**3GPP TSG-SA3 Meeting #116 *S3-242454***

**Jeju, Korea (Republic Of), 20th May 2024 - 24th May 2024**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.3* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  | **33.518** | **CR** | **draftCR** | **rev** | **-** | **Current version:** | **18.0.0** |  |
|  | | | | | | | | |
| *For* ***HE******LP*** *on using this form: comprehensive instructions can be found at  http://www.3gpp.org/Change-Requests.* | | | | | | | | |
|  | | | | | | | | |

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

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | |
| ***Title:*** | Add parameters to NRF discovery authorization | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** | BSI (DE) | | | | | | | | | |
| ***Source to TSG:*** | S3 | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** | SCAS\_5G\_Maint | | | | |  | ***Date:*** | | | 2024-05-10 |
|  |  | | | |  | |  | | |  |
| ***Category:*** | **F** |  | | | | | ***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. | | | | | | | | *Use one of the following releases: Rel-8 (Release 8) Rel-9 (Release 9) Rel-10 (Release 10) Rel-11 (Release 11) … Rel-17 (Release 17) Rel-18 (Release 18) Rel-19 (Release 19)  Rel-20 (Release 20)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | | Currently, only the authorization of the NF consumer's slice of the discovery request is checked. However, security requirements demand that other scopes of authorization, such as PLMN or NfType, are checked when responding to a discovery request. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | Add new test cases that check the authorization of the discovery endpoint consumer. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | In the worst case, NRF implementations do not correctly check the authorization of the discovery consumer, resulting in an attack vector that allows an attacker to retrieve internal information about the network architecture. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | | 4.2.2.2.1 | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **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 1st CHANGE \*\*\*\*\*\*\*\*\*\*

#### 4.2.2.2 NF discovery procedure

##### 4.2.2.2.1 NF discovery authorization based on expected NF profile

*Requirement Name*: NF discovery authorization for specific scopes

*Requirement Reference:* TS 33.501 [3], clause 5.9.2.2, TS 23.502 [4], clause 4.17.4, and TS 29.510 [5], clause 6.2.3.2.3.1.

*Requirement Description*:

NRF is expected to be able to ensure that NF Discovery and registration requests are authorized as specified in TS 33.501 [3], clause 5.9.2.1.

The NRF authorizes the Nnrf\_NFDiscovery\_Request. Based on the profile of the expected NF/NF service and the type of the NF service consumer, the NRF determines whether the NF service consumer is allowed to discover the expected NF instance(s). If the expected NF instance(s) or NF service instance(s) are deployed in a certain network slice, NRF authorizes the discovery request according to the discovery configuration of the Network Slice, e.g. the expected NF instance(s) are only discoverable by the NF in the same network slice as specified in TS 23.502 [4], clause 4.17.4.

The Network Repository Function (NRF) receives NF Discovery Request from an NF instance, provides the information of the discovered NF instances to the NF instance, and maintains NF profiles. The NRF uses authorization parameters (allowedXXX parameters) and allowedRuleSet in NF (Service) Profile of NF Service Producer to determine whether a given NF Instance / NF Service Instance can be discovered by an NF Service Consumer as specified in the TS 29.510 [68] clause 6.1.6.2.2, 6.2.3.2.3 and Annex C, and only returns NF Instances in the discovery response message whose authorization parameters allow the requester NF Service Consumer to access their services, as specified in the TS 33.501 [3], clause 5.9.2.2.

*Threat References*: TR 33.926 [6], clause H.2.2.1, Authorization of NF discovery based on Authorization Parameters

*Test Case*:

**Test Name:** TC\_DISC\_AUTHORIZATION\_ALLOWED\_PARAMETER

**Purpose:**

Ensure that the NRF being tested does not authorize a discovery request from an NF instance that lacks the correct authorization provided in the request, based on the allowedXXX parameters provided in the NF profile.

**Procedure and execution steps:**

**Pre-Conditions:**

- Test environment with the NF1 and NF2, which may be simulated.

- The NRF documentation provides information on whether unauthorized requests are rejected or accepted, but only returns NF Instances in the discovery response whose authorization status is specified. If this is configurable, the tester is required to test both options.

- If the NRF under test does not support parameters from the allowedList in the table below, the test steps regarding this parameters are not applicable.

**Execution Steps**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case | *parameter NF1* | *parameter NF2* | *allowedList (NF1)* | *requester-type (NF2)* |
| A | NfType NF1 | NfType NF2 | allowedNfTypes | requester-nf-type |
| B | PLMN NF1 | PLMN NF2 | allowedPlmns | requester-plmn-list |
| C | FQDN NF1 | FQDN NF2 | allowedNfDomains | requester-nf-instance-fqdn |
| D | SNPN NF1 | SNPN NF2 | allowedSnpns | requester-snpn-list |
| E | S-NSSAI NF1 | S-NSSAI NF2 | allowedNssais | requester-snssais |
| F | S-NSSAI NF1 and PLMN NF1 | S-NSSAI NF2 and PLMN NF2 | allowedPlmns | requester-plmn-specific-snssai-list |

For all Test Case specific parameters defined in the table, the tester shall repeat the following execution steps.

1. The tester configures NF1 with *parameter NF1* and NF2 with *parameter NF2*, where the two parameter values are different. The tester should select the mandatory and optional profile parameters for NF1 and NF2, ensuring that they do not conflict with other authorization test cases in this section.

2. The tester configures NF1 to ensure that it is not accessible by a certain NF by disallowing *parameter NF2* via the *allowedList parameter* in the profile NF1.

3. The tester triggers NF1 and NF2 to carry out the OAuth2.0 Access Token Request Procedure and also to register as a new NF instance via the NFManagement API at the NRF under test.

4. The tester triggers NF2 to send an Nnrf\_NFDiscovery\_Request message to the NRF under test with *target-nf-type* set to NfType NF1 and *requester-type parameter* set to the corresponding *parameter NF2*

**Expected Results:**

If the NRF under test is configured to reject unauthorized requests to all requests the NRF responds with a “403 Forbidden” status code, as specified in clause 5.3.2.2.2 of TS 29.510 [5]. If the NRF under test is configured to accept unauthorised requests, but only returns NF instances whose authorisation is accepted in the discovery response, the discovery response will not contain any information about the NF B.

**Expected format of evidence:**

Evidence suitable for the interface, e.g., evidence can be presented in the form of packet trace (pcap-file).

\*\*\*\*\*\*\*\*\*\* END OF CHANGE \*\*\*\*\*\*\*\*\*\*