**3GPP TSG-SA3 Meeting #119 S3-245360**

Orlando, US, 11 -15 November 2024

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.1* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  | **33.117** | **CR** | **0059** | **rev** | **-** | **Current version:** | **18.3.0** |  |
|  | | | | | | | | |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* | | | | | | | | |
|  | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME |  | Radio Access Network |  | Core Network |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | |
| ***Title:*** | Corrections and test case updating to TS 33.117 | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** | Huawei; HiSilicon, BSI (DE), Nokia, Nokia Shanghai Bell | | | | | | | | | |
| ***Source to TSG:*** | S3 | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** | SCAS\_5G\_Maint | | | | |  | ***Date:*** | | | 2024-08-17 |
|  |  | | | |  | |  | | |  |
| ***Category:*** | **B** |  | | | | | ***Release:*** | | | Rel-19 |
|  | *Use one of the following categories:* ***F*** *(correction)* ***A*** *(mirror corresponding to a change in an earlier release)* ***B*** *(addition of feature),* ***C*** *(functional modification of feature)* ***D*** *(editorial modification)*  Detailed explanations of the above categories can be found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | | | | | | | | *Use one of the following releases: Rel-8 (Release 8) Rel-9 (Release 9) Rel-10 (Release 10) Rel-11 (Release 11) … Rel-15 (Release 15) Rel-16 (Release 16) Rel-17 (Release 17) Rel-18 (Release 18)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | | This is a living CR merged S3-243435,436,687,437,688, 968, 970, 972, 661, 404  For clause 4.2.2.2.3.1  In the current version of the TC\_AUTHORIZATION\_TOKEN\_VERIFICATION\_FAILURE\_ONE\_PLMN test case, a sub-test case where the NF function checks for the presence/absence of an OAuth2.0 token is missing. This is an important test, because in the current form, the tester assumes that this is checked by the NF.  For clause 4.3.4.2  - Fixing spelling errors  - Clarifying the wording of execution steps  For clause 4.3.4.6  The current test description is not general enough to fulfil the test purpose.  The test case was probably created with only apache web server in mind.  The goal of this test case should be not only to verify that the “upload” directory does not contain CGI/scripting related files or that the directories configured for CGI/scripting are distinct from the “upload” directory.  The test should verify that not any directory the web server user has write permission to, shall contain CGI/scripting related files or that the directories configured for CGI/scripting are distinct from any web server user writable directory.  For clause 4.3.4.8  Make the test case more generic.  Configuration directory read/write access is as crucial as access for single configuration files.  Expected results focus on system configuration files, while the test case is for web servers.  For clause 4.3.4.9  The GSMA NESAS has reviewed and proposed changes to the existing test case via the document S3-234423. The proposed changes in this document are with reference to Slide#37.  After intensive review and discussions it seems that if we introduce the term ‘explicitly needed default content’, we simply replace the term ‘default content’ by ‘explicitly needed default content’, basically, we have to provide clarity about what is a default content or what is explicitly needed default content.  For clause 4.3.4.10  The GSMA NESAS has reviewed and proposed changes to the existing test case via the document S3-234423. The proposed changes in this document are with reference to Slide#38.  Merger with BSI (DE) CR S3-242973  The current test description does not clearly order the tester to perform directory listing attempts.  If the web server does not comply to the directives given in its config, directory listing could still be possible.  Most web servers allow the directory listing configuration individual per endpoint. All endpoints should be checked.  For Clause 4.3.4.11  The GSMA NESAS has reviewed and proposed changes to the existing test case via the document S3-234423. The proposed changes in this document are with reference to Slide#39.  For Clause 4.3.4.12  The GSMA NESAS has reviewed and proposed changes to the existing test case via the document S3-234423. The proposed changes in this document are with reference to Slide#40.  Merged with BSI (DE) CR S3-242974  The current test description does not specify exactly how and what to test for.  There are a lot of different http error codes (e.g. 20+ in the 4xx class) that could potentially result in a custom error page or error message.  The vendor should provide documentation on how he handles custom error pages and messages and how to trigger those.  Four clause 4.3.4.13  Removing of unnecessary text.  For other clauses:  A lot of tests contain boiler plate text in the format of evidence description.  Those passages usually refer to something like: “a test report has to be provided” and/or “pass or fail”.  Demanding a test report specificaly or demanding pass/fail is redundant, since this is the general prupose of SCAS test execution. It is also already mentioned in the GSMA NESAS scheme. Furthermore, this is not mentioned in all test cases in 33.117 or any other SCAS document.  Display of “pass/fail” is not an evidence. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | For clause 4.2.2.2.3.1  - Add sub test case to check on the absence of the OAuth2.0 token  - Add necessary prerequisites to the test case  - Rename sub-test cases to A-I to be consistent with other sub-test cases.  - Change evidence to a packet trace (pcap)  For clause 4.3.4.2  - Fixed spelling errors in 1b  - Added the subject “tester” to all execution steps  For clause 4.3.4.6  - generified the requirement description  - generified the purpose  - generified the execution steps  - generified the expected results  For clause 4.3.4.8  - clarification of execution steps:  - identification of web server user  - specified access check for configuration files and directories  - clarification of expected results  - changed system to web server configuration files  - added configuration directories  For clause 4.3.4.9  GSMA suggested changes:  a) Change the Purpose text to: “To verify that any default content on the web server is explicitly needed for web server operation” for clarification.  Define what is or is not to be used as an automatic assessment tool.  b) Define “default content”. The test doesn’t align to the purpose (i.e. default content is permitted for operation of the web server).  c) As for other similar tests, these are repeating the requirement, rather than specifying the steps of the testing that is required for the test.  For clause 4.3.4.10  GSMA suggested changes:  Define what is or is not to be used as an automatic assessment tool.  Define what needs to be checked and what are the expected settings.  As for other similar tests, these are repeating the requirement, rather than specifying the steps of the testing that is required for the test.  BSI changes:  clarification of execution steps  - 1. step focusses on the actual web server configuration  - added step to perform directory listing attempts on all identified endpoints  - clarification of expected results according to execution steps  For clause 4.3.4.11  GSMA suggested changes:  Define what is or is not to be used as an automatic assessment tool.  As for other similar tests, these are repeating the requirement, rather than specifying the steps of the testing that is required for the test.  For clause 4.3.4.12  GSMA suggested changes:  Define what is or is not to be used as an automatic assessment tool.  Which error message/pages are to be checked?  As for other similar tests, these are repeating the requirement, rather than specifying the steps of the testing that is required for the test.  Four clause 4.3.4.13  Removing of unnecessary text.  For other clauses:  Removed occurences of demanding test report in format of evidence.  Removed occurences of demanding pass/fail in format of evidence.  If demanding pass/fail was the only format of evidence: replaves with generic format of evidence line fitting the test purpose. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | Network functions can be audited with the GSMA NESAS scheme, but the OAuth2.0 checks are incorrectly implemented. An attacker can exploit this fact and access network information while not being authorized.  Spelling mistakes in the test case  unclear if tester does action or system behaviour is observed  Test case does not handle directory access rights, which could lead to vulnerabilities.  The test case in its current form is not applicable to a generic web server.  Inconsistent test results and misinterpretation of test execution steps.  Enforcement of directory listing restriction in the web server configuration is not checked.  Redundancy and inconsistency in the format of evidence paragraphs. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | | 4.2.2.2.3.1, 4.2.3.4.1.1, 4.2.3.5.1, 4.2.3.5.2, 4.2.3.6.1, 4.2.3.6.2, 4.2.3.6.3, 4.2.4.1.1.2, 4.2.4.1.1.3, 4.2.4.1.2.1, 4.2.5.2.1, 4.2.6.2.3, 4.2.6.2.4, 4.3.2.1, 4.3.2.2, 4.3.2.3, 4.3.2.4, 4.3.2.5, 4.3.2.6, 4.3.3.1.1, 4.3.3.1.4, 4.3.4.2, 4.3.4.3, 4.3.4.4, 4.3.4.5, 4.3.4.6, 4.3.4.8, 4.3.4.9, 4.3.4.10, 4.3.4.11, 4.3.4.12, 4.3.4.13, 4.3.5.1, 4.3.6.2, 4.3.6.3, 4.3.6.4, 4.4.4 | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **Y** | **N** |  | | | |  | | |
| ***Other specs*** | |  | **X** | Other core specifications | | | | TS/TR ... CR ... | | |
| ***affected:*** | |  | **X** | Test specifications | | | | TS/TR ... CR ... | | |
| ***(show related CRs)*** | |  | **X** | O&M Specifications | | | | TS/TR ... CR ... | | |
|  | |  | | | | | | | | |
| ***Other comments:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
| ***This CR's revision history:*** | |  | | | | | | | | |

\*\*\*\*\*\*\*\*\*\*\*\*\* start of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

##### 4.2.2.*2*.2 Protection at the transport layer

*Requirement Name:* Protection at the transport layer

*Requirement Reference:* TS 33.501 [10], clause 5.9.2.1, clause 13.1, clause 13.3.2

*Requirement Description*:

NF Service Request and Response procedure supports mutual authentication between NF consumer and NF producer as specified in TS 33.501 [10], clause 5.9.2.1 .

All network functions support TLS. Network functions support both server-side and client-side certificates. The TLS profile follows the profile given in Annex E of TS 33.310 [9] with the restriction to be compliant with the profile given by HTTP/2 as defined in RFC 7540 [11] as specified in TS 33.501 [10], clause 13.1.

Authentication between network functions within one PLMN uses one of the following methods:

- If the PLMN uses protection at the transport layer as described in clause 13.1, authentication provided by the transport layer protection solution is used for authentication between NFs as specified in TS 33.501 [10], clause 13.3.2.

NOTE 1: This test case only applies to service-based interfaces.

*Threat References*: TR 33.926 [4], clause 5.3.6.3, Weak cryptographic algorithms

*Test case*:

**Test Name:** TC\_PROTECT\_TRANSPORT\_LAYER

**Purpose:**

Verify that TLS protocol for NF mutual authentication and NF transport layer protection is implemented in the network products based on the profile required.

**Procedure and execution steps:**

**Pre-Conditions:**

Network product documentation containing information about supported TLS protocol and certificates is provided by the vendor.

A peer implementing the TLS protocol configured by the vendor is available.

The tester bases the tests on the profile defined by 3GPP in Annex E of TS 33.310 [9] with the restriction that it is compliant with the profile given by HTTP/2 as defined in RFC 7540 [11].

**Execution Steps**

1. The tester checks that compliance with the TLS profile can be inferred from detailed provisions in the network product documentation.

2. The tester eestablishes a secure connection between the network product under test and the peer and verify that all TLS protocol versions and combinations of cryptographic algorithms that are mandated by the TLS profile are supported by the network product under test.

3. The tester trries to establish a secure connection between the network product under test and the peer and verify that this is not possible when the peer only offers a feature, including protocol version and combination of cryptographic algorithms, that is forbidden by the TLS profile.

**Expected Results:**

- The network product under test and the peer establishes TLS if the TLS profiles used by the peer are compliant with the profile requirements in TS 33.310 [9] Annex E and RFC 7540 [11].

- The network product under test and the peer fail to establish TLS if the TLS profiles used by the peer are forbidden in TS 33.310 [9] Annex E or RFC 7540 [11].

**Expected format of evidence:**

Provide evidence of the check of the product documentation in plain text. Save the logs and the communication flow in a .pcap file.

\*\*\*\*\*\*\*\*\*\*\*\*\* 2nd of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

###### 4.2.2.*2*.3.1 Authorization token verification failure handling within one PLMN

*Requirement Name*: Authorization token verification failure handling within one PLMN

*Requirement Reference:* TS 33.501 [10], clause 13.4.1.1

*Requirement Description*:

According to TS 33.501 [10], clause 13.4.1.1, the NF Service producer verifies 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 verifies the claims in the access token as follows:

NOTE: Void.

- 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 checks that it serves the corresponding slice(s).

- If an NF Set ID present, the NF Service Producer checks 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 executes the requested service and responds back to the NF Service consumer. Otherwise it replies based on Oauth 2.0 error response defined in RFC 6749 [12]. The NF service consumer optionally stores 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.

*Threat References*: TR 33.926 [4], clause 6.3.3.1, Incorrect Verification of Access Tokens

*Test Case*:

**Test Name:** TC\_AUTHORIZATION\_TOKEN\_VERIFICATION\_FAILURE\_ONE\_PLMN

**Purpose:**

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.

**Procedure and execution steps:**

**Pre-Conditions:**

- The tester shall know if the network product supports the following optional access token verification claims. If an optional claim is not supported, the associated sub-test case does not apply:

- S-NSSAI (Test Case F)

- NSI (Test Case G)

- NF Set ID (Test Case H)

- additional scope (Test Case I)

- Test environment with an 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 has 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.

**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 Cases A1~E4 are tests on failure handling by the network product under test when the mandatory claims in access token failed verification.

Test Case A: No access token

1) The tester sends a request without a token to the network product under test.2) The network product under test recognized the absence of the access token and the verification of the access token fails.

Test Case B: Verification failure of the access token integrity

1) 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.

2) The integrity verification of the access token by the network product under test fails.

Test Case C: Incorrect audience claim in the access token

1) 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.

2) The network product under test verifies that the integrity of the access token is valid. However, the audience claim in the access token does not match its identity or type.

Test Case D: Incorrect scope claim in the access token

1) 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.

2) The network product under test verifies that the integrity of the access token and the audience claim are valid. However, the scope does not match the requested service operation.

Test Case E: Expired access token

1) 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.

2) The network product under test verifies that the integrity of the access token, the audience and scope claims are all valid. However, the expiration time in the access token has expired against the current data/time.

Test Cases F~I are tests on failure handling by the network product under test when the optional claims in access token failed verification.

NOTE: The test cases below only apply to the NFs which support identifying and understanding the optioanl claims in the received access token.

Test Case F: Incorrect list of S-NSSAIs in the access token

1) The tester computes an access token correctly, except that the list of S-NSSAIs is incorrect, i.e., the network product under test does not serve the slices indicated in the list of S-NSSAIs, and then includes the access token in the NF Service Request sent from NF service consumer to the network product under test.

2) The network product under test verifies that the integrity of the access token, the audience, scope and expiration time claims are all valid. Then it further checks the list of S--NSSAIs included in the access token.

Test Case G: Incorrect list of NSIs in the access token

1) The tester computes an access token correctly, except that the list of NSIs is incorrect, i.e., the network product under test does not serve the slices indicated in the list of NSIs, and then includes the access token in the NF Service Request sent from NF service consumer to the network product under test.

2) The network product under test verifies that the integrity of the access token, the audience, scope and expiration time claims are all valid. Then it further checks the list of NSIs included in the access token.

Test Case H: Incorrect NF Set ID in the access token

1) The tester computes an access token correctly, except that the NF Set ID is incorrect, i.e. the NF Set ID in the claim does not match the NF Set ID 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.

2) The network product under test verifies that the integrity of the access token, the audience, scope and expiration time claims are all valid. Then it further checks the NF Set ID included in the access token.

Test Case I: Incorrect additional scope in the access token

1) The tester computes an access token correctly, except that the additional scope information is incorrect, i.e. the allowed resources and allowed actions on the resources do not match the requested service operations, and then includes the access token in the NF Service Request sent from the NF service consumer to the network product under test.

2) The network product under test verifies that the integrity of the access token, the audience, scope and expiration time claims are all valid. Then it further checks the additional scope included in the access token.

**Expected Results:**

For test cases A~E on verification failure of mandatory claims in the access token, the network product under test rejects the NF service consumer’s service request based on OAuth 2.0 error response defined in RFC 6749 [12].

For test cases F~I on verification failure of optional claims in the access token, if the network product under test understands these optional claims (list of S-NSSAIs, list of NSIs, NF Set ID, additional scope), it rejects the NF service consumer’s service request based on OAuth 2.0 error response defined in RFC 6749 [12].

**Expected format of evidence:**

Evidence suitable for the interface, e.g., S packet trace (pcap file).

\*\*\*\*\*\*\*\*\*\*\*\*\* 3rd of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

###### 4.2.3.4.1.1 Successful authentication and authorization of system functions

*Requirement Name:* Authentication and authorization forSystem functions

*Requirement Reference: In accordance with industry best practice*

*Requirement Description*:

The usage of a system function without successful authentication on basis of the user identity and at least one authentication attribute (e.g. password, certificate) shall be prevented. 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. An exception to the authentication and authorization requirement are functions for public use such as those for a Web server on the Internet, via which information is made available to the public.

*Threat References:* TR 33.926 [4], clause 5.3.6, Information disclosure

*Test case*:

**Test Name**: TC\_SYS\_FUN\_USAGE

**Purpose:**

To ensure that system functions shall not be used without successful authentication and authorization.

**Procedure and execution steps:**

**Pre-Conditions:**

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

2. The vendor shall supply the list of access entries for system functions.

**Execution Steps**

The tester is required to execute the following steps:

1. The tester verifies that the access entries to use system functions, which are listed by the vendor, require successful authentication on basis of the user name and at least one authentication attribute. The tester also verifies that the access entries to use system functions require authorization via an access control mechanism (e.g. Discretionary access control/Ownership/Capabilities or Mandatory access control). This applies to both system functions that are locally accessible and those that are remotely accessible via a network interface.

**Expected Results:**

1. The network product does not allow access to any system function provided by the vendor without a successful user authentication and authorization.

**Expected format of evidence:**

- Description of executed tests and commands

- Relevant output (e.g. Screenshot)

\*\*\*\*\*\*\*\*\*\*\*\*\* 4th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

##### 4.2.3.5.1 Protecting sessions – logout function

*Requirement Name*: Protecting sessions – logout function

*Requirement References*: In accordance with industry best practice

*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 log out. 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.

*Threat References*: TR 33.926 [4] , clause 5.3.6, Information disclosure

**Test Name**: TC\_PROTECTING\_SESSION\_LOGOUT

**Purpose:**

To ensure a signed in user can logout at any time.

**Procedure and execution steps:**

**Pre-Conditions:**

- The vendor 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.

**Execution Steps:**

The tester is required to execute the following steps:

1) The tester creates a new account.

2) The tester uses the new account or an existing account to log into network product. After x minutes the tester tries to logout network product.

NOTE: The value of x can be arbitrarily set by the tester.

**Expected Results:**

- The tester can use a new account or an existing account to log into network product and logout network product after x minutes.

**Expected format of evidence:**

- Settings, and configurations used

\*\*\*\*\*\*\*\*\*\*\*\*\* 5th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

##### 4.2.3.5.2 Protecting sessions – Inactivity timeout

*Requirement Name*: Protecting sessions – inactivity timeout

*Requirement Reference:* In accordance with industry best practice

*Requirement Description:* An O&M user interactive session shall be terminated automatically after a specified period of inactivity. It shall be possible to configure an inactivity time-out period.

Note: The kind of activity required to reset the timeout timer depends on the type of user session.

*Threat References*: TR 33.926 [4], clause 5.3.6, Information disclosure

**Test Name:** TC\_PROTECTING\_SESSION\_ INAC TIMEOUT

**Purpose:**

To ensure an O&M user interactive session shall be terminated at inactivity timeout.

**Procedure and execution steps:**

**Pre-Conditions:**

- The tester has privileges to create an O&M user interactive session.

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

- Session log should be enabled.

**Execution Steps**

1. The tester creates O&M 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 O&M user A interaction session has been terminated automatically.

**Expected Results:**

- In step 3, O&M user A interaction session has been terminated automatically after x minute.

**Expected format of evidence:**

- Session log

- Settings, protocols and configurations used

\*\*\*\*\*\*\*\*\*\*\*\*\* 6th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

##### 4.2.3.6.1 Security event logging

*Requirement Name*: Security event logging

*Requirement Reference:* In accordance with industry best practice

*Requirement Description*: Security events shall be 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.

IETF RFC 3871 [3], section 2.11.10 specifies the minimum set of security events. Each vendor shall document what security events the product logs so that it can be verified by testing.

In particular, it shall be possible to log the following events (which are intended to be supported by the network product and which can be enabled by default at manufacturing time or at a later time by the network operator):

|  |  |  |
| --- | --- | --- |
| EventTypes | Description | Event data to be logged |
| Incorrect login attempts | Records any user incorrect login attempts to the network product | • Username,  • Source (IP address) if remote access  • Timestamp |
| Administrator access | Records any access attempts to accounts that have system privileges. | • Username,  • Timestamp,  • Length of session,  • Source (IP address) if remote access |
| Account administration | Records all account administration activity, i.e. configure, delete, enable, and disable. | • Administrator username,  • Administered account,  • Activity performed (configure, delete, enable and disable)  • Timestamp |
|  |  |  |
| Resource Usage | 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.)  • Timestamp |
| Configuration change | Changes to configuration of the network device | • Change made  • Username |
| Reboot/shutdown/crash | This event records any action on the network device that forces a reboot or shutdown OR where the network device has crashed. | • Action performed (reboot, shutdown, etc.)  • Username (for intentional actions)  • Timestamp |
| Interface status change | Change to the status of interfaces on the network device (e.g. shutdown) | • Interface name and type  • Status (shutdown, missing link, etc.)  • Timestamp |

In addition, optionally it shall be possible to log also the following event (if supported):

|  |  |  |
| --- | --- | --- |
| EventTypes | Description | Event data to be logged |
| Change of group membership or accounts | Any change of group membership for accounts | • Administrator username,  • Administered account,  • Activity performed (group added or removed)  • Timestamp. |

*Threat References:* TR 33.926 [4], clause 5.3.4.4, Log Tampering

*Test case*:

***Test Name*:** TC\_SECURITY\_EVENT\_LOGGING

**Purpose:**

To verify that the network product correctly logs all required security event types.

**Procedure and execution steps:**

**Pre-Conditions:**

- 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 in clause 4.1 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.

**Execution Steps**

For each O&M service perform the following test steps

- The Tester sequentially triggers each security event listed in the requirement, while covering each option detailed in the individual security event descriptions.

- 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.

**Expected Results:**

All security events are appropriately logged, including all required event data.

**Expected format of evidence:**

- List of O&M services

- Commands executed per O&M services

- The relevant parts of the logs in appropriate form (e.g. file, screenshot)

\*\*\*\*\*\*\*\*\*\*\*\*\* 7th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

##### 4.2.3.6.2 Log transfer to centralized storage

*Requirement Name*: Log transfer to centralized storage

*Requirement Reference:* In accordance with industry best practice

*Requirement Description*:

a) The Network Product shall support forwarding of security event logging data to an external system. Secure transport protocols in accordance with clause 4.2.3.2.4, shall be used.

b) Log functions should support secure uploading of log files to a central location or to an external system for the Network Product that is logging.

*Threat References:* TR 33.926 [4], clause 5.3.4.4, Log Tampering

**Test Name**: TC\_LOG TRANS\_TO\_CENTR STORAGE

**Purpose:**

To ensure log shall be transferred to centralized storage.

**Procedure and execution steps:**

**Pre-Conditions:**

- The vendor shall list the protocols which transfer security event logging data (in accordance with clause 4.2.3.2.4).

- 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.

**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 (in accordance with clause 4.2.3.2.4).

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 protocol (in accordance with clause 4.2.3.2.4).

6. The tester checks whether the central location or external system for network product log functions has stored the related logs.

**Expected Results:**

- 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.

**Expected format of evidence:**

- Settings, protocols and configurations used,

- Screenshot

\*\*\*\*\*\*\*\*\*\*\*\*\* 8th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

##### 4.2.3.6.3 Protection of security event log files

*Requirement Name*: Protection of security event log files

*Requirement References:* In accordance with industry best practice

*Requirement Description*: The security event log shall be access controlled (file access rights) so only privileged users have access to the log files.

*Threat References*: TR 33.926 [4], clause 5.3.6.12, Log Disclosure

*Test case*:

**Purpose:**

Verify that the log(s) is(are) only accessible by privileged user(s).

**Procedure and execution steps:**

**Pre-Conditions:**

- Documentation describing where logs are stored and how these logs are accessed and the Network Product interfaces that these logs can be access from.

**Execution Steps**

1. The tester attempts to access log files using users accounts with and without the correct permissions for accessing log files.

2. Repeat the test as described in step 1 using each of the interfaces as described in the Network Product documentation.

**Expected Results:**

The tester checks that log files are accessible when a user with the appropriate authorisation attempts to access them and fails when a user without the correct permissions attempts to access them

**Expected format of evidence:**

Evidence containing the operational results as, e.g. screenshots, log files, packet captures, error messages.

\*\*\*\*\*\*\*\*\*\*\*\*\* 9th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

###### 4.2.4.1.1.2 Handling of ICMP

*Requirement Name*: Processing of ICMPv4 and ICMPv6 packets

*Requirement Reference:* In accordance with industry best practice

*Requirement Description*:

Processing of ICMPv4 and ICMPv6 packets which are not required for operation shall be disabled on the network product. In particular, there are certain types of ICMP4 and ICMPv6 that are not used in most networks, but represent a risk.

ICMP message types which on receipt lead to responses or to configuration changes are not mentioned in this requirement, but they may be necessary to support relevant and specified networking features. Those shall be documented.

Certain ICMP types are generally permitted and do not need to be specifically documented. Those are marked as "Permitted" in below table.

The network product shall not send certain ICMP types by default, but it may support the option to enable utilization of these types (e.g. for debugging). This is 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 | Neigbor Solicitation | Permitted | Permitted |
| N/A | 136 | Neighbor Advertisement | Permitted | N/A |

The network product shall not respond to, or process (i.e. do changes to configuration), under any circumstances certain ICMP message types as marked in 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 |

*Threat Reference*: TR 33.926 [4], clause 5.3.7, Denial of service

*Test Case*:

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 actually provides functionality equivalent to the steps described below.

**Test Name:** TC\_HANDLING\_OF\_ICMP

**Purpose:**

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

**Procedure and execution steps:**

**Pre-Conditions:**

- The vendor provides documentation 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.

**Execution Steps**

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 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 that

- the messages are dropped on receipt by the network product (e.g. by means of appropriate firewall rules),

- or the network product's applicable system configuration remains unchanged upon receipt of the messages,

- or there are other means ensuring that the ICMP messages cannot lead to configuration changes.

The tester verifies consistency between the documentation in regard to ICMP and the network product.

**Expected Results:**

The ICMP messages which are "Not Permitted" 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.

**Expected format of evidence:**

- Tools used and their configuration

- Tool output

\*\*\*\*\*\*\*\*\*\*\*\*\* 10th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

###### 4.2.4.1.1.3 Handling of IP options and extensions

*Requirement Name*: Handling of IP options and extensions

*Requirement Reference:* In accordance with industry best practice

*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.

*Threat References*: TR 33.926 [4], clause 5.7.3, Denial of service

*Test Case*:

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 actually provides functionality equivalent to the steps described below.

**Test Name**: TC\_HANDLING-IP-OPTIONS-AND-EXTENSIONS

**Purpose:** To verify that the network product provides functionality to filter out IP packets with unnecessary options or extension headers in reference to RFC 7126 [20].

**Procedure and execution steps:**

**Pre-Conditions:**

- The vendor 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 are skipped**.**

- A network traffic analyser on the network product (e.g. TCPDUMP) 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).

**Execution Steps**

1. The tester logs in the network product.

2. The tester configures on the network product a filtering rule to drop all IP packets containing an IP Option set

a) The tester establishes an O&M session on if1 interface

b) Using the tool (e.g. Scapy) the tester sends from the tester machine an IPv4 TCP SYN packet with an appropriate destination portto if1 interface without setting any IP Options

c) 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.

d) 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

e) 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.

3. 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. Skip Step 3 if the network product does not support IPv6.

a) 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

b) 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.

c) 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

d) 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.

**Expected Results:**

The network product discards IPv4 packets with unnecessary options in reference to RFC 7126 [20] or IPv6 packets (assuming the network product supports IPv6) with extension header.

**Expected format of evidence:**

- Used tools and their configurations

- Settings and configurations used

- Pcap trace

- Screenshot

\*\*\*\*\*\*\*\*\*\*\*\*\* 11th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

###### 4.2.4.1.2.1 Authenticated Privilege Escalation only

*Requirement Name*: Authenticated Privilege Escalation only.

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*:

There shall not be a privilege escalation method in interactive sessions (CLI or GUI) which allows a user to gain administrator/root privileges from another user account without re-authentication.. Implementation example: Disable insecure privilege escalation methods so that users are required to (re-)login directly into the account with the required permissions.

*Threat References*: TR 33.926 [4], clause 5.3.8.7, Elevation of Privilege via Unnecessary Network Services

*Test Case*:

**Test Name**: TC\_OS\_PRIVILEGE

**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.

**Procedure and execution steps:**

**Pre-Conditions:**

1. The vendor shall provide documentation of the operating system(s) used in the network product.

2. The vendor 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 or used Linux® capabilities.

3. The vendor 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 or Linux® capabilities;

- capabilities of the aforementioned files.

NOTE: Linux® capabilities can provide a subset of root user privileges to a process rather than granting total root access. Some capabilities can be used for privilege escalation

**Execution Steps**

The tester is required to execute the following steps:

1. The tester logs into the network product and verifies that list "A" matches the vendor provided documentation.

2. 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.

3. The tester logs into the network product and verifies that list "B" is accurate based on the vendor provided documentation mentioned in the pre-conditions in this clause. Unix® example: To list files with SUID and SGID permissions and Linux® capabilities, 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

Capabilities: getcap -r / 2>/dev/null

4. The tester verifies that file entries in the list "B" do not have write permissions for anyone else than the owner.

5. 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.

6. 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.

**Expected Results:**

1. The network product does not allow a user to gain administrator/root privileges from another user account without re-authentication.

2. 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.

3. The system documentation about means for a user to gain administrator/root privileges from another user account accurately describes the network product.

**Expected format of evidence:**

- Documentation provided by the vendor: lists "A" and "B"

- Description of executed tests and commands

- Relevant output (e.g. screenshot or terminal log)

\*\*\*\*\*\*\*\*\*\*\*\*\* 12th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

##### 4.2.5.2.1 Webserver logging

*Requirement Name*: Webserver logging

*Requirement Reference:* In accordance with industry best practice

*Requirement Description*: Access to the webserver shall be logged. The web server log shall contain the following information:

- Access timestamp

- Source (IP address)

- (Optional) Account (if known)

- (Optional) 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

*Threat References*: TR 33.926 [4], clause 5.3.4.4, Log Tampering

*Test case*:

**Test Name**: **TC\_WEBSERVER\_LOGGING**

**Purpose:**

Verify that all accesses to the webserver are logged with the required information.

**Procedure and execution steps:**

**Pre-Condition:**

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.

**Execution Steps**

**Execute the following steps:**

1. The tester tries to login to the webserver using the correct and incorrect login credentials.

2. The tester verifies whether the login attempts were logged correctly with all of the required information.

**Expected Results:**

All webserver events are logged with all of the required information.

**Expected format of evidence:**

Log file showing the captured information.

\*\*\*\*\*\*\*\*\*\*\*\*\* 13th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

##### 4.2.6.2.3 GTP-C Filtering

*Requirement Name:* GTP-C Filtering

*Requirement Reference*: In accordance with industry best practice

*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.

NOTE 1: The check could be performed e.g. against an allow list or disallow list of permitted message type / sender identity combinations.

- 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:

- The Network Product supports the capability described above and this is stated in the product documentation.

- The Network Product's product documentation states that the capability is not supported and that the Network Product needs to be deployed together with a separate entity which provides the capability described above.

NOTE 2: Such a separate entity could e.g. be a GTP Firewall.

NOTE 3: Test cases for this separate entity are not provided in the present document, but are believed to be similar to them.

NOTE 4: The test cases are only applicable to all network product classes utilizing GTP-C based protocol.

*Threat References:* TR 33.926 [4], clause 5.3.7, Denial of service

*Test case*:

The test case described here apply only when GTP-C filtering is provided on the Network Product itself.

**Test Name**: TC\_GTP-C\_FILTERING

**Purpose:**

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.

**Procedure and execution steps:**

**Pre-Conditions:**

- The network product has at least two physical interfaces, named if1 and if2.

- The tester has the privileges to configure GTP-C filtering on the network product.

- The vendor declares that the GTP-C filtering is supported.

- The vendor 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.

**Execution Steps**

1. The tester log in the network product.

2. The tester configures the network product with the following rules:

a) Accept only GTP-C EchoRequest messages on if1.

b) Discard all GTP-C messages on if2.

c) For each rule above the accounting is also enabled.

3. The tester turns on the network traffic analyser on if2.

4. The tester sends on if2 EchoRequest messages replaying a pcap file or using a network generator.

a) Using the network analyser the tester verifies that the network product correctly receives the EchoRequest messages on if2.

b) Using the accounting, the tester verifies that the messages are discarded and that any response is sent back by the network product.

5. The tester sends to if1 EchoRequest messages replaying a pcap file or using a network generator.

a) Using the network analyser, the tester verifies that the messages are correctly received by the network product.

b) The tester verifies that the GTP-C EchoRequest messages are not discarded because EchoResponse messages are sent back by the network product.

6. The tester verifies that the matching messages are correctly accounted for both rules.

7. The tester sends to if1 GTP-C messages different from EchoRequest replaying a pcap file or using a network generator.

a) Using the network analyser, the tester verifies that the messages are correctly received by the network product.

b) Using the accounting, the tester verifies that the messages are discarded and that any response is sent back by the network product.

8. The tester deletes the previous rules and configures a new rule, i.e. to accept only GTP-C EchoRequest on if1 coming from a certain IP Address named IP1.

9. The tester sends GTP-C EchoRequest messages with source IP Address set to IP1:

a) Using the network analyser, the tester verifies that the messages are correctly received by the network product.

b) The tester verifies that the GTP-C EchoRequest messages are not discarded and EchoResponse messages are sent back by the network product.

10. The tester sends GTP-C EchoRequest messages with source IP Address set to IP2 different from IP1 using a network traffic generator or replaying a pcap file.

a) Using the network analyser the tester verifies that the messages are correctly received by the network product.

b) The tester verifies that the GTP-C EchoRequest messages are discarded and that no EchoResponse messages are sent back.

**Expected Results:**

- For steps 4, 5, 6 and 7 the tester receives GTP-C EchoResponse messages from if1 only.

- For steps 4, 5, 6 and 7 the messages matching the rules are correctly accounted.

- For steps 8, 9, 10 the tester receives GTP-C EchoResponse messages only for the authorized source IP address.

**Expected format of evidence:**

- The used tool(s) name and version information

- Settings and configurations used

- Pcap trace

- Screenshot

\*\*\*\*\*\*\*\*\*\*\*\*\* 14th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.2.6.2.4 GTP-U Filtering

*Requirement Name:* GTP-U Filtering

*Requirement Reference:* In accordance with industry best practice

*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.

NOTE 1: The check could be performed e.g. against an allow list or disallow list of permitted message type / sender identity combinations.

- 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:

- The Network Product supports the capability described above and this is stated in the product documentation.

- The Network Product's product documentation states that the capability is not supported and that the Network Product needs to be deployed together with a separate entity which provides the capability described above.

NOTE 2: Such a separate entity could e.g. be a GTP Firewall.

NOTE 3: Test cases for this separate entity are not provided in the present document, but are believed to be similar to them.

NOTE 4: The test cases are only applicable to all network product classes utilizing GTP-U based protocol.

*Threat References:* TR 33.926 [4] clause 5.3.7, Denial of service

*Test case*:

The test case described here apply only when GTP-U filtering is provided on the Network Product itself.

**Test Name**: TC\_GTP-U\_FILTERING

**Purpose:**

To verify that the network product provides filtering functionalities for incoming GTP-U messages. In particular this test case verifies that:

1. The network product provides filtering of incoming GTP-U messages on any interface.

2. It is possible to block all GTP-U messages on those network product interfaces where they are unwanted.

3. It is possible to specify defined actions for each rule.

**Procedure and execution steps:**

**Pre-Conditions:**

- 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 vendor declares that the GTP-U filtering is supported.

- The vendor 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.

NOTE: If the network product has only one physical interface named if1, execution steps on if2 are not needed.

**Execution Steps**

1. The tester log in the network product.

2. The tester configures the network product with the following rules:

a) Accept only GTP-U EchoRequest messages on if1.

b) Discard all GTP-U messages on if2.

c) For each rule above the accounting is also enabled.

3. The tester turns on the network traffic analyser on if2.

4. The tester sends on if2 EchoRequest messages replaying a pcap file or using a network generator.

a) Using the network analyser the tester verifies that the network product correctly receives the EchoRequest messages on if2.

b) Using the accounting, the tester verifies that the messages are discarded and that any response is sent back by the network product.

5. The tester sends to if1 EchoRequest messages replaying a pcap file or using a network generator.

a) Using the network analyser, the tester verifies that the messages are correctly received by the network product.

b) The tester verifies that the GTP-U EchoRequest messages are not discarded because EchoResponse messages are sent back by the network product.

6. The tester verifies that the matching messages are correctly accounted for both rules.

7. The tester sends to if1 GTP-U messages different from EchoRequest replaying a pcap file or using a network generator.

a) Using the network analyser, the tester verifies that the messages are correctly received by the network product.

b) Using the accounting, the tester verifies that the messages are discarded and that any response is sent back by the network product.

8. The tester deletes the previous rules and configures a new rule, i.e. to accept only GTP-U EchoRequest on if1 coming from a certain IP Address named IP1.

9. The tester sends GTP-U EchoRequest messages with source IP Address set to IP1:

a) Using the network analyser, the tester verifies that the messages are correctly received by the network product.

b) The tester verifies that the GTP-U EchoRequest messages are not discarded and EchoResponse messages are sent back by the network product.

10. The tester sends GTP-U EchoRequest messages with source IP Address set to IP2 different from IP1 using a network traffic generator or replaying a pcap file.

a) Using the network analyser the tester verifies that the messages are correctly received by the network product.

b) The tester verifies that the GTP-U EchoRequest messages are discarded and that no EchoResponse messages are sent back.

**Expected Results:**

- For steps 4, 5, 6 and 7 the tester receives GTP-U EchoResponse messages from if1 only.

- For steps 4, 5, 6 and 7 the messages matching the rules are correctly accounted.

- For steps 8, 9, 10 the tester receives GTP-U EchoResponse messages only for the authorized source IP address.

**Expected format of evidence:**

- The used tool(s) name and version information

- Settings and configurations used

- Pcap trace

- Screenshot

\*\*\*\*\*\*\*\*\*\*\*\*\* 15th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.2.1 No unnecessary or insecure services / protocols

*Requirement Name*: No unnecessary or insecure services / protocols

*Requirement Reference:* In accordance with industry best practice

*Requirement Description*:

The network product shall only run protocol handlers and services which are needed for its operation, and which do not have any known security vulnerabilities. In particular, by default the following services shall be initially configured to be disabled on the network product by the vendor except if services are needed during deployment. In that case those services shall be disabled according to vendor’s instructions after deployment is done. Disabled protocols can still be enabled for other reasons by the network operators, e.g. remote diagnostics.

- FTP

- TFTP

- Telnet

- rlogin, RCP, RSH

- HTTP

- SNMPv1 and v2

- SSHv1

- TCP/UDP Small Servers (Echo, Chargen, Discard and Daytime)

- Finger

- BOOTP server

- Discovery protocols (CDP, LLDP)

- IP Identification Service (Identd)

- PAD

- MOP

NOTE 1: As an alternative to disabling the HTTP service, it is also possible for this service to remain active for reasons of user friendliness. In this case, however, queries to the web service are not answered directly on this port but from a redirected to HTTPS service.

Note 2: Full documentation of required protocols and services of the network product and their purpose needs to be provided by the vendor as prerequisite for the test case.

*Threat References:* TR 33.926 [4], clause 5.3.7.3, Insecure Network Services

*Test Case*:

**Test Name**: TC\_NO\_UNNECESSARY\_SERVICE

**Purpose:**

To ensure that on all network interfaces, there are no unsecure services or protocols that might be running.

**Procedure and execution steps:**

**Pre-Conditions:**

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;

- and a description of their purposes.

The tool used shall be capable to detect and identify the protocol handlers and running services in the system.

**Execution Steps**

The tester is required to execute the following steps:

1. Verification of the compliance to the prerequisites:

a. Verification that the list of available network services and protocol handlers is available in the documentation of the Network Product.

b. Validation that all entries in the list are necessary for the operation of the Network Product class.

2. Identification of the network services and protocol handlers by means of tools or any other testing means.

3. Validation that there are no entries in the list of network services and handlers apart from the ones that have been mentioned for the operation of the Network Product in the attached documentation.

4. The tester shall reboot the network product and re-execute execution steps 2 and 3 without further configuration.

**Expected Results:**

The report will contain:

- The names and version of the tool(s) used.

- Information of all the protocol handlers and services running in the network product.

Result will show:

- There are no unnecessary services running in the network product except for the ones which are necessary for its operation.

- Any undocumented services running on the network product should be highlighted and brought out in the report.

- The network product behaves the same after reboot as before.

**Expected format of evidence:**

- The used tool(s) name and version information;

- Settings and configurations, and commands used (if applicable);

- The output pertaining to the test case performed and

- The test results i.e. services existing or not existing in the Network Product.

\*\*\*\*\*\*\*\*\*\*\*\*\* 16th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.2.2 Restricted reachability of services

*Requirement Name*: Restricted reachability of services

*Requirement Reference:* In accordance with industry best practice

*Requirement Description*:

The network product shall restrict the reachability of services so that they can only be reached on interfaces where their usage is required. On interfaces were services are active, the reachability should be limited to legitimate communication peers. This limitation shall be realized on the network product itself (without measures (e.g. firewall) at network side) according to the requirement detailed in clause 4.2.6.2.1 Packet Filtering.

Example: Administrative services (e.g. SSH, HTTPS, RDP) shall be restricted to interfaces in the management network to support separation of management traffic from user traffic.

*Threat References*: TR 33.926 [4], clause 5.3.7.3, Insecure Network Services

*Test Case*:

**Test Name***:* TC\_RESTRICTED\_ REACHABILITY \_OF\_SERVICES

**Purpose:**

To verify that it is possible to bind the services only to the interfaces from which they are expected to be reachable.

Note: The test case developed for the requirement " 4.2.6.2.1 Packet Filtering" implicitly verifies that the network product permits to limit the reachability of the services only to legitimate communication peers,

**Procedure and execution steps:**

**Pre-Conditions:**

- 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.

**Execution Steps**

**For every available interface if\_n:**

1. The tester runs a network port scanner (e.g. nmap) or uses local network interface information on if\_n and verifies that the configured services (according to the vendor documentation) are open/reachable.

2. The tester runs a network port scanner (e.g. nmap) or uses local network interface information on all other available interfaces (except if\_n) and verifies that the services configured for if\_n are not open/reachable.

**Expected Results:**

Services can be enabled on per-interface basis.

**Expected format of evidence:**

- The network product configuration showing the mapping between interfaces and configured service.

- Pcap files.

- Screenshot.

- Software name and version of the used port scanner, log of the executed commands.

- Network port scanner results (e.g. files containing this results).

\*\*\*\*\*\*\*\*\*\*\*\*\* 17th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.2.3 No unused software

*Requirement Name*: No unused software

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*:

Unused software components or parts of software which are not needed for operation or functionality of the network product shall not be installed or shall be deleted after installation. This includes also parts of a software, installed as examples but typically not be used (e.g. default web pages, example databases, test data).

*Threat References*: TR 33.926 [4], clause 5.3.6.13, Unnecessary Applications

*Test Case*:

**Test Name**: TC\_NO\_UNUSED\_SOFTWARE

**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.

**Procedure and execution steps:**

**Pre-Conditions:**

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.

**Execution Steps**

1. The tester verifies that the list of software is available in the documentation of the Network Product and is compliant to the prerequisites, e.g. completeness of the information.

2. The tester identifies the software / libraries or components which are installed in the system usingany command line tools or any means of determination.

NOTE 1: The identification of software could be done by, e.g. consulting the package manager of the OS/distribution (e.g. apt, dpkg, rpm, pacman, flatpack, snap…) and package managers of available runtimes (e.g. pip (Python), npm (JavaScript), composer (PHP)…), scanning for executables (global or focused on PATH variable of all available users), scanning for script files related to the available interpreters or listing images and their dependencies when virtualization or containerization is used.

3. The tester validates 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. The tester checks for default configuration or example files mentioned in the software documentation or evident in the file system for the software installed on the system.

**Expected Results:**

The report will contain the names and version of the tool(s) used for finding out what software /libraries is 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.

**Expected format of evidence:**

- 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

\*\*\*\*\*\*\*\*\*\*\*\*\* 18th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.2.4 No unused functions

*Requirement Name*: No unused functions.

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*:

During installation of software and hardware often functions are activated that are not required for operation or function of the system. If unused functions of software cannot be deleted or deinstalled individually as required in clause "4.3.2.3 No unused software" of the present document, such functions shall be deactivated in the configuration of the network product permanently. Also, hardware functions which are not required for operation or function of the system (e.g. unused interfaces) shall be permanently deactivated. Permanently means that they shall not be reactivated again after network product reboot.

NOTE 1: A function within the scope of this test case is a hardware, software or operating system functionality present in the network product under test. It is therefore distinct from the 3GPP defined network function, which the network product provides in accordance with the design objectives from TS 23.501 [18] and TS 33.501 [10]. Examples of functions in the sense of this test are modules used in webservers, debugging functionality or software and hardware interfaces for network communication like Bluetooth ®.

*Threat References:* TR 33.926 [4], clause 5.3.6.13, Unnecessary Applications

*Test Case*:

**Test Name**: TC\_NO\_UNUSED\_FUNCTIONS

**Purpose**:

To ensure that all active hardware functions or software functions are explicitly required for operation or functionality of the network product.

**Procedure and execution steps:**

**Pre-Conditions:**

NOTE 2: If the network product under test is pure software, the hardware aspects of this test case do not apply.

A list of all available software or hardware and associated components containing at least the following information shall be included in the documentation accompanying the Network Product:

- name of the software or hardware;

- version of the software or hardware installed;

- list of dependencies and versions;

- any add-ons and functions;

- any special hardware/debugging ports;

- software or hardware support type;

- licensing information;

- requirement during functioning of system;

- brief description of their purpose.

**Execution Steps:**

1. The tester verifies that the list of hardware functions and software functions is available in the documentation of the Network Product.

2. The tester identifies the hardware functions and software functions which are installed in the system or might have been disabled using any command line tools or any means of determination.

NOTE 3: The identification of software could be done by, e.g. consulting the package manager of the OS/distribution (e.g. apt, dpkg, rpm, pacman, flatpack, snap…) and package managers of available runtimes (e.g. pip (Python), npm (JavaScript), composer (PHP)…), scanning for executables (global or focused on PATH variable of all available users), scanning for script files related to the available interpreters or listing images and their dependencies when virtualization or containerization is used.

NOTE 4: The identification of hardware could be done by, e.g. consulting any type of device manager or hardware information tool (e.g. hwinfo, inxi, lshw, lspci, lscpu, lsusb…).

3. The tester validates that there are no entries in the list of hardware functions 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.

**Expected Results:**

The report will contain the names and version of the tool(s) used for finding out what software functions or hardware functions are installed in the system. The detailed report will contain the name and version information of all the software components or hardware components installed in the system generated by the test tool.

The list of all available software functions and hardware functions 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 functions or hardware function not in the list of allowed software functions or hardware functions 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.

**Expected format of evidence:**

- The used tool(s) name and version information

- Settings and configurations used

- The list of software functions and hardware functions

- the test results i.e. allowed list of functions

\*\*\*\*\*\*\*\*\*\*\*\*\* 19th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.2.5 No unsupported components

*Requirement Name*: No unsupported components.

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*:

The network product shall not contain software and hardware components that are no longer supported by their vendor, manufacturer or developer, such as components that have reached end-of-life or end-of-support. Excluded are components that have a special support contract. This contract shall guarantee the correction of vulnerabilities over components' lifetime.

*Threat References*: TR 33.926 [4], clause 5.3.6.13, Unnecessary Applications

*Test Case:*

**Test Name**: TC\_NO\_UNSUPPORTED\_COMPONENTS

**Purpose:**

To ensure that all software and hardware components running in the network product are still supported and have not reached either their end-of-life or end-of-support.

**Procedure and execution steps:**

**Pre-Conditions:**

NOTE 1: If the network product under test is pure software, the hardware aspects of this test case do not apply.

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 or hardware component;

- 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.

**Execution Steps**

1. The tester identifies the hardware and software components available in the network product, version information and the kind of support available for the software provided by the vendor, the manufacturer, the developer or other contractual partner of the network operator using any tool or any means of determination.

NOTE 2: The identification of software could be done by, e.g. consulting the package manager of the OS/distribution (e.g. apt, dpkg, rpm, pacman, flatpack, snap…) and package managers of available runtimes (e.g. pip (Python), npm (JavaScript), composer (PHP)…), scanning for executables (global or focused on PATH variable of all available users), scanning for script files related to the available interpreters or listing images and their dependencies when virtualization or containerization is used.

NOTE 3: The identification of hardware could be done by, e.g. consulting any type of device manager or hardware information tool (e.g. hwinfo, inxi, lshw, lspci, lscpu, lsusb…).

2. The tester validates 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.

**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.

**Expected format of evidence:**

- 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

\*\*\*\*\*\*\*\*\*\*\*\*\* 20th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.2.6 Remote login restrictions for privileged users

*Requirement Name*: Remote login restrictions for privileged users

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*: Direct login as root or equivalent highest privileged user shall be limited to the system console only. Root user will not be allowed to login to the system remotely.

*Threat References*: TR 33.926 [4], clause 5.3.8.1, Misuse by authorized users

*Test Case*:

**Test Name**: TC\_REMOTE\_LOGIN\_RESTRICTIONS\_PRIVILEGED\_USERS

**Purpose:**

Verify that root or equivalent highest privileged user will not be allowed to login to the system remotely.

**Procedure and execution steps:**

**Pre-Condition:**

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

**Execution Steps**

**Execute the following steps:**

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.

**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.

**Expected format of evidence:**

Evidence containing the operational results as, e.g. screenshots, log files, packet captures, error messages.

\*\*\*\*\*\*\*\*\*\*\*\*\* 21st of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

##### 4.3.3.1.1 IP-Source address spoofing mitigation

*Requirement Name*: IP-Source address spoofing mitigation

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*:

Systems 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.

*Threat References*: TR 33.926 [4], clause 5.3.3.5, IP Spoofing

*Test Case*:

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 actually provides functionality equivalent to the steps described below.

**Test Name:** TC\_IP\_SPOOFING\_MITIGATION

**Purpose:**

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.

**Procedure and execution steps:**

**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.

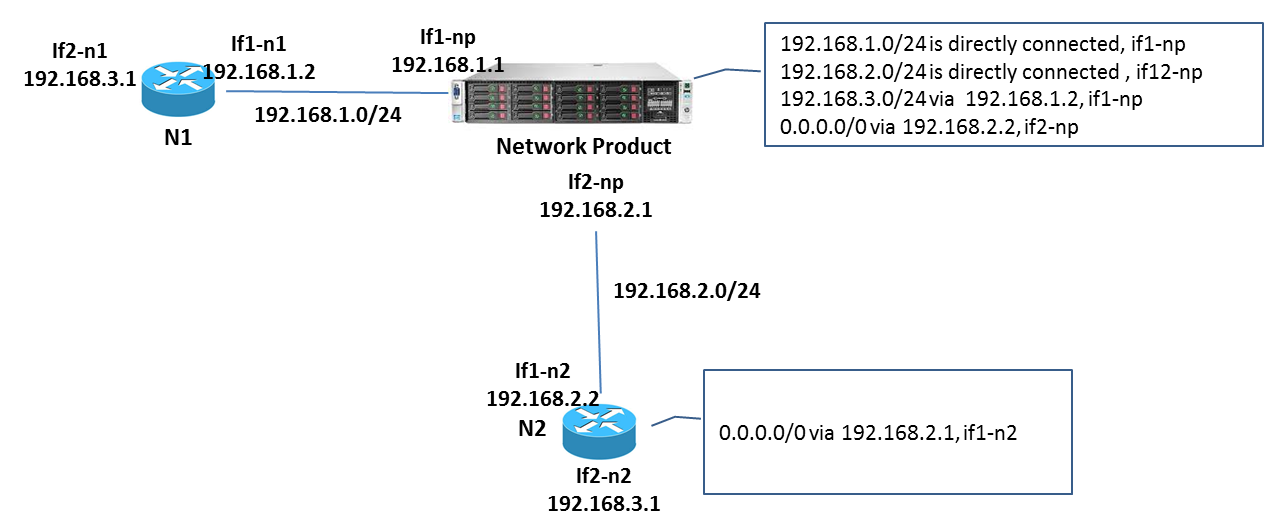


Figure 1: Configurations for the network product, N1 and N2

- 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

**Execution Steps**

1. The tester starts to send ping messages to if1-np interface of the network product.

2. The tester verifies, through the network traffic analyser, that the ping reaches correctly the if1-np interface and that responses are sent back.

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 uses the same network configuration of the tester machine.

- The tester sends ping messages to if1-np interface of the network product.

- 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.

- The tester sends ping messages to if2-np interface of the network product.

- 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.

- If the dropped packets are logged, the testers verifies that these packets are recorded.

**Expected Results:**

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.

**Expected format of evidence:**

- The user settings and configurations

- Pcap files

- Log file if available

\*\*\*\*\*\*\*\*\*\*\*\*\* 22nd of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

##### 4.3.3.1.4 SYN Flood Prevention

*Requirement Name:* Syn Flood Prevention

*Requirement Reference*: In accordance with industry best practice

*Requirement Description:*

The network product shall support a mechanism to prevent Syn Flood attacks (e.g. implement the TCP Syn Cookie technique in the TCP stack by setting net.ipv4.tcp\_syncookies = 1 in the linux sysctl.conf file). This feature shall be enabled by default.

*Threat References*:TR 33.926 [4], clause 5.3.7.2, Implementation Flaw

*Test Case:*

**Test Name**: TC\_SYN\_FLOOD\_PREVENTION

**Purpose:**

Verify that the Network Product supports a Syn Flood Prevention technique.

**Procedure and execution steps:**

**Pre-Conditions:**

- Vendor documentation describing the SYN flood attack prevention mechanism or setting and where to check for them.

- The Network Product is listening on a TCP port on 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)

**Execution Steps**

1. The tester verifies the prevention mechanism or setting described in the vendor documentation.

2. The tester configures the tool to send a large quantity huge amount of TCP Syn packets against the Network Product (e.g. hping3 -i <waiting time between each packet> -S -p <TCP port> -d <Data Size> -c <Number of packets> < Network Product IP>)

NOTE: To calculate the large quantity number of packets the tester checks in the product documentation the link speed supported by the DUT in bytes (L). The tester chooses a size packet for the attack in bytes (S). Based on L and S, the tester calculates the amount of *packets per second* (P) to use with this formula:  
 P = L / S

3. The tester verifies that the Network Product is still functioning as expected, its services are still accessible and responsive to typical service function requests, and the memory or CPU usage does not exceed acceptable thresholds. Additionally, the tester confirms there are no crashes or deadlocks.

a. While the SYN Flood attack is ongoing.

b. After the SYN Flood attack was executed.

**Expected Results:**

The Network Product does not become inoperative.

**Expected format of evidence:**

- Executed commands or script used for the SYN flood attack.

- The number of SYN packets sent per second.

- Part of the configuration (plaintext or screenshot) showing the prevention mechanism or setting.

\*\*\*\*\*\*\*\*\*\*\*\*\* 23rd of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.2 No system privileges for web server

*Requirement Name*: No system privileges for web server.

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*:

No web server processes shall run with system privileges. This is best achieved if the web server runs under an account that has minimum privileges. If a process is started by a user with system privileges, execution shall be transferred to a different user without system privileges after the start.

*Threat References*: TR 33.926 [4], clause 5.3.8, Elevation of privilege

*Test Case*:

***Test Name*:** TC\_NO\_SYSTEM\_PRIVILEGES\_WEB\_SERVER

**Purpose:**

Verify that the Web server is not run under system privileges.

**Procedure and execution steps:**

**Pre-Conditions:**

- The tester has needed administrative privileges.

- A tester machine is available.

- Recommended: an automatic assessment tool has been configured /script adapted in line with the Requirement Description.

**Execution Steps**

1. The tester checks that no web server processes run with system privileges. The tester checks that this is the case even for processes that may have been started by a user with system privileges.

a. The tester starts the web server process as web server user and checks process privileges.

b. If possible, the tester starts the web server process with system privileges and check if process privileges get dropped.

2. The tester checks in relevant system settings and web server configurations that a web server user is configured with minimal privileges needed to run the web server and the web server is executable by that user.

**Expected Results:**

- There are no findings of web server processes that run with system privileges.

- System settings are set to ensure that no processes will run with system privileges.

**Expected format of evidence:**

- Log files / command line output and screen shots of test executions

- Part of web server and/or system configuration (plain text or screenshot) showing the configured user for the web server process

\*\*\*\*\*\*\*\*\*\*\*\*\* 24th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.3 No unused HTTP methods

*Requirement Name*: No unused HTTP methods

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*:

HTTP methods that are not required shall be deactivated. Standard requests to web servers use GET, HEAD, and POST. If other methods are required, e.g, PUT, DELETE, PATCH, they shall not introduce security leaks such as TRACK or TRACE.

*Threat References*: TR 33.926 [4] clause 5.3.6.11, Unnecessary Services

*Test Case*:

***Test Name*:** TC\_NO\_UNUSED\_HTTP\_METHODS

**Purpose:**

Verify that the Web server has deactivated all HTTP methods that are not required.

**Procedure and execution steps**

**Pre-Conditions:**

- The tester has needed administrative privileges.

- A tester machine is available.

- Recommended: an automatic assessment tool has been configured / script adapted in line with the Requirement Description.

**Execution Steps**

- Check that relevant system settings and configurations are in place to ensure fulfilment of the requirement.

**Expected Results:**

- System settings and configurations have been found and in normal operation, for all Web components of the system, to ensure that unneeded HTTP methods are deactivated.

**Expected format of evidence:**

- Log files and screen shots of test executions

\*\*\*\*\*\*\*\*\*\*\*\*\* 25th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.4 No unused add-ons

*Requirement Name*: No unused add-ons

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*: All optional add-ons and components of the web server shall be deactivated if they are not required. In particular, CGI or other scripting components, Server Side Includes (SSI), and WebDAV shall be deactivated if they are not required.

*Threat References*: TR 33.926 [4], clause 5.3.6.11, Unnecessary Services

*Test Case*:

***Test Name*:** TC\_NO\_UNUSED\_ADD-ONS

**Purpose:**

To verify that the Web server has deactivated unneeded add-ons and unneeded scripting components.

**Procedure and execution steps**

**Pre-Conditions:**

- 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.

- Recommended: an automatic assessment tool has been configured / script adapted in line with the Requirement Description.

**Execution Steps**

1. 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. See also guidance under 4.3.4.12.

2. Check that nothing else has been installed than the web server.

3. Check that relevant system settings and configurations are correct to ensure fulfilment of the requirement.

**Expected Results:**

- System settings and configurations have been found, for all Web components of the system, to ensure that all unneeded add-ons or script components are deactivated.

**Expected format of evidence:**

- Log files and screen shots of test executions.

\*\*\*\*\*\*\*\*\*\*\*\*\* 26th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.5 No compiler, interpreter, or shell via CGI or other server-side scripting

*Requirement Name*: No compiler, interpreter, or shell via CGI or other server-side scripting.

*Requirement Reference*: In accordance with industry best practice

*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 (e.g. PERL® interpreter, PHP interpreter/compiler, Tcl interpreter/compiler or operating system shells).

*Threat Reference*: TR 33.926 [4], clause 5.3.6, Information disclosure

*Test Case*:

***Test Name*:** TC\_NO\_COMPILER\_FOR\_CGI

**Purpose:**

To verify that there are no compilers, interpreters or shell accessible via CGI or other scripting components.

**Procedure and execution steps**

**Pre-Conditions:**

- The tester has administrative privileges

- A tester machine is available.

- Recommended: an automatic assessment tool has been configured /script adapted in line with the Requirement Description.

**Execution Steps**

1. Consult the web server configuration to identify all directories used for CGI or other scripting components.

2. Check 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).

**Expected Results:**

There are no compilers, interpreters or shells in directories accessible via CGI or other scripting components.

**Expected format of evidence:**

- Log files and screen shots of test executions.

- Part of web server configuration (plaintext or screenshot) showing all directories accessible by the CGI/scripting components.

- List of files (with types and permissions, if available) inside the directories accessible by the CGI/scripting components.

\*\*\*\*\*\*\*\*\*\*\*\*\* 27th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.6 No CGI or other scripting for uploads

*Requirement Name*: No CGI or other scripting for uploads.

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*: If CGI or other scripting technology is used, all directories where the web server has write permissions shall be distinct from all directories containing CGI/script or executable code.

*Threat References*: TR 33.926 [4], clause 5.3.8.3, Folder Write Permission Abuse

*Test Case*:

**Test Name:** TC\_NO\_CGI\_OR\_SCRIPTING\_FOR\_UPLOADS

**Purpose:**

To ensure that directories with write permissions for the web server do not contain executable code such as CGI scripts.

**Procedure and execution steps:**

**Pre-Condition:**

If the web server is configured with CGI/Scripting on, this test applies.

**Execution Steps**

**Execute the following steps:**

1. The tester identifies directories where the web server user has write permissions.

2. The tester verifies that these writable directories do not contain any executable scripts, CGI programs, or other executable code.

3. The tester verifies that directories configured for CGI/Scripting do not have write permissions for the web server.

**Expected Results:**

Web server user writable directories are different from those containing executable code or the ones configured to be used for CGI/scripting.

**Expected format of evidence:**

A part of the configuration file / screenshot of the configuration showing that the web server is properly configured and the corresponding file system permissions.

\*\*\*\*\*\*\*\*\*\*\*\*\* 28th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.8 Access rights for web server configuration

*Requirement Name*: Access rights for web server configuration files

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*: Access rights for web server configuration files shall only be granted to the owner of the web server process or to a user with system privileges. Implementation example: Delete "read" and "write" access rights for "others." Only grant "write" access to the user who configures the web server.

*Threat References*: TR 33.926 [4], clause 5.3.8, Elevation of privilege

*Test Case*:

***Test Name*:** TC\_ACCESS\_RIGHTS\_WEB\_SERVER\_FILES

**Purpose:**

To verify that the access rights for Web server configuration files are correctly set.

**Procedure and execution steps**

**Pre-Conditions:**

- The tester has administrative privileges

- A tester machine is available.

- Recommended: an automatic assessment tool has been configured / script adapted in line with the Requirement Description.

**Execution Steps**

1. The tester identifies the user owning the web server process.

2. The tester verifies that only the owner of the web server process and users with system privileges have "read" and "write" access rights for all web server configuration files and configuration directories.

**Expected Results:**

- Access rights for web server configuration files and directories are adequately set.

**Expected format of evidence:**

- Log files and screen shots of test executions

\*\*\*\*\*\*\*\*\*\*\*\*\* 29th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.9 No default content

*Requirement Name*: No default content.

*Requirement Reference*: In accordance with industry best practice.

*Requirement Description*: Default content (examples, help files, documentation, aliases) that is provided with the standard installation of the web server shall be removed.

*Threat References*: TR 33.926 [4], clause 5.3.6.8, Insecure Default Configuration

*Test Case*:

***Test Name*:** TC\_NO\_DEFAULT\_CONTENT

**Purpose:**

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.

**Procedure and execution steps**

**Pre-Conditions:**

- The tester has needed administrative privileges.

- A tester machine is available.

- The tester should have configured a script, or an automatic assessment tool adapted in line with the Requirement Description.

NOTE: The term ‘default content’ is not clearly defined and is therefore different for different web servers (e.g., web server welcome page, default error page, etc.).

**Execution Steps**

1. The tester checks that all default content (examples, help files, documentation, aliases) that is provided with the standard installation of the web server has been removed.

**Expected Results:**

- No default content (examples, help files, documentation, aliases, un-needed directories or manuals) has been found to remain on any Web server component.

**Expected format of evidence:**

- Log files and screen shots of test executions.

 \*\*\*\*\*\*\*\*\*\*\*\*\* 30th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.10 No directory listings

*Requirement Name*: No directory listings / Directory Browsing.

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*: Directory listings (indexing) / "Directory browsing" shall be deactivated.

*Threat References*: TR 33.926 [4], clause 5.3.6.9, File/Directory Read Permissions Misuse

*Test Case*:

***Test Name*:** TC\_NO\_DIRECTORY\_LISTINGS

**Purpose:**

To verify that Directory listings / Directory browsing has been deactivated in all Web server components.

**Procedure and execution steps**

**Pre-Conditions:**

- The tester has administrative privileges

- A tester machine is available.

- The tester should have configured a script, or an automatic assessment tool adapted in line with the Requirement Description.

**Execution Steps**

- 1. The tester checks the web server configuration for Directory listings (indexing) / "Directory browsing" to be deactivated in all Web server components.

2. The tester attempts directory listings on all endpoints (domains, subdomains and directories) offered by the web server.

NOTE 1: Whether directory listings have been deactivated could be done by checking the webserver configuration file specifically the parameters related to directory listing. The directory listing could be turned off in the web server configuration file, and there is no activation capability.

NOTE 2: Directory listings could be obtained by entering a valid URL (e.g., /var/www/test\_1) that does not contain any index file.

**Expected Results:**

- Directory listing / Directory browsing has been deactivated in all Web server components configurations.

- The tester is unable to perform Directory listing / Directory browsing on all endpoints (domains, subdomains and directories) offered by the web server.

**Expected format of evidence:**

- Log files and screen shots of test executions

- Text excerpt of the web server configuration showing that directory listing is disabled

 \*\*\*\*\*\*\*\*\*\*\*\*\* 31st of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.11 Web server information in HTTP headers

*Requirement Name*: Web server information in HTTP headers.

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*: The HTTP header shall not include information on the version of the web server and the modules/add-ons used.

*Threat References*: TR 33.926 [4], clause 5.3.6.5, System Fingerprinting

*Test Case*:

***Test Name*:** TC\_NO\_WEB\_SERVER\_HEADER\_INFORMATION

**Purpose:**

To verify that HTTP headers do not include information on the version of the web server and the modules/add-ons used.

**Procedure and execution steps**

**Pre-Conditions:**

- The tester has administrative privileges.

- A tester machine is available.

- The tester should have configured a script, or an automatic assessment tool adapted in line with the Requirement Description.

**Execution Steps**

1. The tester checks that HTTP headers do not include information on the version of the web server and the modules/add-ons used.

NOTE 1: The header information could be checked by examining a captured http packet or by observing the response to a manual request (e.g. curl –I <address:port>). Header fields to look for could be, but are not limited to, ‘Server’, ‘X-Powered-By’, ‘Via’ or custom header fields. Furthermore, unwanted web server information could be part of the response body (e.g. in HTML comments or meta tags) or a server banner.

NOTE 2: The settings responsible for limiting the header information could be checked from the webserver configuration file (e.g. Apache configuration has a ‘ServerTokens’ directive, which could be set to ‘Prod’; nginx configuration has a ‘server\_tokens’ directive, which could be set to ‘off’).

**Expected Results:**

- Evidence that HTTP headers do not include information on the version of the web server and the modules/add-ons used.

**Expected format of evidence:**

- Log files and screen shots of test executions.

 \*\*\*\*\*\*\*\*\*\*\*\*\* 32nd of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.12 Web server information in error pages

*Requirement Name*: Web server information in error pages.

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*: User-defined error pages shall not include version information about the web server and the modules/add-ons used. Error messages shall not include internal information such as internal server names, error codes, etc. Default error pages of the web server shall be replaced by error pages defined by the vendor.

*Threat References*: TR 33.926 [4], 5.3.6.5, System Fingerprinting

*Test Case*:

***Test Name*:** TC\_NO\_WEB\_SERVER\_ERROR\_PAGES\_INFORMATION

**Purpose:**

To verify that error pages and error messages do not include information about the web server.

**Procedure and execution steps**

**Pre-Conditions:**

- The tester has needed administrative privileges.

- A tester machine is available.

- The vendor provides documentation on user-defined error pages (e.g. location, content, where configured) and messages.

- The vendor provides a list of potential parameters/commands to trigger events resulting in an http status code 3xx, 4xx, 5xx.

- The tester should have configured a script, or an automatic assessment tool adapted in line with the Requirement Description.

**Execution Steps**

1. The tester verifies that the web server configuration does replace default error pages with error pages defined by the vendor.

2. The tester verifies that the vendor defined error pages do not contain information about the web server.

3. The tester triggers and captures at least one occurrence of the following HTTP status code classes:

a) Redirection error response (300-399)

b) Client error response (400-499)

c) Server error response (500-599)

NOTE 1: Possible error pages that could be displayed are: 3xx: redirection, 4xx: client errors, 5xx: server errors.

NOTE 2: The 3xx error pages could be triggered by permanent or temporary move of content to other URL and the page is found because redirected.

NOTE 3: The 4xx error page could be triggered by trying to access a URL pointing to a non-existent or restricted resource.

NOTE 4: The 5xx error page could be triggered by requesting a HTTP method the web server does not support or disabled (e.g. CONNECT, PUT, PATCH).

**Expected Results:**

- Generated error pages and error messages do not include information about the web server.

**Expected format of evidence:**

- Log files and screen shots of test executions

 \*\*\*\*\*\*\*\*\*\*\*\*\* 33rd of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.4.13 Minimized file type mappings

*Requirement Name*: Minimized file type mappings *Requirement Reference*: In accordance with industry best practice

*Requirement Description*: File type- or script-mappings that are not required shall be deleted, e.g. php, phtml, js, sh, csh, bin, exe, pl, vbe, vbs.

*Threat References*: TR 33.926 [4], clause 5.3.6.13, Unnecessary Applications

*Test Case*:

***Test Name*:** TC\_NO\_WEB\_SERVER\_FILE\_TYPE MAPPINGS

**Purpose:**

To verify that file type- or script-mappings that are not required have been deleted.

**Procedure and execution steps**

**Pre-Conditions:**

- The tester has needed administrative privileges.

- A tester machine is available.

- The tester should have configured a script, or an automatic assessment tool adapted in line with the Requirement Description.

**Execution Steps**

- The tester checks that all file type- or script-mappings that are not required have been deleted.

**Expected Results:**

- Evidence that all file type- or script-mappings, that are not required, have been deleted.

**Expected format of evidence:**

- Log files and screen shots of test executions.

\*\*\*\*\*\*\*\*\*\*\*\*\* 34th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.5.1 Traffic Separation

*Requirement Name*: Traffic Separation

*Requirement Reference*: In accordance with industry best practice

*Requirement Description*:

The network product shall support 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 [3] for further information.

*Threat References*: TR 33.926 [4], clause 5.3.6.15, lack of GNP traffic isolation

*Test case*:

**Test Name:** TC\_TRAFFIC\_SEPARATION

**Purpose:**

To test whether traffic belonging to different network domains is separated.

**Procedure and execution steps:**

**Pre-Condition:**

NOTE: This test applies if the network product is meant to handle traffic from different network domains, e.g. both O&M and control plane traffic.

The network product has at least two separate (logical) interfaces dedicated to different network domains. The vendor provides this domain related information for the tester. Network products for which the test applies and that fail to meet this precondition fail the test by definition.

**Execution Steps**

**Execute the following steps:**

1. The tester checks whether the network product refuses traffic intended for one network domain on all interfaces meant for the other network domain, and vice versa.

2. Step 1 is to be performed for all pairs of different network domains.

**Expected Results:**

The two tests are successful.

Traffic should not be passed to a domain from which it did not originate.

**Expected format of evidence:**

Evidence containing the operational results as, e.g. screenshots, log files, packet captures, error messages.

\*\*\*\*\*\*\*\*\*\*\*\*\* 35th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.6.2 No code execution or inclusion of external resources by JSON parsers

*Requirement Name*: No code execution or inclusion of external resources by JSON parsers.

*Requirement Reference*: In accordance with industry best practice

*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.

*Threat References*: TR 33.926 [4], clause 6.3.2.1, JSON Parser Exploits

*Test Case*:

**Test Name:** TC\_JSON\_PARSER\_CODE\_EXEC\_INCL

**Purpose:**

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 shall be considered untrusted, blindly executing code fragments or loading resources from a local path or Uniform Resource Identifier (URI) shall not be possible.

**Procedure and execution steps:**

**Pre-Conditions:**

- The tester has the privileges to log in the network product and to access to 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 in the form of an OpenAPI3.0 interface specification;

- The tester has access to a Web Application Security (WAS) test tool that allows the tester to generate HTTP messages exploiting JSON parsers that do not prevent the above-mentioned scenarios of code execution and loading external resources. The 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) or an external traffic analyser directly connected to the network product and on a tester machine is available.

**Execution Steps**

1. The tester uses ae WAS test tool to generate HTTP requests (as described above in pre-conditions) towards the network product’s API endpoints via its Service Based Interfaces.

2. 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 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.

**Expected Results:**

- 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.

**Expected format of evidence:**

- 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

\*\*\*\*\*\*\*\*\*\*\*\*\* 36th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.6.3 Unique key values in Information Elements (IEs)

*Requirement Name*: Validation of the unique key values in IEs.

*Requirement Reference:* TS 29.501 Principles and Guidelines for Services Definition [13], clause 6.2.

*Requirement Description*: For data structures where values are accessible using names (sometimes referred to as keys), e.g. a JSON object, the name is expected to be unique. The occurrence of the same name (or key) twice within such a structure leads to an error and the rejection of the message.

*Threat References*: TR 33.926 [4], clause 6.3.2.2, JSON Parser not Robust

*Test Case*:

NOTE: This requirement can also be verified as part of Robustness and Protocol fuzzing tests as defined in clause 4.4.4 Robustness and fuzz testing according to referenced requirements.

**Purpose:**

Verify that the API implementation fullfills the requirements as specified in 29.501 [13], clause 6.2.

**Pre-Conditions:**

Test environment with network product under test so that the tester is able to send HTTP requests with keys (valid and duplicate) in message IE payload towards the network product under test. Rest of the network and network products may be simulated.

**Execution Steps**

1) The test equipment sends HTTP requests with duplicate keys in message IE payload to the network product under test.

2) The test equipment sends valid requests to network product under test

**Expected Results:**

1) Network product under tests responses with an error message

2) Network product under test still responses normally to valid requests

**Expected format of evidence:**

- 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).

- Log/evidence tracing possible crashes

- Information of any input causing unspecified, undocumented, or unexpected behaviour

\*\*\*\*\*\*\*\*\*\*\*\*\* 37th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

#### 4.3.6.4 The valid format and range of values for IEs

*Requirement Name*: Validation of the IEs limits.

*Requirement Reference:* TS 29.501 Principles and Guidelines for Services Definition [13], clause 6.2

*Requirement Description*: The valid format and range of values for each IE, when applicable, is defined unambiguously:

- For each message the number of leaf IEs does not exceed 2048K.

- The maximum size of the JSON body of any HTTP request does not exceed 16 million octets.

- The maximum nesting depth of leaves does not exceed 32.

*Threat References*: TR 33.926 [4], clause 6.3.2.2, JSON Parser not Robust

*Test Case*:

NOTE 1: This requirement can also be verified as part of Robustness and Protocol fuzzing tests as defined in clause 4.4.4 Robustness and fuzz testing according to referenced requirements.

**Purpose:**

Verify that the API implementation fullfills the requirements as specified in 29.501[13], clause 6.2.

**Pre-Conditions:**

Test environment with network product under test so that the tester is able to send HTTP requests with “out of bound IEs” towards the network product under test.. Rest of the network may be simulated.

NOTE 2: IEs having invalid format and/or not in the defined range of values can be considered as out of bound IEs.

**Execution Steps**

1) The test equipment sends HTTP requests with out of bounds IEs towards the network product under test.

**Expected Results:**

- Network product under tests responses with an error message.

**Expected format of evidence:**

- 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).

- Log/evidence tracing possible crashes.

- Information of any input causing unspecified, undocumented, or unexpected behaviour.

\*\*\*\*\*\*\*\*\*\*\*\*\* 38th of Change\*\*\*\*\*\*\*\*\*\*\*\*\*

### 4.4.4 Robustness and fuzz testing

*Requirement Name:* Robustness and fuzz testing

*Requirement Reference:* 4.2.6.2.2. – Interface Robustness

*Requirement Description*:

It shall be ensured that externally reachable services are robust enough to detect or dismiss unexpected or malformed input.

*Threat References*: TR 33.926 [4], clause 5.3.7, Denial of service

*Test case*:

**Test Name**: TC\_BVT\_ROBUSTNESS AND FUZZ TESTING

**Purpose:**

To verify that the network product provides externally reachable services which are robust against unexpected or malformed input. The target of this test are the protocol stacks (e.g. diameter stack) rather than the applications (e.g. web app).

**Procedure and execution steps:**

**Pre-Conditions:**

- 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 be capable 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 effective fuzz testing tools available especially for protocols proprietary to the Telco industry. Taking into account note 4 in clause 7.2.4 of TR 33.916 [19], 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 tester is expected to recognize faults, misuse, or crashes in the protocol under test to determine the level of effectiveness of the available tools.

- A network traffic analyser on the network product (e.g. TCPDUMP) or an external traffic analyser directly connected to the network product and on a tester machine is available.

**Execution Steps**

The tester is required to execute the following steps:

1. Execution of fuzzing tools against the protocols available via interfaces providing IP-based protocols of the Network Product for a coverage of tests sufficient to be effective.

2. Execution of robustness test tools against the protocols available via interfaces providing IP-based protocols of the Network Product for a coverage of tests 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 processed correctly 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.

**Expected Results:**

A list of all of the protocols of the network product reachable externally on an IP-based interface, together with an indication whether 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 shall be used to 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.

**Expected format of evidence:**

- 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

- Log/evidence tracing possible crashes

- Any input causing unspecified, undocumented, or unexpected behaviour

\*\*\*\*\*\*\*\*\*\*\* End of Change\*\*\*\*\*\*\*\*\*\*