GET and POST request.

f. Send a GET request with a GET parameter of Sessionid.

GET http://localhost:8080/?sessionId=8cb55158-0fa8-f375-9bd1-8736a94ae607 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

This request does not have a body

Body Cookies Headers (3) Test Results

Status: 401 Unauthorized Time: 7 ms Size: 182 B Save Response

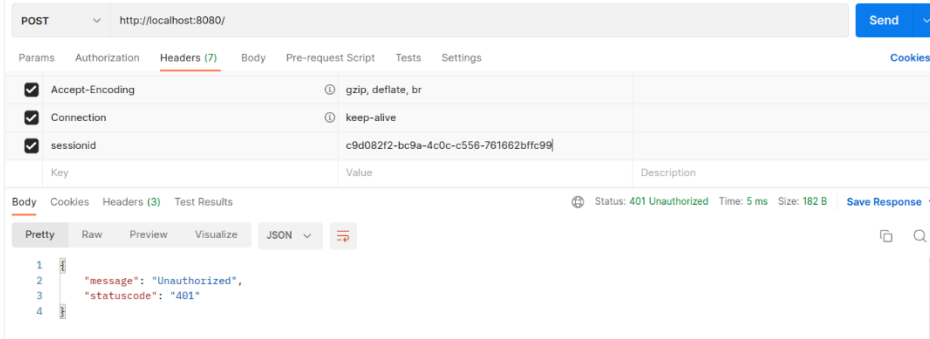
```

1 {
2   "message": "Unauthorized",
3   "statusCode": "401"
4 }

```

(We can see that upon sending a **valid sessionId** in a get parameter it returned **Unauthorized**)

g. Send a POST request with a Sessionid in Header.



(We can see that upon sending a **valid sessionid** in a header it returned **Unauthorized**)

Above test should be performed for custom generated session ID and those tests shall give Unauthorized message any other message the test case will fail.

keep a session alive for longer than the configured maximum lifetime (by default 8 hours).

And verify that when the maximum lifetime expires, the session shall be closed, the session ID shall be deleted, and the user shall be forced to (re)authenticate in the web application and to establish a new session.

e. **Test Observations:**

- The tester A list of session IDs and user IDs that are different between sessions even when the tester has logged in with the same user and that are unpredictable as is confirmed by the entropy calculation.
- It Is verified that the correct variables are set.
- It is verified that the DUT denied access to the tester when attempting to login via custom SESSIONID and using GET and POST requests.

12) **Test Case Result:**

Sl. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_SESSIONID_COOKIE_TESTS		

2.12.1 TSTP for Evaluation of No code execution or inclusion of external resources by JSON parsers

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.12.1
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
 - <DUT Software Version:>
 - <Digest Hash of OS>
 - <Digest Hash of Configuration>
 - <Applicable ITSAR: >
 - <ITSAR Version No:>
 - <OEM Supplied Document list: >
1. **<ITSAR Section No & Name>** Section 12: General SBA/SBI Aspects
 2. **<Security Requirement No & Name >** 2.12.1 No code execution or inclusion of external resources by JSON parsers
 3. **<Requirement Description: >** Parsers used by Network Functions (NF) shall not execute JavaScript or any other code contained in JSON objects received on Service Based Interfaces (SBI). Further, these parsers shall not include any resources external to the received JSON object itself, such as files from the NF's filesystem or other resources loaded externally.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section - 4.3.6.2]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No

Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

- Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: docker images --digests (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. Preconditions

- The tester has the privileges to log in the network product and to access to the all system resources (e.g. log files)
- A list of all available network services containing at least the following information shall be included in the documentation accompanying the Network Product:
 - o all interfaces providing IP-based protocols;
 - o the available transport layer protocols on these interfaces;
 - o their open ports and associated services in the form of an OpenAPI3.0 interface specification;
- The tester should have access to an effective Web Application Security (WAS) test tool that allows to generate HTTP messages exploiting JSON parsers that do not prevent the above-mentioned scenarios of code execution and loading external resources. The accredited test lab is expected to have sufficient expertise to recognize the level of effectiveness of the available tools.
- A network traffic analyser on the network product (e.g., TCPDUMP/WIRESHARK) or an external traffic analyser directly connected to the network product and on a tester machine is available.

7. **Test Objective:** NFs implementing SBI transfer application data serialized as JSON objects. When receiving such data, an NF parses this JSON representation and creates equivalent internal data structures. Since the contents of the JSON objects must be considered untrusted, blindly executing code fragments or loading resources from a local path or Uniform Resource Identifier (URI) must not be possible.

8. Test Plan:

8.1. **Number of Test Scenarios:** *Securing Networks*

8.1.1. Test Scenario for Evaluation of No code execution or inclusion of external resources by JSON parsers

8.2. **Test Setup Diagram:**



8.3 **Tools Used:** Postman, Wireshark.

8.4 **Test Execution Steps**

1. Execution of available WAS test tools against the network product's API endpoints via its Service Based Interfaces.
2. Using a network traffic analyzer on the network product, e.g., TCPDUMP or an external traffic analyzer directly connected to the network product, the tester verifies that no external resources get loaded during JSON parsing.
3. Depending on the actual JavaScript code in the HTTP message, the tester verifies that the network product does not execute any of the contained actions

9. **Expected Results for Pass:**

- The NF does not load any resources external to the JSON object itself.
- The NF does not execute any JavaScript code contained in JSON objects

10. **Expected Format of Evidence:** A testing report provided by the testing agency which will consist of the following information:

- The used tool(s) name and version information
- Settings and configurations used
- The output log file of the chosen tool that displays the results (passed/failed).
- Screenshot
- Test result (Passed or not)

11. **Test Execution:**

➤ Test Case Number: 01

a. **Test Case Name:** TC_JSON_PARSER_CODE_EXEC_INCL

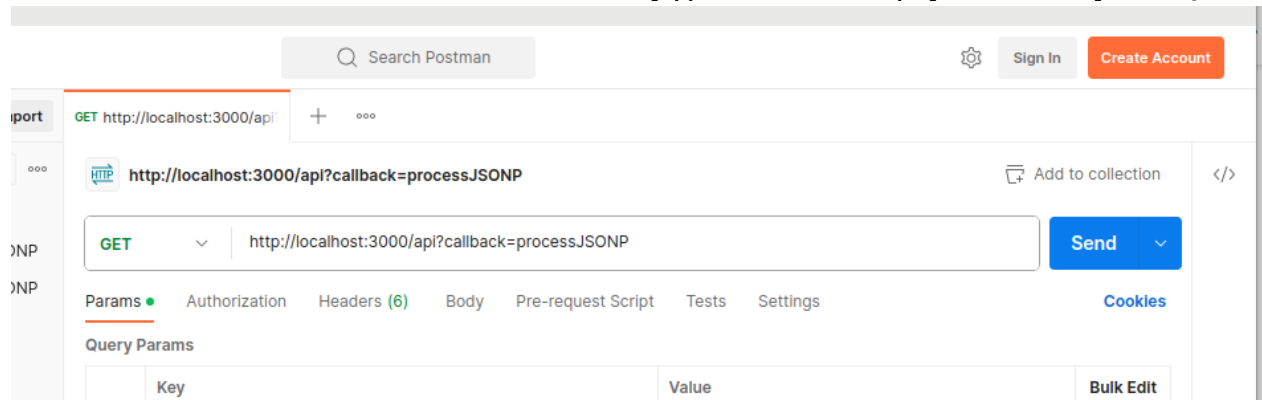
b. **Test Case Description:** NFs implementing SBI transfer application data serialized as JSON objects. When receiving such data, an NF parses this JSON representation and creates equivalent internal data structures. Since the contents of the JSON objects must

be considered untrusted, blindly executing code fragments or loading resources from a local path or Uniform Resource Identifier (URI) must not be possible. There are 3 methods for WAS: - Static Application Security Testing (SAST) Tools Dynamic Application Security Testing (DAST) Tools (Primarily for web apps) Interactive Application Security Testing (IAST) Tools - (Primarily for web apps and web APIs)

c. Execution Steps:

i. Execution of available WAS test tools against the network product's API endpoints via its Service Based Interfaces. Here for simulation, we sent a get request to execute a callback function on the server:

Our DUT had port 3000 Opened for API queries, so we used the following command to execute the callback function on the server DUT: ***http://localhost:3000/api?callback=processJSONP***



(We used Postman for simulation purpose, but actual WAS tool must be used for proper testing)

ii. Using a network traffic analyser on the network product, e.g., Wireshark or an external traffic analyser directly connected to the network product, the tester verifies that no external resources get loaded during JSON parsing.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.17.0.1	172.17.0.2	TCP	74	50356 → 3000 [SYN] Seq=0 Win=64240 Len=0 M
2	0.000023934	172.17.0.2	172.17.0.1	TCP	74	3000 → 50356 [SYN, ACK] Seq=0 Ack=1 Win=65
3	0.000041281	172.17.0.1	172.17.0.2	TCP	66	50356 → 3000 [ACK] Seq=1 Ack=1 Win=64256 L
4	0.000143673	172.17.0.1	172.17.0.2	TCP	74	50364 → 3000 [SYN] Seq=0 Win=64240 Len=0 M
5	0.000156629	172.17.0.2	172.17.0.1	TCP	74	3000 → 50364 [SYN, ACK] Seq=0 Ack=1 Win=65
6	0.000166378	172.17.0.1	172.17.0.2	TCP	66	50364 → 3000 [ACK] Seq=1 Ack=1 Win=64256 L
7	0.000170135	172.17.0.1	172.17.0.2	TCP	66	50356 → 3000 [FIN, ACK] Seq=1 Ack=1 Win=64
8	0.000323335	172.17.0.1	172.17.0.2	HTTP	292	GET /api?callback=processJSONP HTTP/1.1
9	0.000330641	172.17.0.2	172.17.0.1	TCP	66	3000 → 50364 [ACK] Seq=1 Ack=227 Win=65024
10	0.000748678	172.17.0.2	172.17.0.1	HTTP	343	HTTP/1.1 200 OK (text/html)
11	0.000759495	172.17.0.1	172.17.0.2	TCP	66	50364 → 3000 [ACK] Seq=227 Ack=278 Win=641
12	0.000877925	172.17.0.2	172.17.0.1	TCP	66	3000 → 50356 [FIN, ACK] Seq=1 Ack=2 Win=65
13	0.000882917	172.17.0.1	172.17.0.2	TCP	66	50356 → 3000 [ACK] Seq=2 Ack=2 Win=64256 L

(Here we see we a get a 200 OK response, which means the callback function was executed and shown as text/html on the server. This should not have happened if above 200 OK is received test fails)

iii. Depending on the actual JavaScript code in the HTTP message, the tester verifies that the network product does not execute any of the contained actions.

Here we have used example of callback functions. A callback function is a function passed into another function as an argument, which is then invoked inside the outer function to complete some kind of routine or action.

The consumer of a callback-based API writes a function that is passed into the API. The provider of the API (called the caller) takes the function and calls back (or executes) the function at some point inside the caller's body. The caller is responsible for passing the right parameters into the callback function. The caller may also expect a particular return value from the callback function, which is used to instruct further behaviour of the caller.

A Callback function is not a valid test to pass this TSTP. It is just for simulation purpose.

d. Test Observations:

- Test Passes if response is 400 (Bad Request):

```

7 0.000174358 172.17.0.1 172.17.0.2 TCP 74 39530 → 3000 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=
8 0.000181921 172.17.0.2 172.17.0.1 TCP 74 3000 → 39530 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460
9 0.000190131 172.17.0.1 172.17.0.2 TCP 66 39530 → 3000 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1965661
10 0.000501315 172.17.0.1 172.17.0.2 HTTP 284 GET /api?callback=safe HTTP/1.1
11 0.000507114 172.17.0.2 172.17.0.1 TCP 66 3000 → 39530 [ACK] Seq=1 Ack=219 Win=65024 Len=0 TSval=36157
12 0.001061352 172.17.0.2 172.17.0.1 TCP 66 3000 → 39526 [ACK] Seq=1 Ack=2 Win=65280 Len=0 TSval=3615728
13 0.004782524 172.17.0.2 172.17.0.1 HTTP 313 HTTP/1.1 400 Bad Request (text/html)
14 0.004793790 172.17.0.1 172.17.0.2 TCP 66 39530 → 3000 [ACK] Seq=219 Ack=248 Win=64128 Len=0 TSval=196
15 0.005369418 172.17.0.2 172.17.0.1 TCP 66 3000 → 39526 [FIN, ACK] Seq=1 Ack=2 Win=65280 Len=0 TSval=36
16 0.005378261 172.17.0.1 172.17.0.2 TCP 66 39526 → 3000 [ACK] Seq=2 Ack=2 Win=64256 Len=0 TSval=1965661
17 5.009994222 172.17.0.2 172.17.0.1 TCP 66 3000 → 39530 [FIN, ACK] Seq=248 Ack=219 Win=65024 Len=0 TSva
18 5.011202897 172.17.0.1 172.17.0.2 TCP 66 39530 → 3000 [FIN, ACK] Seq=219 Ack=249 Win=64128 Len=0 TSva
19 5.011232535 172.17.0.2 172.17.0.1 TCP 66 3000 → 39530 [ACK] Seq=249 Ack=220 Win=65024 Len=0 TSval=361

```

- Else Test Fails

12. Test Case Result:

Sl. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_JSON_PARSER_CODE_EXEC_INCL		

2.12.2 TSTP Report for Evaluation of Validation of the unique key values in IEs

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.12.2
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **ITSAR Section No & Name:** Section 12 General SBA/SBI Aspects
2. **Security Requirement No & Name:** 2.12.2 Validation of the unique key values in IEs
3. **Requirement Description:** For data structures where values are accessible using names (sometimes referred to as keys), e.g. a JSON object, the name shall be unique. The occurrence of the same name (or key) twice within such a structure shall be an error and the message shall be rejected.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section - 4.3.6.3]

4. **DUT Confirmation Details:**

Securing Networks

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No

Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal
```

5. **DUT Configuration:** Check the Operating system and its version for the DUT to know the commands for setting access permissions for data and application execution.

To get the hash of configuration file if the file is an ASCII text file.

Command - **sha256sum DUT_config.conf**

Digest Hash of Tested Configuration: DUT_config.conf: -

19be54b42975f7f414c53a280373aa9466398781ecd77a2e9d090dbc6dbfdd8

To get the hash of OS if using docker Command - docker images --digests

Digest Hash of OS: DUT_IMAGE: -

fd363f0e0d146c869d60649bc4f42e7008829d9d67a5dfdaf3b38ad24af7d53a

Latest version of python should be installed.

6. **Preconditions:**

- Vendor provides the list of all the platforms which can receive/send duplicate keys in message IE payload and needs to be checked.

- Test environment with network product under test. Rest of the network and network products may be simulated.
7. **Test Objective:** To ensure that implementation fulfills the requirements, that is no duplicate values should be sent in structures where values are accessible using name.
8. **Test Plan:**
- 8.1 Number of test scenarios/test cases: 1
 - 8.2 Tools used: Python, Command Line of the DUT.
 - 8.3 Test case Execution: The accredited evaluator's test lab is required to execute the following steps:
 - a. The test equipment sends requests with duplicate keys in message IE payload to the network product under test.
 - b. The test equipment sends valid requests to network product under test.
9. **Expected Results for Pass:**
- 1) Network product under tests responses with an error message.
 - 2) Network product under test still responses normally to valid requests.
10. **Expected Format of Evidence:** A testing report provided by the testing agency which will consist of the following information:
- The used tool(s) name and version information,
 - Settings and configurations used
 - The output log file of the chosen tool that displays the results (passed/failed).
 - Test result (Passed or not)
 - Log/evidence tracing possible crashes
 - Information of any input causing unspecified, undocumented, or unexpected behavior.
11. **Test Execution:**
- **Test Case Number:** 1
 - a. **Test Case Name:** TC_VALID_UNIQUE_KEY_IE
 - b. **Test Case Description:** To ensure that API implementation fulfills the requirements, that is no duplicate values should be sent in structures where values are accessible using name.
 - c. **Test Execution:** The accredited evaluator's test lab is required to execute the following steps:
 - Runs the script that sends requests with duplicate keys in message IE payload to the network product under test.
 - Runs the script that sends valid requests to the network product under test.

d. Test Observations:

- **Case 1:** The test equipment sends requests with duplicate keys in message IE payload to the network product under test. As shown in the screenshot below, the network product should throw an error that a duplicate key is found.

```
supriya@supriya-Super-Server:~/TSTP/Validation_2.12.2$ python server.py
Socket created ...
socket is listening
('got request from ', ('127.0.0.1', 50974))
('Duplicate key specified:', u'IP')
Duplicate values found, cannot parse this request.
supriya@supriya-Super-Server:~/TSTP/Validation_2.12.2$
```

```
supriya@supriya-Super-Server:~/TSTP/Validation_2.12.2$ python client.py
({'IP': '192.168.1.5', 'IP': '192.168.1.8'}, 'was sent!')
supriya@supriya-Super-Server:~/TSTP/Validation_2.12.2$
```

- **Case 2:** The test equipment sends valid requests to the network product under test. Network product successfully parses the request and no error is thrown.

```
supriya@supriya-Super-Server:~/TSTP/Validation_2.12.2$ python server.py
Socket created ...
socket is listening
('got request from ', ('127.0.0.1', 50966))
('Json received -->', '{"SIP": "192.168.1.5", "DIP": "192.168.1.8"}')
supriya@supriya-Super-Server:~/TSTP/Validation_2.12.2$
```

```
supriya@supriya-Super-Server:~/TSTP/Validation_2.12.2$ python client.py
({'SIP': '192.168.1.5', 'DIP': '192.168.1.8'}, 'was sent!')
supriya@supriya-Super-Server:~/TSTP/Validation_2.12.2$
```

12. Test Case Result:

Sl. No	OUTCOME OF RUNNING THE SCRIPT (CASE 1/CASE2)	PASS/FAIL
1	TC_VALID_UNIQUE_KEY_IE	

2.12.3 TSTP Report for Validation of the IEs limits

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.12.3
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **ITSAR Section No & Name:** Section 12 General SBA/SBI Aspects
2. **Security Requirement No & Name:** 2.12.3 Validation of the IEs limits
3. **Requirement Description:** The valid format and range of values for each IE, when applicable, shall be defined unambiguously:
 - For each message the number of leaf IEs shall not exceed 16000.
 - The maximum size of the JSON body of any HTTP request shall not exceed 16 million bytes.
 - The maximum nesting depth of leaves shall not exceed 32.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section - 4.3.6.4]

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No

Command used: ifconfig -a (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. **DUT Configuration:** Check the Operating system and its version for the DUT to know the commands for setting access permissions for data and application execution.

To get the hash of configuration file if the file is an ASCII text file.

Command - **sha256sum DUT_config.conf** Digest Hash of Tested Configuration:

DUT_config.conf: -

19be54b42975f7f414c53a280373aa9466398781ecd77a2e9d090dbc6dbfdd8

To get the hash of OS if using docker Command - **docker images --digests**

Digest Hash of OS: DUT_IMAGE: -

fd363f0e0d146c869d60649bc4f42e7008829d9d67a5dfdaf3b38ad24af7d53a

Latest version of python should be installed.

6. **Preconditions:** Test environment with network product under test. Rest of the network and network products may be simulated.

7. **Test Objective:** To ensure that API implementation fulfills the requirements, that is for each message the number of leaf IEs shall not exceed 16000. The maximum size of the JSON body of any HTTP request shall not exceed 16 million bytes. The maximum nesting depth of leaves shall not exceed 32.

8. **Test Plan:**

8.1 Number of test scenarios/test cases: 1

8.2 Tools used: Python, Command Line of the DUT.

8.3 Test case Execution: The accredited evaluator's test lab is required to execute the following steps:

- a. The test equipment sends requests with the following characteristics in message IE payload to the network product under test:
 - i. For each message the number of leaf IEs shall not exceed 16000.
 - ii. The maximum size of the JSON body of any HTTP request shall not exceed 16 million bytes.
 - iii. The maximum nesting depth of leaves shall not exceed 32.
- b. The test equipment sends valid requests to network product under test.

9. **Expected Results for Pass:**

- 1) Network product under tests responses with an error message.
- 2) Network product under test still responses normally to valid requests.

10. **Expected Format of Evidence:** A testing report provided by the testing agency which will consist of the following information:

- The used tool(s) name and version information,
- Settings and configurations used
- The output log file of the chosen tool that displays the results (passed/failed).
- Test result (Passed or not)
- Log/evidence tracing possible crashes
- Information of any input causing unspecified, undocumented, or unexpected behavior.

11. **Test Execution:**

- a. **Test Case Number:** 1
- b. **Test Case Name:** TC_VALID_NUMBER_IE

- c. **Test Case Description:** To ensure that API implementation fulfills the requirements, that is no duplicate values should be sent in structures where values are accessible using name.
- d. **Test Execution:** The accredited evaluator's test lab is required to execute the following steps:
- Runs the script that sends requests with with the following characteristics in message IE payload to the network product under test:
 - For each message the number of leaf IEs will exceed 16000.
 - The maximum size of the JSON body of any HTTP request will exceed 16 million bytes.
 - The maximum nesting depth of leaves will exceed 32.
 - Runs the script that sends valid requests to the network product under test.
- e. **Test Observations:**

Case 1: The test equipment sends requests with with the following characteristics in message IE payload to the network product under test:

- For each message the number of leaf IEs will exceed 16000.
- The maximum size of the JSON body of any HTTP request will exceed 16 million bytes.
- The maximum nesting depth of leaves will exceed 32.

As shown in the screenshot below, the network product should throw an error that:
 The number of leaf IEs exceeds 16000.
 The maximum size of the JSON body of any HTTP request exceeds 16 million bytes.
 The maximum nesting depth of leaves exceeds 32.

```

newslab-ks422@newslab-ks422: ~/TSTP_5G
newslab-ks422@newslab-ks422:~/TSTP_5G$ python3 server_leaf.py
Socket created ...
socket is listening
got request from ('127.0.0.1', 35074)
b'{"IP":"192.168.1.5", "IP":"192.168.1.8"}'
Duplicate key specified: IP
The number of leaf IEs exceeds 16000 , cannot parse this request.

newslab-ks422@newslab-ks422:~/TSTP_5G
newslab-ks422@newslab-ks422:~/TSTP_5G$ python3 client_leaf.py
b'{"IP":"192.168.1.5", "IP":"192.168.1.8"}' was sent!
newslab-ks422@newslab-ks422:~/TSTP_5G$
  
```

Case 2: The test equipment sends valid requests to the network product under test.
 Network product successfully parses the request and no error is thrown.

```

newslab-ks422@newslab-ks422:~/TSTP_5G
newslab-ks422@newslab-ks422:~/TSTP_5G$ python3 server_leaf.py
Socket created ...
socket is listening
got request from ('127.0.0.1', 34234)
b'{"SIP":"192.168.1.5", "DIP":"192.168.1.8"}'
Json received --> {"SIP":"192.168.1.5", "DIP":"192.168.1.8"}
newslab-ks422@newslab-ks422:~/TSTP_5G$

newslab-ks422@newslab-ks422:~/TSTP_5G
newslab-ks422@newslab-ks422:~/TSTP_5G$ python3 client_leaf.py
b'{"SIP":"192.168.1.5", "DIP":"192.168.1.8"}' was sent!
newslab-ks422@newslab-ks422:~/TSTP_5G$
  
```

12. Test Case Result:

Sl.No	OUTCOME OF RUNNING THE SCRIPT (CASE 1/CASE2)	PASS/FAIL
1	TC_VALID_NUMBER_IE	

2.12.4 TSTP Report for Evaluation of Protection at the transport layer

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.12.4
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 12 - General SBA/SBI Aspects
2. **<Security Requirement No & Name >** 2.12.4 Protection at the transport layer
3. **<Requirement Description: >** NF Service Request and Response procedure shall support mutual authentication between NF consumer and NF producer.

All network functions shall support TLS. Network functions shall support both server-side and client-side certificates.

Authentication between network functions within one PLMN can use the following method: - If the PLMN uses protection at the transport layer, authentication provided by the transport layer protection solution shall be used for authentication between NFs.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.2.2.2]

Note: This may not be applicable for UPF and N3IWF.

4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No

Command used: **ifconfig** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

Command used : **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5. DUT Configuration

Run NMAP or any port scanning tool to get running services

Command used: **Nmap -p 443 <IP Address of DUT>**


```

[ashwini@ashwini-news]~$ nmap -p 443 172.18.0.8
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-23 11:15 IST
Nmap scan report for 172.18.0.8
Host is up (0.00034s latency).

PORT      STATE SERVICE
443/tcp   open  https

Nmap done: 1 IP address (1 host up) scanned in 0.06 seconds

```

Command used to read the installed certificates in DUT regarding TLS `cat /etc/ssl/certs/ca-certificates.crt`

```

amf@localhost $ cat /etc/ssl/certs/ca-certificates.crt
-----BEGIN CERTIFICATE-----
MIIH0zCCBbugAwIBAgIIXsO3pkN/pOAwdQYJKoZIhvcNAQEFBQAwQjESMBAGA1UE
AwwJQUJUNDVlLJBVS0xMRwYDQYDQDAdQSO1BQ0NWMMQ0wCwYDVQQKDARBQ0NWMMQsw
CQYDVQQGEWJFUzAeFw0xMTA1MDUwOTMzMzdaFw0zMDEyMzdaMEIxEjAQ
BgNVBAMMCUFDDQ1ZSQUlaMTEQMA4GA1UECwwHUETJQUJUNDVjENMA5GA1UECgwEQUND
VjELMAKGA1UEBhMCRVMwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwAwggIKAoICAQCb
qau/YUqXry+XZpp0X9DZlv3P4uRm7x8fRzPCRKPFmt4ftVTdFXxpNRFvu8gMjmoY
HtiP2Ra8EEg2XPBjs5BaXCQ316PWylxufEBcoSwfDtNgM3802/J+Nq2DoLSRYWo
G2ioPej0RGy9ocLLA76MPHMAHN9KSMDjIgro6TenGEyxQ0jVn8ETdkXhBilyNpA
lHPrzg5XPA0B0p0KoVdDaaxXbXmQe0W1tDvYvEyNKKGno6e6Ak4l0Squ7a4Dlrrh
IA8wKFSVf+DuzgpmndFALW4lr50awQUZ0m/A8p/4e7MCQvtQqR0tkw8jq8bBD5L/
0KIV9VMJcRz/RR0E5iZe+OCIHAR8Fraocwa48G0EAQDGWuzndN9wrqODJerWx5eH
k6fGioozL2A3ED6XPm4pFdahD9GILBKfb6qkxLrQaLjLUPTAYVtjrs78yM2x/47
4KElB0iryYl0/wiPgL/AlmXz7uxLaL2diMMxs0Dx6M/20Luc5NF/10VYm3z61PM0
m3WR5LpSLhl+0fXNWhn8ugb2+1Ko55ke3fj5tItQo05iifCHJPqDQsGH+tUtKSpa
cXpkatcnYGMN285J9Y0fkIkyF/hzQ7jSwP0GYdbhdQrqewZ2iE9x6wQl1gpaepPl
uUsXQA+xtrn13k/c4L0s0xFwYIRKQ26ZIMApCqRAZQIDAQABo4ICyzCCAscwfQYI
KwYBBQUHAQEETBvMEwGCCsGAQUFBzACHkBoDHRwOi8vd3d3LmFjY3YuZXNvZmZl
ZWZkbWwL0FyY2hpdmd9zL2NlcnRpZmljYWRvcy9yYWL6YWNjdjEuY3J0MB8GCCsG
AQUBFzABhhNodHRwOi8vb2Nzc5hY2N2LmVzMB0GA1UdDgQWBBS7Tj3zcnk1X2

```

List of TLS certificates present in DUTs `/etc/ssl/certs/`

```

amf@localhost $ ls /etc/ssl/certs/
002c0b4f.0      Entrust_Root_Certification_Authority_-_G4.pem
02265526.0      GDCA_TrustAUTH_R5_ROOT.pem
062cdee6.0      GLOBALTRUST_2020.pem
064e0aa9.0      GTS_Root_R1.pem
06dc52d5.0      GTS_Root_R2.pem
08063a00.0      GTS_Root_R3.pem
09789157.0      GTS_Root_R4.pem
0a775a30.0      GlobalSign_ECC_Root_CA_-_R4.pem
0b1b94ef.0      GlobalSign_ECC_Root_CA_-_R5.pem
0b9bc432.0      GlobalSign_Root_CA.pem
0bf05006.0      GlobalSign_Root_CA_-_R3.pem
0f5dc4f3.0      GlobalSign_Root_CA_-_R6.pem
0f6fa695.0      GlobalSign_Root_E46.pem
1001acf7.0      GlobalSign_Root_R46.pem
106f3e4d.0      Go_Daddy_Class_2_CA.pem
14bc7599.0      Go_Daddy_Root_Certificate_Authority_-_G2.pem
18856ac4.0      HARRICA_TLS_ECC_Root_CA_2021.pem
1d3472b9.0      HARRICA_TLS_RSA_Root_CA_2021.pem
1e08bf01.0      Hellenic_Academic_and_Research_Institutions_ECC_RootCA_2015.pem
1e09b511.0      Hellenic_Academic_and_Research_Institutions_RootCA_2015.pem
244b5494.0      HPKX_Root_CA_-_G1.pem
2923b3f9.0      Hongkong_Post_Root_CA_1.pem
2ae6433e.0      Hongkong_Post_Root_CA_3.pem
2b349938.0      ISRG_Root_X1.pem
32888f65.0      ISRG_Root_X2.pem
3513523f.0      IdenTrust_Commercial_Root_CA_1.pem
3bde41ac.0      IdenTrust_Public_Sector_Root_CA_1.pem
3bde41ac.1      Izenpe.com.pem

```

To get the hash of configuration file if the file is a ASCII text file

Command used: `sha256sum SMF_config.conf` (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30  AMF_Config.conf
```

To get the hash of OS if using docker

Command used: docker images --digests (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. Preconditions

- Network product documentation containing information about supported TLS protocol and certificates is provided by the vendor.
- A peer/tester device configured for communication with DUT over TLS shall be available
- The tester shall base the tests on the profile defined by 3GPP in Annex E of TS 33.310 with the restriction that it shall be compliant with the profile given by HTTP/2 as defined in RFC 7540

7. **Test Objective:** To verify that the DUT communicates with its peer network function only after the mutual authentication is carried out over TLS.

8. Test Plan:

8.1. Number of Test Case Scenarios

8.1.1. Test Scenario for TLS: The requirement is only to be tested with TLS protocol

8.2. Test Setup Diagram



8.3. Tools Used:- Wireshark, Curl and OpenSSH at Tester Device

8.4. Test Execution Steps

- Launch the Wireshark app on the tester device.
- Power up the testbed in order to capture the packets exchanged between DUT and its peer network functions (Case 1).

- Generate a fake certificate at tester device using OpenSSH commands
- Replace certificate in the Peer NF with the fake certificate and repeat the above procedure (Case 2)
- Tester tries to contact DUT using HTTP i.e without using TLS (Case 3)

9. **Expected Results for Pass:**

- Case 1: Peer NF and DUT mutually authenticate each other over TLS before exchanging any messages.
- Case 2: Peer NF and DUT are not able to mutually authenticate each other over TLS and hence no messages are exchanged.
- Case 3: Peer NF and DUT are not able to establish a connection between them

10. **Expected Format of Evidence:** Screenshots of Wireshark and Pcap files capturing the TLS handshake.

11. **Test Execution:**

- **Test Case Number:** 01
- a. Test Case Name:** TC1_PROTECTION_AT_TRANSPORT_LAYER
- b. Test Case Description:** DUT should allow communication with other NFs only after successful mutual authentication via TLS.
- c. Execution Steps:**
 - While the test bed is running, the tester should put appropriate display filter (IP address of DUT & protocol filter as TLS) in Wireshark to verify the successful completion of TLS Handshake which is possible only after mutual authentication is successful.
 - The tester shall then interact with DUT using the API calls provided in the 3GPP API CALLS document of DUT.
 - Example:
- ***curl --cacert <path to ca cert> --cert <path to tester cert> --key <path to tester key> https://<DUT address along with port>/<path>***

50	10.977045	172.18.0.5	172.18.0.4	TLSv1.3	306 Client Hello
52	10.978004	172.18.0.4	172.18.0.5	TLSv1.3	165 Hello Retry Request, Change Cipher Spec
54	10.978136	172.18.0.5	172.18.0.4	TLSv1.3	345 Change Cipher Spec, Client Hello
56	10.979879	172.18.0.4	172.18.0.5	TLSv1.3	2456 Server Hello, Application Data, Application Data, Application Data, Ap
58	10.980423	172.18.0.5	172.18.0.4	TLSv1.3	140 Application Data
60	10.980534	172.18.0.4	172.18.0.5	TLSv1.3	224 Application Data, Application Data
61	10.980535	172.18.0.5	172.18.0.4	TLSv1.3	140 Application Data
64	10.980659	172.18.0.5	172.18.0.4	TLSv1.3	186 Application Data
66	10.980696	172.18.0.4	172.18.0.5	TLSv1.3	112 Application Data
68	10.980762	172.18.0.5	172.18.0.4	TLSv1.3	97 Application Data
71	10.980833	172.18.0.4	172.18.0.5	TLSv1.3	268 Application Data
73	10.980763	172.18.0.5	172.18.0.4	TLSv1.3	195 Application Data
83	11.005625	172.18.0.5	172.18.0.4	TLSv1.3	306 Client Hello
85	11.005700	172.18.0.4	172.18.0.5	TLSv1.3	165 Hello Retry Request, Change Cipher Spec
87	11.005938	172.18.0.5	172.18.0.4	TLSv1.3	345 Change Cipher Spec, Client Hello
89	11.007222	172.18.0.4	172.18.0.5	TLSv1.3	2456 Server Hello, Application Data, Application Data, Application Data, Ap
91	11.007918	172.18.0.5	172.18.0.4	TLSv1.3	140 Application Data
93	11.007977	172.18.0.5	172.18.0.4	TLSv1.3	140 Application Data

Content Type: Handshake (22)
Version: TLS 1.0 (0x0301)
Length: 235
+ Handshake Protocol: Client Hello
Handshake Type: Client Hello (1)
Length: 231
Version: TLS 1.2 (0x0303)
Random: 8eed70127f7356cc83f0a406aa6b0b04fc33a7f2656d17..
Session ID Length: 32
Session ID: 37c67023de0dbf60c5afa8bae36b0f469fa510e8130556..
Cipher Suites Length: 0
+ Cipher Suites (4 suites)
+ Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
+ Cipher Suite: TLS_CHACHA20_POLY1305_SHA384 (0x1303)

- The tester then shall stop the testbed and Wireshark.
- The tester shall generate a fake certificate with a newly generated private key using following command:
- ***openssl req -x509 -newkey rsa:4096 -keyout key.pem -out cert.pem -sha256 -days 3650 -nodes -subj "/C=XX/ST=StateName/L=CityName/O=CompanyName/OU=CompanySectionName/CN=CommonNameOrHostname"***
- The tester then shall replace the original certificate (location to be provided by vendor document) with the newly generated fake certificate.
- After replacing the certificate, the tester shall relaunch the testbed and Wireshark and start capturing packets with appropriate filter after running the same curl command.
- The tester should verify that no messages are exchanged after the TLS handshake fails.

Time	Source	Destination	Protocol	Length	Info
23 0.024217	172.18.0.5	172.18.0.4	TLSv1.3	306	Client Hello
25 0.024395	172.18.0.4	172.18.0.5	TLSv1.3	165	Hello Retry Request, Change Cipher Spec
27 0.024539	172.18.0.5	172.18.0.4	TLSv1.3	345	Change Cipher Spec, Client Hello
29 0.025090	172.18.0.4	172.18.0.5	TLSv1.3	1601	Server Hello, Application Data, Application Data, Application Data, Application Data
31 0.026144	172.18.0.5	172.18.0.4	TLSv1.3	73	Alert (Level: Fatal, Description: Unknown CA)

- The tester shall try to contact DUT without using TLS i.e over HTTP and verify that connection is not getting established.

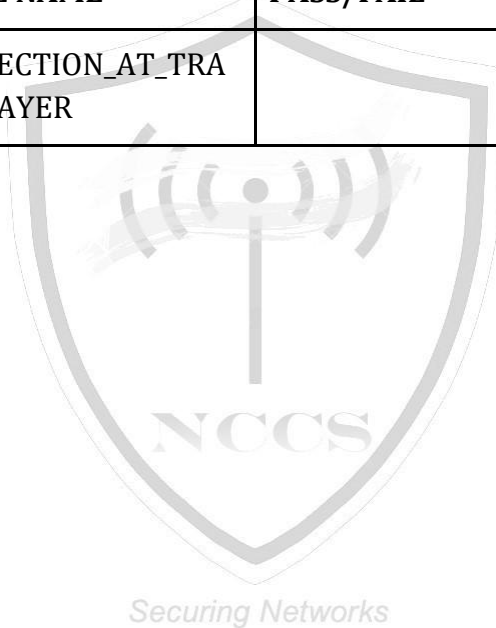
d. Test Observations:

- (Case 1) If DUT and Peer NF are able to mutually authenticate each other over TLS, further communication takes place.
- (Case 2) If DUT and Peer NF are not able to mutually authenticate each other over TLS, further communication does not take place.
- (Case 3) If DUT and Peer NF are not able establish a connection with each other, further communication does not take place.

e. Evidence Provided: Screenshot of pcap file and the pcap file should be shared.

12. **Test Case Result:**

Sl. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC1_PROTECTION_AT_TRANSPORT_LAYER		



2.12.5 TSTP for Evaluation of Authorization token verification failure handling within one PLMN

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.12.5
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >
 - 1) <ITSAR Section No & Name> Section 12: General SBA/SBI Aspects
 - 2) <Security Requirement No & Name > 2.12.5 Authorization token verification failure handling within one PLMN
 - 3) <Requirement Description: >The NF Service producer shall verify the access token as follows:
 - The NF Service producer ensures the integrity of the access token by verifying the signature using NRF's public key or checking the MAC value using the shared secret. If integrity check is successful, the NF Service producer shall verify the claims in the access token as follows: - It checks that the audience claim in the access token matches its own identity or the type of NF service producer. If a list of NSSAIs or list of NSI IDs is present, the NF service producer shall check that it serves the corresponding slice(s).
 - If an NF Set ID is present, the NF Service Producer shall check the NF Set ID in the claim matches its own NF Set ID.
 - If the access token contains "additional scope" information (i.e. allowed resources and allowed actions (service operations) on the resources), it checks that the additional scope matches the requested service operation.
 - If scope is present, it checks that the scope matches the requested service operation.
 - It checks that the access token has not expired by verifying the expiration time in the access token against the current data/time.If the verification is successful, the NF Service producer shall execute the requested service and respond back to the NF Service consumer. Otherwise, it shall reply based on the Oauth 2.0 error response defined in RFC 6749. The NF

service consumer may store the received token(s). Stored tokens may be re-used for accessing service(s) from producer NF type listed in claims (scope, audience) during their validity time.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.2.2.3.1]

Note: This may not be applicable for UPF and SEPP.

4) DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No

- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

- Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) DUT Configuration:

- To get the hash of configuration file if the file is a ASCII text file
Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf
```

- To get the hash of OS if using docker
Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6) Preconditions

- Test environment with a NF service consumer.
- The NF service consumer may be simulated.
- The network product under test has already mutually authenticated with the NF service consumer.
- The tester shall have access to the interface between the NF service consumer and the network product under test.
- The tester has the NRF's private key or the shared key.
- The network product under test is preconfigured with the NRF's public key or the shared key.

- 7) **Test Objective:** To Verify that the NF service producer does not grant service access if the verification of authorization token from a NF service consumer in the same PLMN fails.

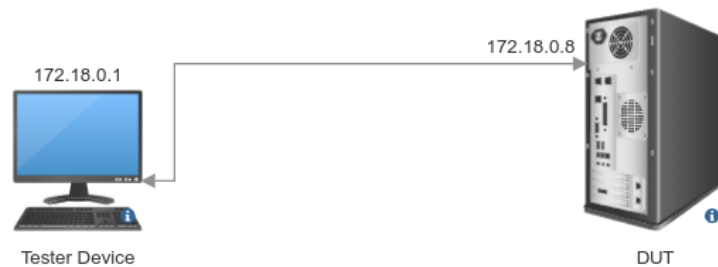
8) Test Plan:

8.1. Number of Test Scenarios:

8.1.1. Test Scenario for HTTP

- Check traffic is properly protected
- Check TLS is being used or not

8.2. Test Setup Diagram:



8.3 Tools Used: Wireshark in Tester Device

8.4 Test Execution Steps

- The network product under test receives the access token sent from the NF service consumer, verifies the access token based on OAuth 2.0.
- Test Case 1: Verification failure of the access token integrity
 - The tester computes an access token correctly, except that the signature or the MAC is incorrect, e.g., the signature or the MAC is randomly selected, and then includes the access token in the NF Service Request sent from the NF service consumer to the network product under test.
 - The integrity verification of the access token by the network product under test fails.
- Test Case 2: Incorrect audience claim in the access token
 - The tester computes an access token correctly, except that the audience claim is incorrect, i.e., the audience claim in the access token does not match the identity or the type of the network product under test, and then includes the access token in the NF Service Request sent from NF service consumer to the network product under test.
 - The network product under test verifies that the audience claim in the access token does not match its identity or type.
- Test Case 3: Incorrect scope claim in the access token
 - The tester computes an access token correctly, except that the scope is incorrect, i.e., the scope does not match the requested service operation, and then includes the access token in the NF Service Request sent from the NF service consumer to the network product under test.

- The network product under test verifies that the integrity verification of the access token and audience claim verification are correct. However, the scope does not match the requested service operation.
- Test Case 4: Expired access token
- The tester computes an access token correctly, except that the expiration time has expired against the current data/time, and then includes the access token in the NF Service Request sent from the NF service consumer to the network product under test.
- The network product under test verifies that the expiration time in the access token has expired against the current data/time.

9) **Expected Results for Pass:** For test cases 1~4, the network product under test rejects the NF service consumer's service request based on OAuth 2.0 error response defined in RFC 6749

10) **Expected Format of Evidence:** Evidence suitable for the interface, e.g., Screenshot containing the operational results.

11) **Test Execution:**

➤ **Test Case Number:** 01

a) **Test Case Name:** TC_AUTHORIZATION_TOKEN_VERIFICATION_FAILURE_ONE_PLMN

b) **Test Case Description:**

c) **Execution Steps:**

- The network product under test receives the access token sent from the NF service consumer, verifies the access token based on OAuth 2.0.
- Test Case 1: Verification failure of the access token integrity
- The tester computes an access token correctly, except that the signature or the MAC is incorrect, e.g., the signature or the MAC is randomly selected, and then includes the access token in the NF Service Request sent from the NF service consumer to the network product under test.
- The integrity verification of the access token by the network product under test fails.

```
Generated JWT token: b'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjoxMjM0LCJyb2xlcYI6WyJhZG1pbGlzInRlc3RlcjJdLCJleHAiOiJlE2OTA3ODQxOTl9.rjDjLNIQOEyN0k75rh0ExiqEFHR3TX5w8evANeR5qKQ'
Tampered JWT token: b'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjoxMjM0LCJyb2xlcYI6WyJhZG1pbGlzInRlc3RlcjJdLCJleHAiOiJlE2OTA3ODQxOTl9.GCglYlokPTMuMbFKHR60fSJKY5cxhQMrZNSogXDnTHG31kozMESUAYnxsvGyDV0y'

Token validation failed: Signature verification failed
Tampered token is rejected according to OAuth 2.0 standards.
Error response: {'error': 'invalid_token', 'error_description': 'The access token is invalid.'}
```

- Test Case 2: Incorrect audience claim in the access token
- The tester computes an access token correctly, except that the audience claim is incorrect, i.e., the audience claim in the access token does not match the identity or the type of the network product under test, and then includes the access token in the NF Service Request sent from NF service consumer to the network product under test.

- The network product under test verifies that the audience claim in the access token does not match its identity or type.

```
Generated JWT token: b'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjoxMjM0LCJyb2xlcYI6WyJhZG1pbI
IsInRlc3RlcjJdLCJleHAiOiJlE2OTA3ODQyMjksImF1ZCI6ImLuY29ycmVjdF9hdWRpZW5jZSJ9.vt6Tuz5_3f7nq0XvpZj5PNwZA2N
Nj05tfQh-1oo8vbc'
Token validation failed: Invalid audience
Token is rejected according to OAuth 2.0 standards due to the incorrect audience claim.
Error response: {'error': 'invalid_token', 'error_description': 'The access token is invalid.'}
```

- Test Case 3: Incorrect scope claim in the access token
- The tester computes an access token correctly, except that the scope is incorrect, i.e., the scope does not match the requested service operation, and then includes the access token in the NF Service Request sent from the NF service consumer to the network product under test.
- The network product under test verifies that the integrity verification of the access token and audience claim verification are correct. However, the scope does not match the requested service operation.

```
Generated JWT token: b'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjoxMjM0LCJyb2xlcYI6WyJhZG1pbI
IsInRlc3RlcjJdLCJleHAiOiJlE2OTA3ODQyMjksImF1ZCI6ImLuY29ycmVjdF9hZDZlE29wZTEgaw5jb3JyZW50X3Njb3B1MiJ9.I9dJlK0Dck08EK6peM2WW_ZF4fQ
Im-SyLk0CpT3y73M'
Token is rejected according to OAuth 2.0 standards due to the incorrect scope claim.
Error response: {'error': 'insufficient_scope', 'error_description': 'The access token does not contain the required scope 'required_scope'.'}
```

- Test Case 4: Expired access token
- The tester computes an access token correctly, except that the expiration time has expired against the current data/time, and then includes the access token in the NF Service Request sent from the NF service consumer to the network product under test.
- The network product under test verifies that the expiration time in the access token has expired against the current data/time.

```
Generated JWT token: b'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjoxMjM0LCJyb2xlcYI6WyJhZG1pbI
IsInRlc3RlcjJdLCJleHAiOiJlE2OTA3NzcwNzI9.s_ZGk6bfskcl9Mo07q5-qmYsmj0XaXNB3636iZT5Mc4'
Token is rejected according to OAuth 2.0 standards due to expiration.
Error response: {'error': 'invalid_token', 'error_description': 'The access token has expired.'}
```

f. Test Observations:

- For test cases 1~4, the network product under test should reject the NF service consumer's service request based on OAuth 2.0 error response defined in RFC 6749

12) Test Case Result:

Sl. No	TEST CASE NAME	PASS/FAIL	Remarks
1	TC_ AUTHORIZATION_TOKEN_VERIFIC ATION_FAILURE_ONE_PLMN		

2.12.6 TSTP for Evaluation of Authorization token verification failure handling in different PLMNs

Session Management function ITSAR	ITSAR No: ITSAR111092401	Clause no:2.12.6
-----------------------------------	--------------------------	------------------

Note: The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
 - <DUT Software Version:>
 - <Digest Hash of OS>
 - <Digest Hash of Configuration>
 - <Applicable ITSAR: >
 - <ITSAR Version No:>
 - <OEM Supplied Document list: >
- 1) **<ITSAR Section No & Name>** Section 12: General SBA/SBI Aspects
 - 2) **<Security Requirement No & Name >** 2.12.6 Authorization token verification failure handling in different PLMNs
 - 3) **<Requirement Description: >** The NF service producer shall check that the home PLMN ID of the audience claimed in the access token matches its own PLMN identity.

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.2.2.3.2]

Note: This may be applicable for SEPP.

Securing Networks

4) **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No

- Command used: **ifconfig -a** (To find IP information and all interfaces)


```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
    inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
    ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
    RX packets 180276 bytes 177898049 (169.6 MiB)
    RX errors 0 dropped 314 overruns 0 frame 0
    TX packets 57117 bytes 10085270 (9.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111 bytes 10484 (10.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111 bytes 10484 (10.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $ ./amf.out --version
IIT_amf 6.9
```

- Command used : **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

	File: /etc/os-release
1	NAME="Ubuntu"
2	VERSION="20.04.5 LTS (Focal Fossa)"
3	ID=ubuntu
4	ID_LIKE=debian
5	PRETTY_NAME="Ubuntu 20.04.5 LTS"
6	VERSION_ID="20.04"
7	HOME_URL="https://www.ubuntu.com/"
8	SUPPORT_URL="https://help.ubuntu.com/"
9	BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10	PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11	VERSION_CODENAME=focal
12	UBUNTU_CODENAME=focal

5) **DUT Configuration:**

- To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf
```

- To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6) **Preconditions**

- Test environment with a NF service consumer and two SEPPs (one cSEPP, one pSEPP).
- The NF service consumer and SEPPs may be simulated.
- The network product under test has already mutually authenticated with the NF service consumer in a different PLMN via the SEPPs.
- The tester has the NRF's private key or the shared key.
- The network product under test is preconfigured with the NRF's public key or the shared key.
- The tester shall have access to the interfaces of the NF service consumer and the network product under test.

7) **Test Objective:** Verify that the NF service producer does not grant service access if the verification of authorization token from a NF service consumer in a different PLMN fails.

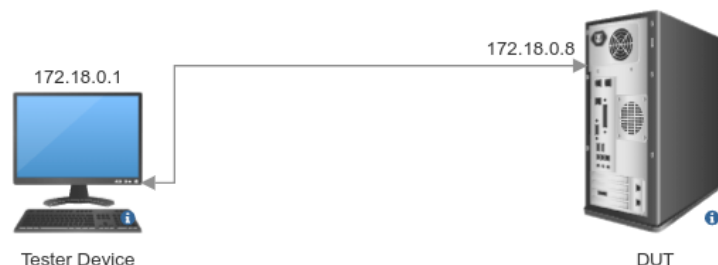
8) **Test Plan:**

8.1. **Number of Test Scenarios:**

8.1.1. Test Scenario for HTTP

- Check traffic is properly protected
- Check TLS is being used or not

8.2. **Test Setup Diagram:**



8.3 **Tools Used:** Wireshark in Tester Device

8.4 **Test Execution Steps**

- The network product under test receives the access token sent from the NF service consumer, verifies the access token in accordance with the execution steps in TSTP 2.12.5, with the following additional test cases:
- Test Case 1: incorrect PLMN ID of the NF service producer in the access token
 - The test computes an access token correctly, except that the PLMN ID in the producerPlmnId claim of the access token is empty or different from the home PLMN ID of the network product under test, and then includes the access token in the NF Service Request sent from the NF service consumer to the network product under test through the SEPPs.
 - The network product under test receives the access token sent from the NF service consumer through the SEPPs, verifies that the PLMN ID in the producerPlmnId claim of the access token is different from its own home PLMN identity.
- Test Case 2: absent PLMN ID of the NF service producer in the access token
 - The test computes an access token correctly, except that no producerPlmnId claim is included in the access token, and then includes the access token in the NF Service Request sent from the NF service consumer to the network product under test through the SEPPs.
 - The network product under test receives the access token sent from the NF service consumer through the SEPPs, verifies that the access token is not a token to be used by the NF service consumer in a different PLMN, based on the absence of PLMN ID of the NF service producer in the access token.

9) **Expected Results for Pass:** For both test cases 1 and 2, the network product under test rejects the NF service consumer's service request based on OAuth 2.0 error response defined in RFC 6749

10) **Expected Format of Evidence:** Evidence suitable for the interface, e.g., Screenshot containing the operational results.

11) **Test Execution:**

➤ **Test Case Number:** 01

a) **Test Case Name:** TC_AUTHORIZATION_TOKEN_VERIFICATION_FAILURE_ONE_PLMN

b) **Test Case Description:** Verify that the NF service producer does not grant service access if the verification of authorization token from a NF service consumer in a different PLMN fails.

c) **Execution Steps:**

- The network product under test receives the access token sent from the NF service consumer, verifies the access token in accordance with the execution steps in TSTP 2.12.5, with the following additional test cases:
- Test Case 1: incorrect PLMN ID of the NF service producer in the access token

- ```
jamesnaan@jamesnaan-Super-Server:~/Desktop/tstps_sarvanan/PLMN_handling$ python3 incorrect_plmn_cla
im.py
Generated JWT token: b'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjoxoxMjM0LCJyb2xlcYI6WyJhZG1
pbIiIsInRlc3RlcjJdLCJwcm9kdWNlClBsbW5JZCI6ImluY29ycmVjdF9wbG1uX2lkIn0.DF81HtIaAQ4ZPTgyKfLh521-PHPTD6
QY11VmodW9ofk'
PLMN ID claim is incorrect.
Error response: {'error': 'invalid_request', 'error_description': "The PLMN ID in the producerPlmnI
d claim is different from the network product's home PLMN ID."}
```

- Test Case 2: absent PLMN ID of the NF service producer in the access token
  - The test computes an access token correctly, except that no producerPlmnId claim is included in the access token, and then includes the access token in the NF Service Request sent from the NF service consumer to the network product under test through the SEPPs.
  - The network product under test receives the access token sent from the NF service consumer through the SEPPs, verifies that the access token is not a token to be used by the NF service consumer in a different PLMN, based on the absence of PLMN ID of the NF service producer in the access token.

```
jamesnaan@jamesnaan-Super-Server:~/Desktop/tstps_sarvanaan/PLMN_handling$ python3 absent_plmn_claim.py
Generated JWT token: b'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjoxMjM0LCJyb2xlcyciOi6WjZhZG1pbiIsInRlc3RlcjJdfQ.Yu5ZpJ2BG7B1y07iBWw0WYR59Zi2ETyqzk5F13nqj5k'
Producer PLMN ID claim is absent.
Error response: {'error': 'invalid_request', 'error_description': 'The producerPlmnId claim is absent in the access token.'}
```

- For test cases 1~4 of 2.12.5 and 1~2 of current TSTP, the network product under test should reject the NF service consumer's service request based on Oauth 2.0 error response defined in RFC 6749

| Sl. No | TEST CASE NAME                                                     | PASS/FAIL | Remarks |
|--------|--------------------------------------------------------------------|-----------|---------|
| 1      | TC_<br>AUTHORIZATION_TOKEN_VERIFICA<br>TION FAILURE DIFFERENT PLMN |           |         |

---

## 2.13.1 TSTP For Remote Diagnostic Procedure - Verification under 5G ITSAR

---

|                                   |                          |                  |
|-----------------------------------|--------------------------|------------------|
| Session Management function ITSAR | ITSAR No: ITSAR111092401 | Clause no:2.13.1 |
|-----------------------------------|--------------------------|------------------|

**Note:** The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 13 – Other Security Requirement
2. **<Security Requirement No & Name >** 2.13.1 Remote Diagnostic Procedure – Verification
3. **<Requirement Description: >** If the SMF is providing Remote access for troubleshooting purposes/alarm maintenance then it shall be allowed only for authorized users, other than the root user. All activities performed by the remote user are to be logged with the following parameters:

1. User id
2. Time stamp
3. Interface type
4. Event level (e.g. CRITICAL, MAJOR, MINOR)
5. Command/activity performed
6. Result type (e.g. SUCCESS, FAILURE).
7. IP Address of remote machine



#### 4. **DUT Confirmation Details:**

- Use the command line interface to get details of the machine on which test is conducted.
  - Use command to get IP and Interfaces details
  - Use command to get Application No/Version
  - Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
 inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
 inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
 ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
 RX packets 180276 bytes 177898049 (169.6 MiB)
 RX errors 0 dropped 314 overruns 0 frame 0
 TX packets 57117 bytes 10085270 (9.6 MiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
 device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 inet6 ::1 prefixlen 128 scopeid 0x10<host>
 loop txqueuelen 1000 (Local Loopback)
 RX packets 111 bytes 10484 (10.2 KiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 111 bytes 10484 (10.2 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

|    | File: /etc/os-release                                                               |
|----|-------------------------------------------------------------------------------------|
| 1  | NAME="Ubuntu"                                                                       |
| 2  | VERSION="20.04.5 LTS (Focal Fossa)"                                                 |
| 3  | ID=ubuntu                                                                           |
| 4  | ID_LIKE=debian                                                                      |
| 5  | PRETTY_NAME="Ubuntu 20.04.5 LTS"                                                    |
| 6  | VERSION_ID="20.04"                                                                  |
| 7  | HOME_URL="https://www.ubuntu.com/"                                                  |
| 8  | SUPPORT_URL="https://help.ubuntu.com/"                                              |
| 9  | BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"                                 |
| 10 | PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy" |
| 11 | VERSION_CODENAME=focal                                                              |
| 12 | UBUNTU_CODENAME=focal                                                               |

## 5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF\_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)



```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

For **Linux and SSH:**

command used: **ssh -V** (To get version information)

```
amf@localhost $ ssh -V
OpenSSH_8.9p1 Ubuntu-3ubuntu0.1, OpenSSL 3.0.2 15 Mar 2022
```

Locate the line that starts with **PermitRootLogin** in the SSH server configuration file. By default, it is set to yes, allowing root login.

Make sure this field is set to No.

Ensure **Allow Users** is set to authorized user account(s) who are allowed remote access.

Here, says remote user is the name of authorized user.

```
Include /etc/ssh/sshd_config.d/*.conf

#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

Ciphers and keying
#RekeyLimit default none

Logging
#SyslogFacility AUTH
#LogLevel INFO

Authentication:

#LoginGraceTime 2m
PermitRootLogin no
AllowUsers remoteuser

#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

Similarly for other remote connection protocols.

For **log file that contains the remote information:** Determine the log file that contains the remote login information. Commonly used log files include **/var/log/auth.log, /var/log/secure, or /var/log/messages.**

The exact location and name of the log file can differ depending on the Linux distribution and the system's configuration.

**For checking if activities performed by the remote user are to be logged:**

**Check** if rsyslog is installed, If rsyslog is installed, the command will show the path to the rsyslogd executable. If it's not installed, the command will show no output.

**Command Used: *which rsyslogd***

```
amf@localhost $ which rsyslogd
/usr/sbin/rsyslogd
```

Go to the rsyslog conf file, and check the path of the log file which will log the commands.

**Command used: *cat /etc/rsyslog.d/bash.conf***

```
amf@localhost $ cat /etc/rsyslog.d/bash.conf
local6.* /var/log/commands.log
```

Based on above output, commands performed by remote user will be stored in the /var/log/command.log

6. **Preconditions:** A document that describes the interfaces to the network product and how the tester can login to them remotely.

7. **Test Objective/ Purpose:** To ensure secure access and proper logging of activities performed by remote users.

8. **Test Plan:**

**8.1 Number of Test Scenarios:**

**8.1.1 Test Scenario for Authorized User Remote Login:** This test scenario checks Remote access to the SMF shall be granted only to authorized users.

**8.1.2 Test Scenario for Secure Logging:** This test scenario checks whether proper logging is done for activities performed by remote users.

**8.2 Testbed Diagram:**



**8.3 Tools required:** Command Line Interface of the DUT, ssh, snmp or other remote access protocols.

**8.4 Test Execution Step:**

1. Log in remotely using the authorized as well as unauthorized user's credentials.
2. Once Logged in remotely with an authorized user, perform a test activity or command on the system.
3. Verify that the user ID, time stamp, interface type, event level, command/activity performed, result type, and remote machine IP address are properly logged.

9. **Expected Results:**

- The authorized user is able to login remotely.
- The logs of activities performed by remote users are maintained.

10. **Expected Format of Evidence:** Evidence suitable for the interface, e.g. screenshot containing the operational results.

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC1\_REMOTE\_DIAGNOSTIC\_PROCEDURE

b. **Test Case Description:** Test needs to be conducted to ensure Remote access to the Network Product shall be granted only to authorized users.

c. **Execution Steps: Note:** There can be many ways through which remote access can be performed like ssh, snmp, telnet, VNC (Virtual Network Computing), RDP (Remote Desktop Protocol) etc. Below steps are for ssh.

Log in remotely with ssh using the authorized user's credentials.

Use below command,

***ssh root@<network\_product\_ip>***

d. **Test Observation:**

➤ **Case 1:** Authorized user able to login (**Positive Testcase**) If the authorized user is able to login remotely successfully, then this test case passes.

➤ **Case 2:** Unauthorized user able to login (**Negative Testcase**) If an unauthorized user is able to login remotely successfully, then this test case fails.

➤ **Test Case Number: 02**

a. **Test Case Name:** TC2\_REMOTE\_DIAGNOSTIC\_PROCEDURE

b. **Test Case Description:** Test needs to be conducted to ensure that activities performed by remote users as well as its details are logged in properly.

c. **Execution Steps:**

- Login remotely via console using the authorized user credentials.
- Go to the /var/log/auth.log file, to check the details about remote users such as uid, ip address of remote machine, time stamps etc.
- Go to the /var/log/commands.log file.
- Check whether the logs for commands performed by remote user have been captured or not.

#### d. Test Observation:

- **Case 1: Logs of Remote Session captured successfully (Positive TestCase)** The below screenshot is of **auth.log** file, which shows the details of uid, ip address of remote user, timestamp etc.

```
amf@localhost $ cat /var/log/auth.log
Jul 28 16:46:23 priyansha sudo: pam_unix(sudo:session): session closed for user root
Jul 28 16:47:14 priyansha sshd[2331094]: Accepted password for priyansha from 192.168.127.177 port 55274 ssh2
Jul 28 16:47:14 priyansha sshd[2331094]: pam_unix(sshd:session): session opened for user priyansha by (uid=0)
Jul 28 16:47:14 priyansha systemd-logind[1147]: New session 340 of user priyansha.
Jul 28 16:47:23 priyansha sshd[2331148]: Received disconnect from 192.168.127.177 port 55274:11: disconnected by user
Jul 28 16:47:23 priyansha sshd[2331148]: Disconnected from user priyansha 192.168.127.177 port 55274
Jul 28 16:47:23 priyansha sshd[2331094]: pam_unix(sshd:session): session closed for user priyansha
Jul 28 16:47:23 priyansha systemd-logind[1147]: Session 340 logged out. Waiting for processes to exit.
Jul 28 16:47:23 priyansha systemd-logind[1147]: Removed session 340.
Jul 28 16:48:55 priyansha sudo: priyansha : TTY=pts/11 ; PWD=/ ; USER=root ; COMMAND=/usr/bin/nano /var/log/auth.log
Jul 28 16:48:55 priyansha sudo: pam_unix(sudo:session): session opened for user root by (uid=0)
Jul 28 16:49:40 priyansha sudo: pam_unix(sudo:session): session closed for user root
```

The below screenshot is of **commands.log** file which shows that logs containing the information about the remote user have been captured successfully.

Here, we can see that the commands performed by remote users are captured,

```
amf@localhost $ cat /var/log/commands.log
Jul 28 15:53:33 priyansha priyansha: priyansha@ [2108469]: sudo systemctl restart rsyslog.service [0]
Jul 28 15:53:36 priyansha priyansha: priyansha@ [2108469]: cat /var/log/commands.log [0]
Jul 28 15:53:52 priyansha priyansha: priyansha@192.168.127.177 [2166905]: exit [0]
Jul 28 15:54:36 priyansha priyansha: priyansha@192.168.127.177 [2166905]: cd Desktop/TSTP [0]
Jul 28 15:54:44 priyansha priyansha: priyansha@192.168.127.177 [2166905]: ls [0]
Jul 28 15:54:50 priyansha priyansha: priyansha@192.168.127.177 [2166905]: cat app.txt [0]
Jul 28 15:55:06 priyansha priyansha: priyansha@ [2170666]: exit [0]
Jul 28 15:55:11 priyansha priyansha: priyansha@ [2170666]: cat app.txt [0]
```

#### e. Evidence Provided: A testing report which will consist of the following information:

- Screenshot of the output of the terminal of the PC through which DUT logs in.

#### 12. Test Case Result:

| Sl. No | TEST CASE NAME                                                     | PASS/FAIL | Remarks |
|--------|--------------------------------------------------------------------|-----------|---------|
| 1      | TC1_REMOTE_DIAGNOSTIC_PROCEDURE<br>TC2_REMOTE_DIAGNOSTIC_PROCEDURE |           |         |

---

## 2.13.2 TSTL Report for No System Password Recovery

---

|                                   |                          |                  |
|-----------------------------------|--------------------------|------------------|
| Session Management function ITSAR | ITSAR No: ITSAR111092401 | Clause no:2.13.2 |
|-----------------------------------|--------------------------|------------------|

**Note:** The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
  - <DUT Software Version:>
  - <Digest Hash of OS>
  - <Digest Hash of Configuration>
  - <Applicable ITSAR: >
  - <ITSAR Version No:>
  - <OEM Supplied Document list: >
1. **<ITSAR Section No & Name>** Section 13: Other Security requirements
  2. **<Security Requirement No & Name >** 2.13.2 No System Password Recovery
  3. **<Requirement Description: >** No provision shall exist for SMF System / Root password recovery.
  4. **DUT Confirmation Details:**  
Use the command line interface to get details of the machine on which test is conducted.
    - Use command to get IP and Interfaces details
    - Use command to get Application No/Version
    - Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
 inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
 inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
 ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
 RX packets 180276 bytes 177898049 (169.6 MiB)
 RX errors 0 dropped 314 overruns 0 frame 0
 TX packets 57117 bytes 10085270 (9.6 MiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
 device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 inet6 ::1 prefixlen 128 scopeid 0x10<host>
 loop txqueuelen 1000 (Local Loopback)
 RX packets 111 bytes 10484 (10.2 KiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 111 bytes 10484 (10.2 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)  
Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $./amf.out --version
IIT_amf 6.9
```

- Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

|    | File: /etc/os-release                                                               |
|----|-------------------------------------------------------------------------------------|
| 1  | NAME="Ubuntu"                                                                       |
| 2  | VERSION="20.04.5 LTS (Focal Fossa)"                                                 |
| 3  | ID=ubuntu                                                                           |
| 4  | ID_LIKE=debian                                                                      |
| 5  | PRETTY_NAME="Ubuntu 20.04.5 LTS"                                                    |
| 6  | VERSION_ID="20.04"                                                                  |
| 7  | HOME_URL="https://www.ubuntu.com/"                                                  |
| 8  | SUPPORT_URL="https://help.ubuntu.com/"                                              |
| 9  | BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"                                 |
| 10 | PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy" |
| 11 | VERSION_CODENAME=focal                                                              |
| 12 | UBUNTU_CODENAME=focal                                                               |

## 5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF\_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker



Command used: ***docker images --digests*** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

Check that the recovery mode is disabled in the operating system. In Linux check the file /etc/default/grub.

```
22 # The resolution used on graphical terminal
23 # note that you can use only modes which your graphic card supports via VBE
24 # you can see them in real GRUB with the command 'vbeinfo'
25 #GRUB_GFXMODE=640x480
26
27 # Uncomment if you don't want GRUB to pass "root=UUID=xxx" parameter to Linux
28 #GRUB_DISABLE_LINUX_UUID=true
29
30 # Uncomment to disable generation of recovery mode menu entries
31 GRUB_DISABLE_RECOVERY="true"
32
33 # Uncomment to get a beep at grub start
34 #GRUB_TNTT_TUNE="480 440 1"
```

## 6. **Preconditions:**

- Tester should have Sudo permissions.
- If Linux is not used, OEM should provide where to check that the recovery mode changes is disabled.

7. **Test Objective:** To check that any user with root credentials should not be able to reset password for the root/system accounts.

## 8. **Test Plan:**

### 8.1 **Number of Test Scenarios: 1**

8.1.1 Test scenario when the password recovery is unset in the /etc/default/grub file (for Linux).

8.1.2 Any other configuration file based on the OS being used. Test whether able to recover passwords for system/root.

### 8.2 Test bed Diagram:



8.3 **Tools Used:** Command line

### 8.4 **Test Execution Steps:**

- Check that the password recovery for grub is turned off.

- Try changing password by entering recovery mode.
9. **Expected Results for Pass:** Tester should not be able to recover password.
10. **Expected Format of Evidence:** Screenshots of the screen showing the inability to recover password.
11. **Test Execution:**
- Test Case Number:** 01
- a. Test Case Name:** TC\_NO\_PASS\_RECOVERY
- b. Test Case Description:** To check that any user with root credentials should not be able to reset password for the root/system accounts.
- c. Execution Steps:**
- Restart the system and select the Advanced options for GNU/Linux” option by pressing the down arrow key and Enter button.



```
GNU GRUB version 2.02

Ubuntu
*Advanced options for Ubuntu
Memory test (memtest86+)
Memory test (memtest86+, serial console 115200)
```

- Press 'e' to edit the commands. System should ask the grub password.



```
Enter username:
student
Enter password:
-
```

- If it asks for password, then the grub is safe, and password cannot be recovered by any user who does not know the grub password.
- If the user can change the following command, mark this requirement as failed.

```
GNU GRUB version 2.02

insmod part_msdos
insmod ext2
set root='hd0,msdos1'
if [x$feature_platform_search_hint = xy]; then
 search --no-floppy --fs-uuid --set=root --hint-bios=hd\
0,msdos1 --hint-efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 829d3cd6-a\
cc6-4c16-a3b8-ad414ba76759
else
 search --no-floppy --fs-uuid --set=root 829d3cd6-acc6-\
4c16-a3b8-ad414ba76759
fi
echo 'Loading Linux 4.15.0-33-generic ...'
linux /boot/vmlinuz-4.15.0-33-generic root=UUID=829d3cd6\
-acc6-4c16-a3b8-ad414ba76759 rw quiet splash $vt_handoff init=/bin/bash
echo 'Loading initial ramdisk ...'
```

**d. Test Observations:**

- Case 1: Tester is not able to recover password by any means. Tester device is pass.
- Case 2: Tester device can recover the password for system/root and the device fails this requirement.

**12. Test Case Result:**

| Sl. No | TEST CASE NAME      | PASS/FAIL | Remarks |
|--------|---------------------|-----------|---------|
| 1      | TC_NO_PASS_RECOVERY |           |         |

---

### 2.13.3 TSTL Report for Secure System Software Revocation

---

|                                   |                          |                   |
|-----------------------------------|--------------------------|-------------------|
| Session Management function ITSAR | ITSAR No: ITSAR111092401 | Clause no:2.13.13 |
|-----------------------------------|--------------------------|-------------------|

**Note:** The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 13: Other Security requirements
2. **<Security Requirement No & Name >** 2.13.3 Secure System Software Revocation
3. **<Requirement Description: >** Once the SMF software image is legally updated/upgraded with New Software Image, it shall normally not be possible to roll back to a previous software image. In case roll back is essential, it shall be done by the administrator with appropriate non-repudiation controls. SMF shall support a well-established control mechanism for rolling back to previous software image.

#### 4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
 inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
 inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
 ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
 RX packets 180276 bytes 177898049 (169.6 MiB)
 RX errors 0 dropped 314 overruns 0 frame 0
 TX packets 57117 bytes 10085270 (9.6 MiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
 device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 inet6 ::1 prefixlen 128 scopeid 0x10<host>
 loop txqueuelen 1000 (Local Loopback)
 RX packets 111 bytes 10484 (10.2 KiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 111 bytes 10484 (10.2 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)  
Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)

```
amf@localhost $./amf.out --version
IIT_amf 6.9
```

- Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

|    | File: /etc/os-release                                                               |
|----|-------------------------------------------------------------------------------------|
| 1  | NAME="Ubuntu"                                                                       |
| 2  | VERSION="20.04.5 LTS (Focal Fossa)"                                                 |
| 3  | ID=ubuntu                                                                           |
| 4  | ID_LIKE=debian                                                                      |
| 5  | PRETTY_NAME="Ubuntu 20.04.5 LTS"                                                    |
| 6  | VERSION_ID="20.04"                                                                  |
| 7  | HOME_URL="https://www.ubuntu.com/"                                                  |
| 8  | SUPPORT_URL="https://help.ubuntu.com/"                                              |
| 9  | BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"                                 |
| 10 | PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy" |
| 11 | VERSION_CODENAME=focal                                                              |
| 12 | UBUNTU_CODENAME=focal                                                               |

## 5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF\_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2ede1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

#### 6. **Preconditions:**

- The system must have a pre-existing software version installed and running. This version will serve as the baseline for future updates or upgrades.
- The system must have a well-established control mechanism in place to manage software image updates, upgrades, and rollbacks.

7. **Test Objective:** To verify that once the system software image is updated/upgraded, it cannot be rolled back without proper authorization, and rollbacks are logged and audited with non-repudiation controls.

#### 8. **Test Plan:**

8.1 **Number of Test Scenarios: 1:** Test to check that once the system software image is updated/upgraded, it cannot be rolled back without proper authorization, and rollbacks are logged and audited with non-repudiation controls.

#### 8.2 **Test bed Diagram:**



8.3 Tools Used: Command line

8.4 Test Execution Steps: - Initial Software Image Version:

- a. Record the current version of the system software image running on the system.
- b. Ensure that this version is well-documented and acknowledged by the administrator.
  - Software Image Update/Upgrade:



- a. Update or upgrade the system software image with a new version through an authorized and well-documented process.
- b. Verify that the new software image is successfully installed and running on the system.

- **Rollback Attempt - Unauthorized User:**

- a. Attempt to roll back the system software image to the previous version without proper authorization as an unauthorized user.
- b. Verify that the system prevents the rollback attempt and generates appropriate error messages or logs.

- **Rollback Attempt - Administrator:**

- a. As an administrator with proper privileges, attempt to roll back the system software image to the previous version.
- b. Verify that the system allows the rollback only for authorized administrators and prompts for appropriate credentials.

9. **Expected Results for Pass:** Non- authorized user should not be able to rollback to any previous versions of the software image.
10. **Expected Format of Evidence:** Versions details of the software image before and after performing the test. It should show that even when the tester tried to roll back the Software image, version remains the same.

11. **Test Execution:**

- **Test Case Number:** 01

- a. **Test Case Name:** TC\_SYSTEM\_SOFTWARE\_REVOCATION

- b. **Test Case Description:** Verify that once the system software image is updated/upgraded, it cannot be rolled back without proper authorization, and rollbacks are logged and audited with non-repudiation controls.

- c. **Execution Steps:**

- **Case 1:** Attempt to roll back the system software image to the previous version without proper authorization as an unauthorized user. Verify that the system prevents the rollback attempt and generates appropriate error messages or logs.

A user with non-administrative rights is denied access -

```
user2@supriya-Super-Server:~$ docker service rollback my-service
Got permission denied while trying to connect to the Docker daemon
/var/run/docker.sock: connect: permission denied
```

- **Case 2:** As an administrator with proper privileges, attempt to roll back the system software image to the previous version. Verify that the system allows the rollback only for authorized administrators and prompts for appropriate credentials.

A user with administrative rights is allowed to rollback -

```
supriya@supriya-Super-Server:~$ docker service rollback my-service
my-service
rollback: manually requested rollback
overall progress: rolling back update: 1 out of 1 tasks
1/1: running [>
verify: Service converged
rollback: rollback completed
```

- d. Test Observations:** The test is considered successful if the system prevents unauthorized rollback attempts, allows rollback only for authorized administrators, generates non-repudiation logs of all rollbacks actions, and complies with the requirement for controlled rollbacks

12. **Test Case Result:**

| Sl. No | TEST CASE NAME                | PASS/FAIL | Remarks |
|--------|-------------------------------|-----------|---------|
| 1      | TC_SYSTEM_SOFTWARE_REVOCATION |           |         |

NCCS  
Securing Networks

---

## 2.13.4 TSTP Software Integrity Check –Installation

---

|                                   |                          |                  |
|-----------------------------------|--------------------------|------------------|
| Session Management function ITSAR | ITSAR No: ITSAR111092401 | Clause no:2.13.4 |
|-----------------------------------|--------------------------|------------------|

**Note:** The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

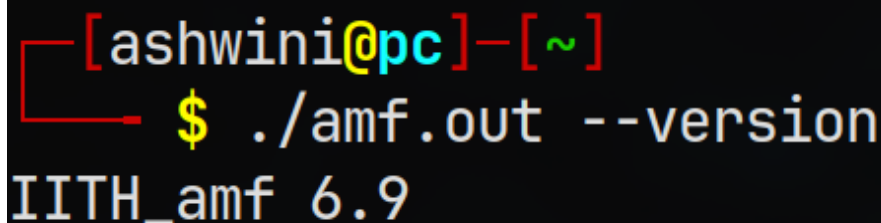
- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 13 - Other Security Requirements
2. **<Security Requirement No & Name >** 2.13.4 Software Integrity Check - Installation
3. **<Requirement Description: >** NF shall validate the software package integrity before the installation/upgrade stage strictly using the Secure cryptographic controls prescribed in Table1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only. Tampered software shall not be executed or installed if integrity check fails.

4. **DUT Confirmation Details:**

- Use command to get Application No/Version
- Use command to get OS Version/No

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)



```
[ashwini@pc]~
$./amf.out --version
IITH_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
[ashwini@pc]~$ cat /etc/os-release
```

|    | File: /etc/os-release                                     |
|----|-----------------------------------------------------------|
| 1  | NAME="Manjaro Linux"                                      |
| 2  | PRETTY_NAME="Manjaro Linux"                               |
| 3  | ID=manjaro                                                |
| 4  | ID_LIKE=arch                                              |
| 5  | BUILD_ID=rolling                                          |
| 6  | ANSI_COLOR="32;1;24;144;200"                              |
| 7  | HOME_URL="https://manjaro.org/"                           |
| 8  | DOCUMENTATION_URL="https://wiki.manjaro.org/"             |
| 9  | SUPPORT_URL="https://forum.manjaro.org/"                  |
| 10 | BUG_REPORT_URL="https://docs.manjaro.org/reporting-bugs/" |
| 11 | PRIVACY_POLICY_URL="https://manjaro.org/privacy-policy/"  |
| 12 | LOGO=manjarolinux                                         |

```
jamesnaan@KS-Laptop:~$ sudo cat /etc/os-release
[sudo] password for jamesnaan:
NAME="Ubuntu"
VERSION="20.04.6 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.6 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
```

## 5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF\_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

For .dsc files with **dpkg**,

Verify that **gpg** is privileged command (To ensure no one but admin can import keys)

Command used: **ls -lrt /bin/gpg**

```
-rwxr--r-- 1 root root 1066992 Jul 4 2022 /bin/gpg
```

Command used: **gpg --version** (To get version information)

```
gpg (GnuPG) 2.2.19
libgcrypt 1.8.5
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: /home/pratik/.gnupg
Supported algorithms:
Pubkey: RSA, ELG, DSA, ECDH, ECDSA, EDDSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
 CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compression: Uncompressed, ZIP, ZLIB, BZIP2
```

Command used: **dpkg --version** (To get version information)

```
Debian 'dpkg' package management program version 1.19.7 (amd64).
This is free software; see the GNU General Public License version 2 or
later for copying conditions. There is NO warranty.
```

Verify that the public key used for signing the .dsc file is present in the system Command used: **gpg --verify <package\_name.dsc>** (Verify the software package)

```
gpg: Signature made Tuesday 26 October 2021 03:16:52 AM IST
gpg: using RSA key 8F8D83B5270649A080648255EAF23F4A0BD34BF0
gpg: issuer "mfo@canonical.com"
gpg: Good signature from "Mauricio Faria de Oliveira <mfo@canonical.com>" [unknown]
gpg: aka "Mauricio Faria de Oliveira <mauricio.oliveira@canonical.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg: There is no indication that the signature belongs to the owner.
Primary key fingerprint: 8F8D 83B5 2706 49A0 8064 8255 EAF2 3F4A 0BD3 4BF0
```

Verify that the hashing and code signing algorithm used for the package are in accordance with “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only.

Command used: **cat <package\_name.dsc>** (Display .dsc file)

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512

Format: 3.0 (quilt)
Source: ufw
Binary: ufw
Architecture: all
Version: 0.36-6ubuntu1
Maintainer: Jamie Strandboge <jamie@ubuntu.co
```

Command used: **gpg --list-keys | grep -B 1 <finger\_print>** (Display the signing algorithm)

```
pub rsa4096 2018-07-16 [SC]
 8F8D83B5270649A080648255EAF23F4A0BD34BF0
```

Similarly for other package types and package managers

## 6. Preconditions:

- A network product document containing information regarding software package integrity checks, including details of how the integrity check is carried out, where public keys or certificates of sources authorized to sign software packages are stored on the network product and who these sources are, and what evidence is created to prove that the integrity

check has been executed and what the result of the check was. Documentation which describes the installation procedure including how a user is authorized and authenticated to perform the installation process.

- A valid network product software load/package and one that is not-valid (or could be deemed to have been tampered with) are available.
7. **Test Objective/ Purpose:** To ensure that the NF validates the software package integrity before installation/upgrade stage strictly using the Secure cryptographic controls prescribed in Table1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only and to prevent the installation/execution of tampered software.

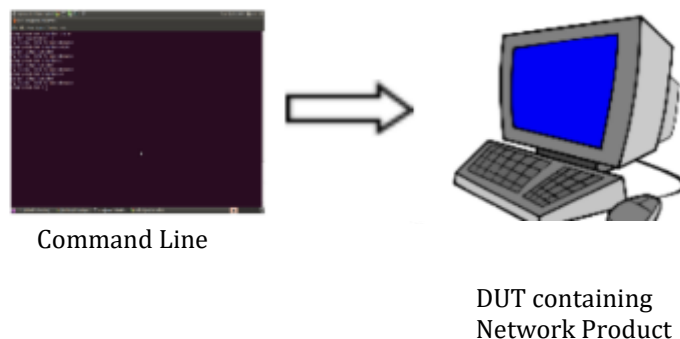
8. **Test Plan:**

**8.1 Number of Test Case Scenarios**

**8.1.1 Test Scenario for .dsc files with dpkg:** This test scenario verifies that integrity checks are carried out before software package is updated. (Additional Test scenarios based on the OEM document)

**8.2 Tools required:** Command Line Interface of the DUT

**8.3 Test Setup:**



**8.4 Test Execution Step:**

- Try to install updates from the valid software package (Case 1)
- Try to install updates from the invalid software package (Case 2)

9. **Expected Results:**

- (Case 1) Software is updated using the provided package.
- (Case 2) Software is not updated using the provided package.



10. **Expected Format of Evidence:** Evidence suitable for the interface, e.g. screenshot containing the operational results.

11. **Test Execution:**

➤ **Test Case Number: 01**

a. **Test Case Name:** TC1\_SOFTWARE\_INTEGRITY\_CHECK\_INSTALLATION

b. **Test Case Description:** Tester tries to verify that integrity checks are carried out on the update package before any updates are carried out Given a genuine .dsc file for the software update, tester uses following command to source the .dsc file which will be used to create .deb file for applying updates

(Case 1) *dpkg-source -x package\_name.dsc*

```
dpkg-source: info: extracting ufw in ufw-0.36
dpkg-source: info: unpacking ufw_0.36.orig.tar.gz
dpkg-source: info: unpacking ufw_0.36-6ubuntu1.debian.tar.xz
dpkg-source: info: using patch list from debian/patches/series
dpkg-source: info: applying 0001-optimize-boot.patch
dpkg-source: info: applying 0002-fix-check-requirements.patch
dpkg-source: info: applying 0003-lp1838764.patch
dpkg-source: info: applying 0004-make-root-tests-chain-order-agnostic.patch
dpkg-source: info: applying 0005-use-only-python3.patch
dpkg-source: info: applying 0006-bug921680.patch
dpkg-source: info: applying 0007-bug921680-pt2.patch
dpkg-source: info: applying 0008-fix-check-requirements-again.patch
dpkg-source: info: applying 0009-empty-non-functioning-iptables-modules.patch
dpkg-source: info: applying 0010-add-other-firewall-checks.patch
dpkg-source: info: applying 0011-suppress-legacy-warnings-in-tests.patch
dpkg-source: info: applying 0012-set-default-policy-after-load.patch
dpkg-source: info: applying 0013-unconditionally-reload-with-delete.patch
```

Given a malformed .dsc file for the software update, tester uses following command to source the .dsc file which will be used to create .deb file for applying updates

(Case 2) *dpkg-source -x package\_name.dsc*

```
gpgv: can't allocate lock for '/home/pratik/.gnupg/trustedkeys.gpg'
gpgv: Signature made Tuesday 26 October 2021 03:16:52 AM IST
gpgv: using RSA key 8F8D83B5270649A080648255EAF23F4A0BD34BF0
gpgv: issuer "mfo@canonical.com"
gpgv: BAD signature from "Mauricio Faria de Oliveira <mfo@canonical.com>"
dpkg-source: error: failed to verify signature on ./ufw_0.36-6ubuntu1.dsc
```

c. **Test Observation:**

- (Case 1) Tester verifies that the package has generated a source code using the .dsc file
- (Case 2) Tester verifies that integrity check fails and no source code has been generated

12. **Test Case Result:**

| Sl. No | TEST CASE NAME                            | PASS/FAIL | Remarks |
|--------|-------------------------------------------|-----------|---------|
| 1      | TC1_SOFTWARE_INTEGRITY_CHECK_INSTALLATION |           |         |

## 2.13.5 TSTP Software Integrity Check – Boot

|                                   |                          |                  |
|-----------------------------------|--------------------------|------------------|
| Session Management function ITSAR | ITSAR No: ITSAR111092401 | Clause no:2.13.5 |
|-----------------------------------|--------------------------|------------------|

**Note:** The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

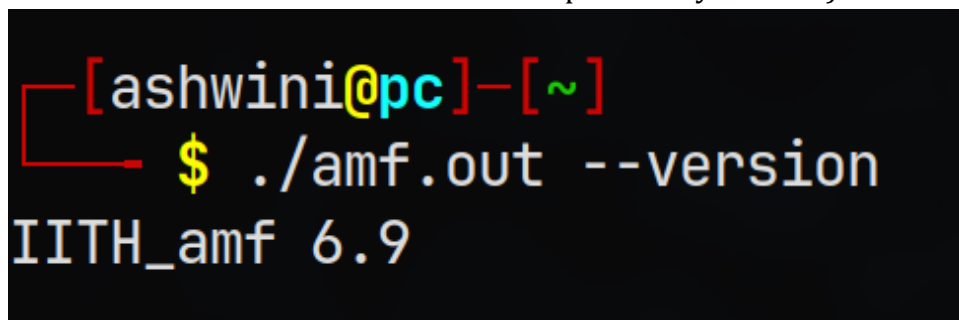
- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 13 - Other Security Requirements
2. **<Security Requirement No & Name >** 2.13.5 Software Integrity Check - Boot
3. **<Requirement Description: >** The SMF shall verify the integrity of a software component by comparing the result of a measurement of the component, typically a standard cryptographic hash generated strictly using the Secure cryptographic controls prescribed in Table 1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” to the expected reference value.

#### 4. DUT Confirmation Details:

- Use command to get Application No/Version
- Use command to get OS Version/No

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)



```
[ashwini@pc]~$./amf.out --version
IITH_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
[ashwini@pc]~$ cat /etc/os-release
```

|    | File: /etc/os-release                                     |
|----|-----------------------------------------------------------|
| 1  | NAME="Manjaro Linux"                                      |
| 2  | PRETTY_NAME="Manjaro Linux"                               |
| 3  | ID=manjaro                                                |
| 4  | ID_LIKE=arch                                              |
| 5  | BUILD_ID=rolling                                          |
| 6  | ANSI_COLOR="32;1;24;144;200"                              |
| 7  | HOME_URL="https://manjaro.org/"                           |
| 8  | DOCUMENTATION_URL="https://wiki.manjaro.org/"             |
| 9  | SUPPORT_URL="https://forum.manjaro.org/"                  |
| 10 | BUG_REPORT_URL="https://docs.manjaro.org/reporting-bugs/" |
| 11 | PRIVACY_POLICY_URL="https://manjaro.org/privacy-policy/"  |
| 12 | LOGO=manjarolinux                                         |

```
jamesnaan@KS-Laptop:~$ sudo cat /etc/os-release
[sudo] password for jamesnaan:
NAME="Ubuntu"
VERSION="20.04.6 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.6 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
```

## 5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF\_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2dea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

## 6. Preconditions:

- A network product document containing information regarding software package integrity checks, including details of how the integrity check is carried out, where public keys or certificates of sources authorized to sign software packages are stored on the network product and who these sources are, and what evidence is created to prove that the integrity check has been executed and what the result of the check was. Documentation which

describes the installation procedure including how a user is authorized and authenticated to perform the installation process.

- A valid network product software load/package and one that is not-valid (or could be deemed to have been tampered with) are available.
7. **Test Objective/ Purpose:** To ensure that the NF validates the software component integrity during boot strictly using the Secure cryptographic controls prescribed in Table1 of the latest document “Cryptographic Controls for Indian Telecom Security Assurance Requirements (ITSAR)” only.

8. **Test Plan:**

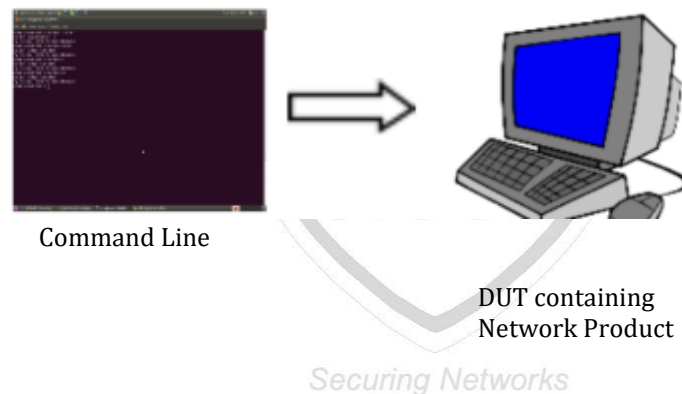
**8.1 Number of Test Case Scenarios**

**8.1.1 Test Scenario for .dsc files with dpkg**

This test scenario verifies that integrity checks are carried out during boot for the software package. (Additional Test scenarios based on the OEM document)

**8.2 Tools required:** Command Line Interface of the DUT

**8.3 Test Setup:**



**8.4 Test Execution Step:**

- Check that all pre-conditions are met.
- Boot the system containing software package
- Calculate the hash of the software component and compare it to the expected reference hash.  
For sha256 use below command,  
  
*sha256sum name\_of\_the\_software*
- If the calculated hash matches the expected hash, it indicates that the integrity check has passed.

9. **Expected Results:** The system boots, and the calculated hash matches the expected reference hash value.
10. **Expected Format of Evidence:** Evidence suitable for the interface, e.g. screenshot containing the operational results.
11. **Test Execution:**
- **Test Case Number: 01**
  - a. **Test Case Name:** TC1\_SOFTWARE\_INTEGRITY\_CHECK\_BOOT
  - b. **Test Case Description:** Tester tries to verify that integrity checks are carried out on the software package during boot.
  - c. **Execution Steps:**
    - Calculate the hash of the bootable software component using a chosen hash function.
    - Reboot the System.
    - After the system boots, log in and open a terminal.
    - Calculate the hash of the bootable software component using the same hash.
    - Compare the calculated hash in with the hash calculated before booting.
  - d. **Test Observation:**
    - Case 1: The system boots successfully, and the calculated hash matches the reference hash value saved earlier. (Positive Test Case)
    - Case 2: The system boots, but the calculated hash does not match the reference hash value saved earlier. (Negative Test Case)

12. **Test Case Result:**

| Sl. No | TEST CASE NAME                    | PASS/FAIL | Remarks |
|--------|-----------------------------------|-----------|---------|
| 1      | TC1_SOFTWARE_INTEGRITY_CHECK_BOOT |           |         |

---

## 2.13.6 TSTP Report for Evaluation of Unused Physical and Logical Interfaces Disabling

---

|                                   |                          |                  |
|-----------------------------------|--------------------------|------------------|
| Session Management function ITSAR | ITSAR No: ITSAR111092401 | Clause no:2.13.6 |
|-----------------------------------|--------------------------|------------------|

**Note:** The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **<ITSAR Section No & Name>**Section 13: Other Security Requirements
2. **<Security Requirement No & Name >** 2.13.6 Unused Physical and Logical Interfaces Disabling
3. **<Requirement Description: >**SMF shall support the mechanism to verify both the physical and logical interfaces exist in the product. Physical and logical accessible interfaces (except console interface) which are not under use shall be disabled so that they remain inactive even in the event of reboot.

Note: This may not be applicable for GVNP Type 1 and Type 2.

### 4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No

Command used: **ifconfig** (To find IP information and all interfaces)



```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
 inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
 inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
 ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
 RX packets 180276 bytes 177898049 (169.6 MiB)
 RX errors 0 dropped 314 overruns 0 frame 0
 TX packets 57117 bytes 10085270 (9.6 MiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
 device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 inet6 ::1 prefixlen 128 scopeid 0x10<host>
 loop txqueuelen 1000 (Local Loopback)
 RX packets 111 bytes 10484 (10.2 KiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 111 bytes 10484 (10.2 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)

**Here we are assuming DUT to be SMF, but this test must be conducted for each network function.)**

```
amf@localhost $./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

|    | File: /etc/os-release                                                               |
|----|-------------------------------------------------------------------------------------|
| 1  | NAME="Ubuntu"                                                                       |
| 2  | VERSION="20.04.5 LTS (Focal Fossa)"                                                 |
| 3  | ID=ubuntu                                                                           |
| 4  | ID_LIKE=debian                                                                      |
| 5  | PRETTY_NAME="Ubuntu 20.04.5 LTS"                                                    |
| 6  | VERSION_ID="20.04"                                                                  |
| 7  | HOME_URL="https://www.ubuntu.com/"                                                  |
| 8  | SUPPORT_URL="https://help.ubuntu.com/"                                              |
| 9  | BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"                                 |
| 10 | PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy" |
| 11 | VERSION_CODENAME=focal                                                              |
| 12 | UBUNTU_CODENAME=focal                                                               |

## 5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF\_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:** A List of all physical and logical interfaces that exist within the Network Product should be submitted by the OEM to the TSTL. The Vendor must specify which physical or logical interfaces are necessary for the operation of the Network Product.
7. **Test Objective:** To ensure that Physical and logical accessible interfaces (except console interface) which are not under use shall be disabled so that they remain inactive even in the event of reboot.
8. **Test Plan:**
  - 8.1 Number of Test case scenarios: 1**
    - Test to physical and logical accessible interfaces (except console interface) which are not under use shall be disabled so that they remain inactive even in the event of reboot.



**8.2 Tools used:** NULL

**8.3 Test Execution Steps:**

1. The tester should verify that the physical & logical interfaces mentioned in the OEM/Vendor documentation exist within the DUT.
  2. The tester should verify that the physical & logical interfaces that are deemed unused in the network product documentation are disabled.
  3. The tester should reboot the DUT and again verify that the physical & logical interfaces that are deemed unused in the network product documentation are disabled.
9. **Expected Results for Pass:** The unused physical and/or logical interfaces (except console interface) within the Network product are disabled and remain inactive even after an event of reboot.

10. **Expected format of Evidence:** Screenshot/log files describing the status(active/inactive) of the Physical and/or logical interfaces.

11. **Test Execution:**

➤ **Test Case Number:** 01

**a. Test Case Name:** TC\_UNUSED\_PHY\_LOG\_INF\_DIS

**b. Test Case Description:** To ensure that Physical and logical accessible interfaces (except console interface) which are not under use shall be disabled so that they remain inactive even in the event of reboot.

**c. Execution Steps:**

**Note:** The commands/tools specified in the below examples are native to the Ubuntu 20.04 OS, the tester must use the tools/commands that are compatible with the distribution and version of OS supported by the DUT.

1. The tester should verify that all the physical & logical interfaces mentioned in the OEM/Vendor documentation exist within the DUT.

We use the *ifconfig* or *ip* utility to obtain information regarding all network interfaces that exist on the system.

The command used for listing all network interfaces along with their details is "*ip a*"

```
siddhesh@stark99:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
 link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
 inet 127.0.0.1/8 scope host lo
 valid_lft forever preferred_lft forever
 inet6 ::1/128 scope host
 valid_lft forever preferred_lft forever
2: eno1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group default qlen 1000
 link/ether ac:1f:6b:8b:95:80 brd ff:ff:ff:ff:ff:ff
 altname enp2s0
3: eno2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
 link/ether 0c:c4:7a:e7:d5:2c brd ff:ff:ff:ff:ff:ff
 altname enp0s31f6
 inet 192.168.126.123/23 brd 192.168.127.255 scope global dynamic noprefixroute eno2
 valid_lft 598561sec preferred_lft 598561sec
 inet6 fe80::632c:3794:7bdb:594c/64 scope link noprefixroute
 valid_lft forever preferred_lft forever
```

We use the *lsusb* command to list all the USB devices connected to the system via USB ports.

```
siddhesh@stark99:~$ lsusb
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 004: ID 1a2c:0042 China Resource Semico Co., Ltd Usb Mouse
Bus 001 Device 002: ID 046d:c31d Logitech, Inc. Media Keyboard K200
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
siddhesh@stark99:~$
```

2. The tester should verify that the physical & logical interfaces that are deemed unused in the network product documentation are disabled.

We check status of each interface using the command *ip -s link show <interfacename>*

### Example of Interface status being showed as disabled (DOWN)

```
siddhesh@stark99:~$ ip -s link show eno1
2: eno1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN mode DEFAULT group default qlen 1000
 link/ether ac:1f:6b:8b:95:80 brd ff:ff:ff:ff:ff:ff
 RX: bytes packets errors dropped overrun mcast
 0 0 0 0 0 0
 TX: bytes packets errors dropped carrier collsns
 0 0 0 0 0 0
 altname enp2s0
```

### Example of Interface status being showed as enabled(UP)

```
siddhesh@stark99:~$ ip -s link show eno2
3: eno2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
 link/ether 0c:c4:7a:e7:d5:2c brd ff:ff:ff:ff:ff:ff
 RX: bytes packets errors dropped overrun mcast
 744458861 2350629 0 0 0 30713
 TX: bytes packets errors dropped carrier collsns
 513432043 1089817 0 0 0 0
 altname enp0s31f6
```

For checking the USB ports, the tester can connect to a specific port and check if the device is being detected on that port, using the 'usb-devices' command (for Ubuntu 20.04)

```
siddhesh@stark99:~$ usb-devices

T: Bus=01 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#= 1 Spd=480 MxCh=16
D: Ver= 2.00 Cls=09(hub) Sub=00 Prot=01 MxPS=64 #Cfgs= 1
P: Vendor=1d6b ProdID=0002 Rev=05.15
S: Manufacturer=Linux 5.15.0-78-generic xhci-hcd
S: Product=xHCI Host Controller
S: SerialNumber=0000:00:14.0
C: #Ifs= 1 Cfg#= 1 Atr=e0 MxPwr=0mA
I: If#=0x0 Alt= 0 #EPs= 1 Cls=09(hub) Sub=00 Prot=00 Driver=hub

T: Bus=01 Lev=01 Prnt=01 Port=00 Cnt=01 Dev#= 2 Spd=1.5 MxCh= 0
D: Ver= 1.10 Cls=00(>ifc) Sub=00 Prot=00 MxPS= 8 #Cfgs= 1
P: Vendor=046d ProdID=c31d Rev=66.01
S: Manufacturer=Logitech
S: Product=USB Keyboard
C: #Ifs= 2 Cfg#= 1 Atr=a0 MxPwr=90mA
I: If#=0x0 Alt= 0 #EPs= 1 Cls=03(HID) Sub=01 Prot=01 Driver=usbhid
I: If#=0x1 Alt= 0 #EPs= 1 Cls=03(HID) Sub=00 Prot=00 Driver=usbhid
```

The above command lists information of the devices connected to the system via USB ports. If the USB port is enabled, device will be detected, and vice versa.

## Securing Networks

3. The tester should reboot the DUT and again verify that the physical & logical interfaces that are deemed unused in the network product documentation are disabled.

Again, we used commands specified in step 2 for verifying status of physical and logical interfaces.

NOTE: Compare the status of Interfaces with the previous status, and mention the outcome of comparison in the report.

### 12. Test Case Result:

| Sl. No | TEST CASE NAME                | PASS/FAIL | Remarks |
|--------|-------------------------------|-----------|---------|
| 1      | TC_<br>UNUSED_PHY_LOG_INF_DIS |           |         |

---

## 2.13.7 TSTP for Evaluation of No Default Profile

---

|                                   |                          |                  |
|-----------------------------------|--------------------------|------------------|
| Session Management function ITSAR | ITSAR No: ITSAR111092401 | Clause no:2.13.7 |
|-----------------------------------|--------------------------|------------------|

**Note:** The test procedure outlined herein is equally applicable to SMF. It has been duly verified that all test scenarios and steps remain consistent throughout. It is ensured that any references to DUT or SMF in the screenshots/commands yield identical results when testing with SMF

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **<ITSAR Section No & Name>**Section 13: Other Security requirements
2. **<Security Requirement No & Name>**2.13.7 No Default Profile
3. **<Requirement Description: >**Predefined or default user accounts (other than Admin/Root) in SMF shall be deleted or disabled.
4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No

*Securing Networks*

- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
 inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
 inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
 ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
 RX packets 180276 bytes 177898049 (169.6 MiB)
 RX errors 0 dropped 314 overruns 0 frame 0
 TX packets 57117 bytes 10085270 (9.6 MiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
 device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 inet6 ::1 prefixlen 128 scopeid 0x10<host>
 loop txqueuelen 1000 (Local Loopback)
 RX packets 111 bytes 10484 (10.2 KiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 111 bytes 10484 (10.2 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Example command used: **./SMF.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF)  
Here we are assuming DUT to be SMF, but this test must be conducted for each network function.

```
amf@localhost $./amf.out --version
IIT_amf 6.9
```

- Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

|    | File: /etc/os-release                                                               |
|----|-------------------------------------------------------------------------------------|
| 1  | NAME="Ubuntu"                                                                       |
| 2  | VERSION="20.04.5 LTS (Focal Fossa)"                                                 |
| 3  | ID=ubuntu                                                                           |
| 4  | ID_LIKE=debian                                                                      |
| 5  | PRETTY_NAME="Ubuntu 20.04.5 LTS"                                                    |
| 6  | VERSION_ID="20.04"                                                                  |
| 7  | HOME_URL="https://www.ubuntu.com/"                                                  |
| 8  | SUPPORT_URL="https://help.ubuntu.com/"                                              |
| 9  | BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"                                 |
| 10 | PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy" |
| 11 | VERSION_CODENAME=focal                                                              |
| 12 | UBUNTU_CODENAME=focal                                                               |

## 5. DUT Configuration:

- To get the hash of configuration file if the file is a ASCII text file  
Command used: **sha256sum SMF\_config.conf** (To get hash/digest of config file)



```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57ee76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

- To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

## 6. Preconditions:

- There should be a documentation provided mentioning all accounts which are pre-installed (mandatory accounts).

## 7. Test Objective:

- All the predefined accounts should be deleted except for mandatory accounts

## 8. Test Plan:

### 8.1 Number of Test case scenarios: 1

- Test to check that all the predefined accounts are deleted except for mandatory accounts

### 8.2. Test Setup Diagram:



### 8.3 Tools Used:

### 8.4 Test Execution Steps

- Check that all pre-conditions are met.
- Login to SMF as root.
- Check the mandatory accounts from the documentation provided.
- We can also check the admin/root accounts using the command also: -  
**getent group root adm admin**

- We can check the accounts present with the command: **-compgen -u**
- Other than the mandatory accounts all the accounts should be deleted, using the command: **-userdel <username>**

9. **Expected Results for Pass:** All the default accounts should be deleted by default.

10. **Expected Format of Evidence:** Screenshots showing there are only mandatory accounts.

11. **Test Execution:**

- **Test Case Number:** 01
- a. **Test Case Name:** TC\_NO\_DEFAULT\_PROFILE
- b. **Test Case Description:** Ensuring there are no default accounts present.
- c. **Execution Steps:**

Can find the admin accounts with this command

```
vm2@vm2:~$ getent group root adm admin
root:x:0:
adm:x:4:syslog,vm2
```

We can find all the present accounts

```
vm2@vm2:~$ compgen -u
root
daemon
bin
sys
sync
games
man
lp
mail
news
uucp
proxy
www-data
backup
list
irc
gnats
nobody
systemd-network
systemd-resolve
messagebus
systemd-timesync
syslog
```

d. **Test Observations:**

It should be ensured that all the predefined accounts should be deleted.

12. **Test Case Result:**

| Sl. No | OUTCOME OF RUNNING THE SCRIPT<br>(CASE 1/CASE2) | PASS/FAIL | Remarks |
|--------|-------------------------------------------------|-----------|---------|
| 1      | TC_NO_DEFAULT_PROFILE                           |           |         |



# Specific Security Requirements

## 3.1.1 TSTP for Evaluation of Priority of UP security Policy

|                                   |                          |                 |
|-----------------------------------|--------------------------|-----------------|
| Session Management function ITSAR | ITSAR No: ITSAR111092401 | Clause no:3.1.1 |
|-----------------------------------|--------------------------|-----------------|

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 1: Priority of UP security policy.
2. **<Security Requirement No & Name >** 3.1.1: Priority of UP Security Policy
3. **<Requirement Description: >** User Plane Security Policy from UDM takes precedence over locally configured User Plane Security Policy.

[Reference: TEC 25883:2022/TSDSI STD T1.3GPP 33.515-16.4.0 V.1.0.0. section 4.2.2.1.1]

### 4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```

amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
 inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
 inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
 ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
 RX packets 180276 bytes 177898049 (169.6 MiB)
 RX errors 0 dropped 314 overruns 0 frame 0
 TX packets 57117 bytes 10085270 (9.6 MiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
 device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 inet6 ::1 prefixlen 128 scopeid 0x10<host>
 loop txqueuelen 1000 (Local Loopback)
 RX packets 111 bytes 10484 (10.2 KiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 111 bytes 10484 (10.2 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Example command used: **./amf.out** (Used in IITH testbed to get AMF version. Check with OEM manufacturer document for command specific to your AMF)

**Here we are assuming DUT to be AMF, but this test must be conducted for each network function.**

```

amf@localhost $./amf.out --version
IIT_amf 6.9

```

Command used: **cat /etc/os-release** (To get OS information)

```

amf@localhost $ cat /etc/os-release
File: /etc/os-release
1 NAME="Ubuntu"
2 VERSION="20.04.5 LTS (Focal Fossa)"
3 ID=ubuntu
4 ID_LIKE=debian
5 PRETTY_NAME="Ubuntu 20.04.5 LTS"
6 VERSION_ID="20.04"
7 HOME_URL="https://www.ubuntu.com/"
8 SUPPORT_URL="https://help.ubuntu.com/"
9 BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
10 PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
11 VERSION_CODENAME=focal
12 UBUNTU_CODENAME=focal

```

## 5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum AMF\_config.conf** (To get hash/digest of config file)

```

amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf

```

To get the hash of OS if using docker

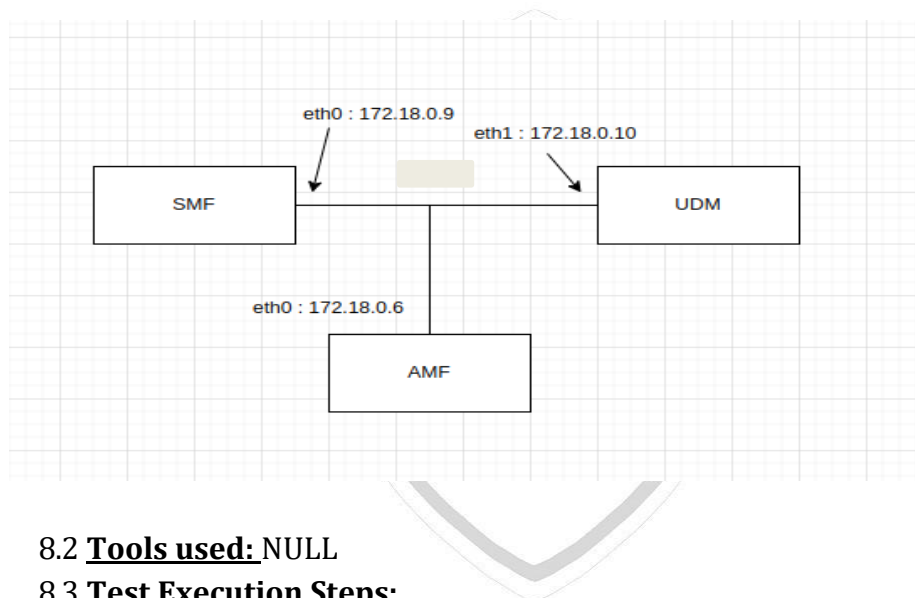
Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:** Test environment with AMF and UDM may be simulated. Both UDM and SMF under test are configured with UP security policy, and the UP-security policies are different. There is no Session Management Subscription data in SMF.
7. **Test Objective:** Verify that the user plane security policy from the UDM takes precedence at the SMF under test over locally configured user plane security policy.

## 8. **Test Plan:**

### 8.1 **Test Bed Diagram:**



### 8.2 **Tools used:** NULL

### 8.3 **Test Execution Steps:**

1. The tester triggers PDU session establishment procedure by sending Nsmf PDUSession Creates Context Request message to the SMF.
  2. The SMF under test retrieves the Session Management Subscription data using Nudm SDM Get service from UDM, where the Session Management Subscription data includes the user plane security policy stored in UDM.
  3. The tester captures the Namf\_Communication\_N1N2MessageTransfer message sent from the SMF under test to the AMF.
  4. Verify the captured-UP Security Policy in the Security Indication IE with the Security Policy configured in UDM and the security policy locally configured at the SMF.
9. **Expected results for Pass:** There is a Security Indication IE in the N2 SM information contained in the Namf\_Communication\_N1N2MessageTransfer message, which is the same with the UP-security policy configured in the UDM.



10. **Expected Format of Evidence:** Wireshark trace/screenshot that captures the request/response messages mentioned in the above execution steps.

11. **Test Execution:**

➤ **Test Case Number:** 01

a. **Test Case Name:** TC\_UP\_POLICY\_PRECEDENCE\_SMF

b. **Test Case Description:** Verify that the user plane security policy from the UDM takes precedence at the SMF under test over locally configured user plane security policy.

c. **Test Execution Steps:**

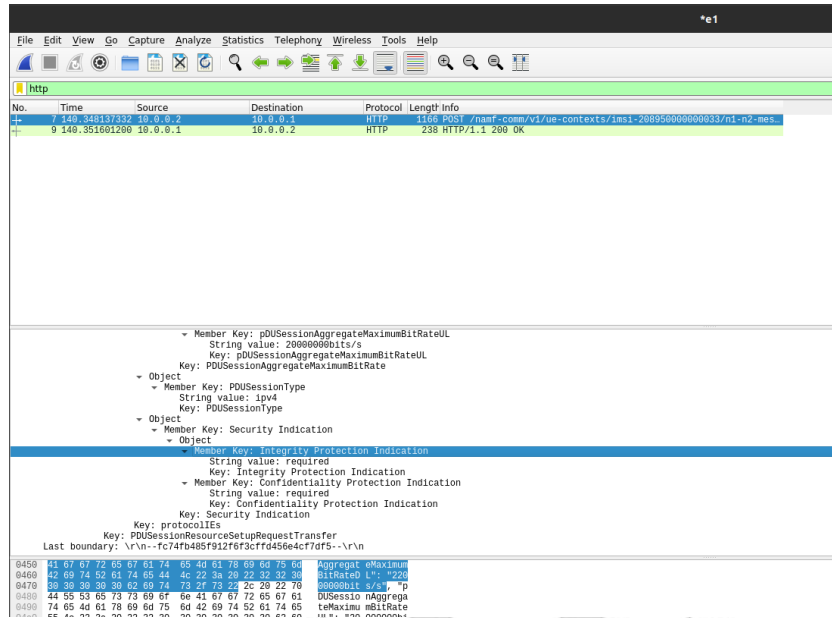
1. The tester triggers PDU session establishment procedure by sending Nsmf\_PDUSession\_CreateSMContext Request message to the SMF.

| [ ip.addr == 172.18.0.9 and http2 ]                                                                         |              |            |             |          |                                                                |
|-------------------------------------------------------------------------------------------------------------|--------------|------------|-------------|----------|----------------------------------------------------------------|
| No.                                                                                                         | Time         | Source     | Destination | Protocol | Length Info                                                    |
| 165                                                                                                         | 6.968381775  | 172.18.0.6 | 172.18.0.9  | HTTP2    | 124 Magic, SETTINGS[0], WINDOW_UPDATE[0]                       |
| 167                                                                                                         | 6.968499388  | 172.18.0.6 | 172.18.0.9  | HTTP2    | 235 HEADERS[1]: POST /Nsmf_PDUSession_CreateSMContext, DATA[1] |
| 169                                                                                                         | 6.968501431  | 172.18.0.6 | 172.18.0.6  | HTTP2    | 380 SETTINGS[0], SETTINGS[0], HEADERS[1]: 200 OK, DATA[1]      |
| 171                                                                                                         | 6.961434312  | 172.18.0.6 | 172.18.0.9  | HTTP2    | 83 GOAWAY[0]                                                   |
| 183                                                                                                         | 6.963338529  | 172.18.0.6 | 172.18.0.9  | HTTP2    | 124 Magic, SETTINGS[0], WINDOW_UPDATE[0]                       |
| 185                                                                                                         | 6.963463857  | 172.18.0.6 | 172.18.0.9  | HTTP2    | 239 HEADERS[1]: POST /Nsmf_PDUSession_UpdateSMContext, DATA[1] |
| 186                                                                                                         | 6.963490676  | 172.18.0.9 | 172.18.0.6  | HTTP2    | 90 SETTINGS[0], SETTINGS[0]                                    |
| 188                                                                                                         | 6.963561685  | 172.18.0.6 | 172.18.0.9  | HTTP2    | 75 SETTINGS[0]                                                 |
| 189                                                                                                         | 6.963799584  | 172.18.0.9 | 172.18.0.6  | HTTP2    | 137 HEADERS[1]: 200 OK, DATA[1]                                |
| 190                                                                                                         | 6.963877469  | 172.18.0.6 | 172.18.0.9  | HTTP2    | 83 GOAWAY[0]                                                   |
| 428                                                                                                         | 19.208056373 | 172.18.0.6 | 172.18.0.9  | HTTP2    | 124 Magic, SETTINGS[0], WINDOW_UPDATE[0]                       |
| 430                                                                                                         | 19.208219491 | 172.18.0.6 | 172.18.0.9  | HTTP2    | 235 HEADERS[1]: POST /Nsmf_PDUSession_CreateSMContext, DATA[1] |
| 431                                                                                                         | 19.208247610 | 172.18.0.9 | 172.18.0.6  | HTTP2    | 90 SETTINGS[0], SETTINGS[0]                                    |
| 435                                                                                                         | 19.208323956 | 172.18.0.6 | 172.18.0.9  | HTTP2    | 75 SETTINGS[0]                                                 |
| 439                                                                                                         | 19.208625978 | 172.18.0.9 | 172.18.0.6  | HTTP2    | 366 HEADERS[1]: 200 OK, DATA[1]                                |
| 440                                                                                                         | 19.208746126 | 172.18.0.6 | 172.18.0.9  | HTTP2    | 83 GOAWAY[0]                                                   |
| 454                                                                                                         | 19.210738799 | 172.18.0.6 | 172.18.0.9  | HTTP2    | 124 Magic, SETTINGS[0], WINDOW_UPDATE[0]                       |
| 457                                                                                                         | 19.210877532 | 172.18.0.6 | 172.18.0.9  | HTTP2    | 239 HEADERS[1]: POST /Nsmf_PDUSession_UpdateSMContext, DATA[1] |
| 458                                                                                                         | 19.210883293 | 172.18.0.9 | 172.18.0.6  | HTTP2    | 90 SETTINGS[0], SETTINGS[0]                                    |
| 461                                                                                                         | 19.210936652 | 172.18.0.6 | 172.18.0.9  | HTTP2    | 75 SETTINGS[0]                                                 |
| 462                                                                                                         | 19.211313235 | 172.18.0.9 | 172.18.0.6  | HTTP2    | 137 HEADERS[1]: 200 OK, DATA[1]                                |
| 463                                                                                                         | 19.211212925 | 172.18.0.6 | 172.18.0.9  | HTTP2    | 83 GOAWAY[0]                                                   |
| 473                                                                                                         | 19.211652984 | 172.18.0.4 | 172.18.0.9  | HTTP2    | 124 Magic, SETTINGS[0], WINDOW_UPDATE[0]                       |
| Frame 167: 235 bytes on wire (1880 bits), 235 bytes captured (1880 bits) on interface br-3b616f41fc2a, id 0 |              |            |             |          |                                                                |
| Ethernet II, Src: 02:42:ac:12:00:06 (02:42:ac:12:00:06), Dst: 02:42:ac:12:00:09 (02:42:ac:12:00:09)         |              |            |             |          |                                                                |
| Internet Protocol Version 4, Src: 172.18.0.6, Dst: 172.18.0.9                                               |              |            |             |          |                                                                |
| Transmission Control Protocol, Src Port: 59458, Dst Port: 8080, Seq: 59, Ack: 1, Len: 169                   |              |            |             |          |                                                                |
| Hypertext Transfer Protocol 2                                                                               |              |            |             |          |                                                                |
| Stream: HEADERS, Stream ID: 1, Length 41, POST /Nsmf_PDUSession_CreateSMContext                             |              |            |             |          |                                                                |
| Length: 41                                                                                                  |              |            |             |          |                                                                |
| Type: HEADERS (1)                                                                                           |              |            |             |          |                                                                |
| Flags: 0x04                                                                                                 |              |            |             |          |                                                                |

2. The SMF under test retrieves the Session Management Subscription data using Nudm\_SDM\_Get service from UDM, where the Session Management Subscription data includes the user plane security policy stored in UDM.

3. The tester captures the Namf\_Communication\_N1N2MessageTransfer message sent from the SMF under test to the AMF.

Namf\_Communication\_N1N2MessageTransfer containing UP Security Policy in Security Indication IE.



4. Verify the captured-UP Security Policy in the Security Indication IE with the Security Policy configured in UDM and the security policy locally configured at the SMF.

**Note:** The UP-security policy configured at UDM must take precedence over the locally configured security policy at SMF, and hence the UP-security policy captured in the Namf\_Communication\_N1N2MessageTransfer message must match with the security policy configured at the UDM and NOT the local policy at SMF.

- d. **Test Observations:** There is a Security Indication IE in the N2 SM information contained in the Namf\_Communication\_N1N2MessageTransfer message, which is the same with the UP-security policy configured in the UDM.

## 12. Test Results:

| SL. No | TEST CASE NAME              | PASS/FAIL | Remarks |
|--------|-----------------------------|-----------|---------|
| 1      | TC_UP_POLICY_PRECEDENCE_SMF |           |         |

### 3.2.1 TSTP for Evaluation of UP security Policy Check

|                                   |                          |                 |
|-----------------------------------|--------------------------|-----------------|
| Session Management function ITSAR | ITSAR No: ITSAR111092401 | Clause no:3.2.1 |
|-----------------------------------|--------------------------|-----------------|

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. **<ITSAR Section No & Name>** Section 2: Security functional requirements on the SMF checking UP security policy
2. **<Security Requirement No & Name >** 3.2.1: UP Security Policy Check
3. **<Requirement Description: >** The SMF shall verify that the UE's UP security policy received from the target ng-eNB/gNB is the same as the UE's UP security policy that the SMF has locally stored. If there is a mismatch, the SMF shall send its locally stored UE's UP security policy of the corresponding PDU sessions to the target gNB. This UP-security policy information, if included by the SMF, is delivered to the target ng-eNB/gNB in the Path-Switch Acknowledge message. The SMF shall log capabilities for this event and may take additional measures, such as raising an alarm.

[Reference: TEC 25883:2022/TSDSI STD T1.3GPP 33.515-16.4.0 V.1.0.0. section 4.2.2.1.3]

#### 4. **DUT Confirmation Details:**

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
amf@localhost $ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
 inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.0.255
 inet6 fe80::7883:83a1:8147:14e8 prefixlen 64 scopeid 0x20<link>
 ether 8c:16:45:32:57:09 txqueuelen 1000 (Ethernet)
 RX packets 180276 bytes 177898049 (169.6 MiB)
 RX errors 0 dropped 314 overruns 0 frame 0
 TX packets 57117 bytes 10085270 (9.6 MiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
 device interrupt 128 base 0x1000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 inet6 ::1 prefixlen 128 scopeid 0x10<host>
 loop txqueuelen 1000 (Local Loopback)
 RX packets 111 bytes 10484 (10.2 KiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 111 bytes 10484 (10.2 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./amf.out** (Used in IITH testbed to get AMF version. Check with OEM manufacturer document for command specific to your AMF)

**Here we are assuming DUT to be AMF, but this test must be conducted for each network function.**

```
amf@localhost $./amf.out --version
IIT_amf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

```
amf@localhost $ cat /etc/os-release
```

|    | File: /etc/os-release                                                               |
|----|-------------------------------------------------------------------------------------|
| 1  | NAME="Ubuntu"                                                                       |
| 2  | VERSION="20.04.5 LTS (Focal Fossa)"                                                 |
| 3  | ID=ubuntu                                                                           |
| 4  | ID_LIKE=debian                                                                      |
| 5  | PRETTY_NAME="Ubuntu 20.04.5 LTS"                                                    |
| 6  | VERSION_ID="20.04"                                                                  |
| 7  | HOME_URL="https://www.ubuntu.com/"                                                  |
| 8  | SUPPORT_URL="https://help.ubuntu.com/"                                              |
| 9  | BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"                                 |
| 10 | PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy" |
| 11 | VERSION_CODENAME=focal                                                              |
| 12 | UBUNTU_CODENAME=focal                                                               |

## 5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum AMF\_config.conf** (To get hash/digest of config file)

```
amf@localhost $ sha256sum AMF_Config.conf
fb0fe18402bd01380a57eef76e87084a15f6999b23e2edea1d08af1844429d30 AMF_Config.conf
```

To get the hash of OS if using docker

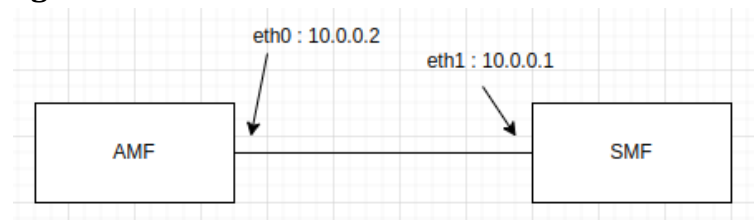
Command used: **docker images --digests** (To get hash/digest of config file)

```
amf@localhost $ docker images ubuntu --digests --format "{{.Repository}} {{.Tag}} {{.ID}} {{.Digest}}"
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

6. **Preconditions:** The SMF under test is preconfigured with a UE UP security policy.
7. **Test Objective:** Verify that the SMF checks the UP-security policy that is sent by the ng-eNB/gNB during handover.

## 8. **Test Plan:**

### 8.1 **Test Bed Diagram:**



### 8.2 **Tools used:** NULL

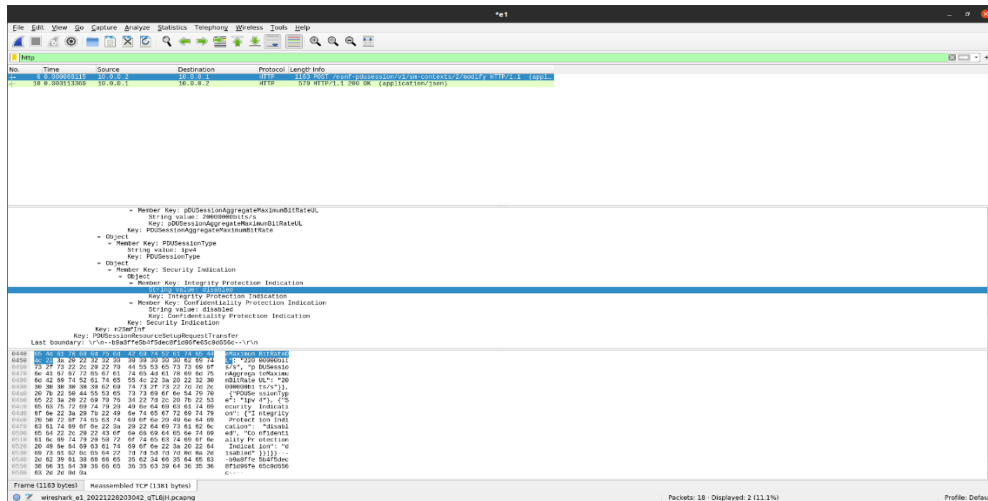
### 8.3 **Test Execution Steps:**

1. The tester sends the Nsmf\_PDU Session\_Update SM Context Request message to the SMF under test. A UE UP security policy different than the one preconfigured at the SMF under test is included in the Request message.
  2. The tester captures the Nsmf\_PDU Session\_Update SM Context Response message sent from the SMF under test.
9. **Expected results for Pass:** The preconfigured UE security policy is contained in the 'n2SmInf' IE in the captured Response message.
  10. **Expected Format of Evidence:** Wireshark trace/screenshot that captures the request/response messages mentioned in the above execution steps.

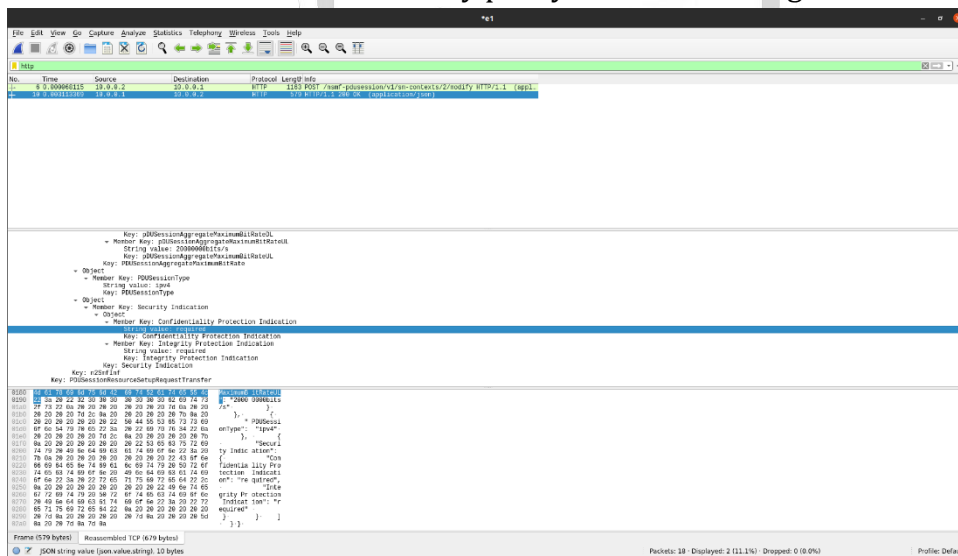
## 11. **Test Execution:**

### ➤ **Test Case Number:** 01

- a. **Test Case Name:** TC\_UP\_SECURITY\_POLICY\_SMF
- b. **Test Case Description:** Verify that the SMF checks the UP-security policy that is sent by the ng-eNB/gNB during handover.
- c. **Test Execution Steps:**
  1. The tester sends the Nsmf\_PDU Session\_Update SM Context Request message to the SMF under test. A UE UP security policy different than the one preconfigured at the SMF under test is included in the Request message. A Nsmf\_PDU Session\_Update SM Context Request is sent to SMF from AMF with UP Security policy different than one stored in SMF.



2. The tester captures the Nsmf\_PDUSession\_UpdateSMContext Response message sent from the SMF under test. The Nsmf\_PDUSession\_UpdateSMContext Response message contains the corrected-UP Security policy as the one configured in SMF.



3. The tester must verify that the UP-Security Policy present in the Nsmf\_PDUSession\_UpdateSMContext Response message with the one locally stored at the SMF.
- d. **Test Observations:** The preconfigured UE security policy is contained in the 'n2SmInf' IE in the captured Response message.

## 12. Test Results:

| SL. No | TEST CASE NAME            | PASS/FAIL | Remarks |
|--------|---------------------------|-----------|---------|
| 1      | TC_UP_SECURITY_POLICY_SMF |           |         |



### 3.3.1 TSTP for Evaluation of Charging ID Uniqueness

|                                   |                          |                 |
|-----------------------------------|--------------------------|-----------------|
| Session Management function ITSAR | ITSAR No: ITSAR111092401 | Clause no:3.3.1 |
|-----------------------------------|--------------------------|-----------------|

- <DUT Details: > Ex: Router
- <DUT Software Version:>
- <Digest Hash of OS>
- <Digest Hash of Configuration>
- <Applicable ITSAR: >
- <ITSAR Version No:>
- <OEM Supplied Document list: >

1. <ITSAR Section No & Name> Section 3: Charging ID Uniqueness
2. <Security Requirement No & Name > 3.3.1 Charging ID Uniqueness
3. <Requirement Description: > The SMF shall support PDU session charging using service-based interface. The SMF shall collect charging information per PDU session for UEs served under 3GPP access and non-3GPP access. Every PDU session shall be assigned a unique identity number for billing purposes per PLMN. (i.e. the Charging Id).

[Reference: TEC 25883:2022/TSDSI STD T1.3GPP 33.515-16.4.0 V.1.0.0. section 4.2.2.1.4]

#### 4. DUT Confirmation Details:

Use the command line interface to get details of the machine on which test is conducted.

- Use command to get IP and Interfaces details
- Use command to get Application No/Version
- Use command to get OS Version/No
- Command used: **ifconfig -a** (To find IP information and all interfaces)

```
root@c554e49511f2:/code# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
 inet 172.18.0.8 netmask 255.255.0.0 broadcast 172.18.255.255
 ether 02:42:ac:12:00:08 txqueuelen 0 (Ethernet)
 RX packets 184 bytes 39903 (39.9 KB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 0 bytes 0 (0.0 B)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 loop txqueuelen 1000 (Local Loopback)
 RX packets 0 bytes 0 (0.0 B)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 0 bytes 0 (0.0 B)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Example command used: **./smf.out** (Used in IITH testbed to get SMF version. Check with OEM manufacturer document for command specific to your SMF.

```
root@c554e49511f2:/code# ./smf.out --version
IIT_smf 6.9
```

Command used: **cat /etc/os-release** (To get OS information)

|    | File: /etc/os-release                                                               |
|----|-------------------------------------------------------------------------------------|
| 1  | NAME="Ubuntu"                                                                       |
| 2  | VERSION="20.04.5 LTS (Focal Fossa)"                                                 |
| 3  | ID=ubuntu                                                                           |
| 4  | ID_LIKE=debian                                                                      |
| 5  | PRETTY_NAME="Ubuntu 20.04.5 LTS"                                                    |
| 6  | VERSION_ID="20.04"                                                                  |
| 7  | HOME_URL="https://www.ubuntu.com/"                                                  |
| 8  | SUPPORT_URL="https://help.ubuntu.com/"                                              |
| 9  | BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"                                 |
| 10 | PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy" |
| 11 | VERSION_CODENAME=focal                                                              |
| 12 | UBUNTU_CODENAME=focal                                                               |

## 5. DUT Configuration:

To get the hash of configuration file if the file is a ASCII text file

Command used: **sha256sum SMF\_config.conf** (To get hash/digest of config file)

```
root@c554e49511f2:/code# sha256sum SMF_config.conf
709c4db264fccd23439d57745cca1b1c2d180e64bb9bd89af4b25379f44e347a SMF_config.conf
```

To get the hash of OS if using docker

Command used: **docker images --digests** (To get hash/digest of config file)

```
ubuntu 20.04 88bd68917189 sha256:db8bf6f4fb351aa7a26e27ba2686cf35a6a409f65603e59d4c203e58387dc6b3
ubuntu latest fb52e22af1b0 sha256:9d6a8699fb5c9c39cf08a0871bd6219f0400981c570894cd8cbea30d3424a31f
```

**6. Preconditions:-** Test environment is set up with a Charging Function (CHF), which may be real or simulated, and the SMF under test. The tester is able to capture the traffic between the SMF under test and the CHF.

**7. Test Objective:-** To verify that the charging ID generated by the SMF for each PDU session is unique.

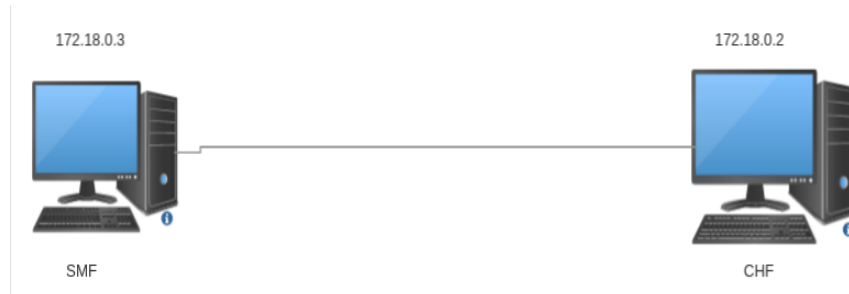
## 8. Test Plan:

8.1. Number of Test Cases:

8.1.1. Test Case ID Uniqueness

- This testcase is to check if the chargingid of every concurrent connection is unique

8.2. Test Diagram



8.3. Tools Used:- Wireshark

8.4. Execution Steps

- Launch the Wireshark app on the tester device.
- Trigger the establishment of the maximum number of concurrent PDU sessions that the SMF under test can handle.
- Stop the wireshark app and testbed as well.

9. **Expected Results for Pass:** The Charging ID is unique in each Charging Data Request.

10. **Expected Format of Evidence:** Screenshots of Wireshark capture

11. **Test Execution:**

➤ Test Case Number: 01

a. **Test Case Name:** TC\_CHARGING\_ID\_UNIQUENESS\_SMF

b. **Test Case Description:** This testcase is to check if the chargingid of every concurrent connection is unique

c. **Execution Steps:**

- Launch the Wireshark app on the tester device.
- Trigger the establishment of the maximum number of concurrent PDU sessions that the SMF under test can handle.
- Stop the Wireshark app and testbed as well.
- In order to view the communication between two communicating parties, put following filter in the Wireshark filter box

• **ip.src == <NF1\_addr> and ip.dst == <NF2\_addr>**

```

82.597... 172.18.0.3 172.18.0.2 TCP 66.58570 - 4443 [ACK] Seq=1 Ack=300 Win=64896 Len=0 TSval=933184213 TSecr=100
92.597... 172.18.0.2 172.18.0.3 TCP 66.4443 - 58570 [ACK] Seq=1 Ack=300 Win=64896 Len=0 TSval=1607756700 TSecr=
102.598... 172.18.0.2 172.18.0.3 TCP 1..4443 - 58570 [PSH, ACK] Seq=1 Ack=300 Win=64896 Len=131 TSval=1607756701
112.598... 172.18.0.3 172.18.0.2 TCP 66.58570 - 4443 [ACK] Seq=300 Ack=132 Win=64128 Len=0 TSval=933184213 TSecr=
122.598... 172.18.0.2 172.18.0.3 TCP 1..4443 - 58570 [PSH, ACK] Seq=132 Ack=300 Win=64896 Len=68 TSval=160775670
132.598... 172.18.0.3 172.18.0.2 TCP 66.58570 - 4443 [ACK] Seq=300 Ack=200 Win=64128 Len=0 TSval=933184213 TSecr=
142.598... 172.18.0.2 172.18.0.3 HTTP 66 HTTP/1.0 201 Created (application/json)
152.598... 172.18.0.3 172.18.0.2 TCP 66.58570 - 4443 [FIN, ACK] Seq=300 Ack=201 Win=64128 Len=0 TSval=933184213
162.598... 172.18.0.2 172.18.0.3 TCP 66.4443 - 58570 [ACK] Seq=201 Ack=301 Win=64896 Len=0 TSval=1607756701 TSecr=

Frame 8: 365 bytes on wire (2920 bits), 365 bytes captured (2920 bits) on interface br-039407f30e5f, id 0
Ethernet II, Src: 02:42:ac:12:00:03 (02:42:ac:12:00:03), Dst: 02:42:ac:12:00:02 (02:42:ac:12:00:02)
Internet Protocol Version 4, Src: 172.18.0.3, Dst: 172.18.0.2
Transmission Control Protocol, Src Port: 58570, Dst Port: 4443, Seq: 1, Ack: 1, Len: 299
Hypertext Transfer Protocol
JavaScript Object Notation: application/json
- Object
 - Member Key: subscriberIdentifier
 - Member Key: tenantIdentifier
 - Member Key: mnsConsumerIdentifier
 - Member Key: chargingId
 Number value: 4
 Key: chargingId

```

- Tester shall find the packets with post request containing “/nchf-converged charging/<version>/charging data” string and verify that each of these requests contain unique charging ID

- (This is a simulated environment with naïve api servers to act as CHF.)

d. **Test Observations:** The Charging ID is unique in each Charging Data Request

e. **Evidence provided:** Screenshots of Wireshark

**12. Test Case Result:**

| SL. No | TEST CASE NAME                | PASS/FAIL | Remarks |
|--------|-------------------------------|-----------|---------|
| 1      | TC_CHARGING_ID_UNIQUENESS_SMF |           |         |

