**3GPP TSG-SA3 Meeting #119 S3-245240**

Orlando, US 11 – 15 November 2024 *revision of S3-244933*

**Source: Nokia**

**Title: Update to KI2 Solution 15 on Authorization token request handling in CAPIF interconnect**

**Document for: Approval, Information, Discussion**

**Agenda Item: 5.19**

# 1 Decision/action requested

***Update to Solution on authorization aspect in CAPIF interconnect***

# 2 References

[1] 3GPP TS 23.700-22

[2] 3GPP TS 33.700-22

# 3 Rationale

*Solution on authorization aspect in CAPIF interconnect to address KI2 needs to be updated to clarify onboarding aspects. Evaluation to be added.*

*The proposed solution does not aim to address problems related to onboarding procedures, but aims to leverage existing ones. Thus, the EN is considered resolved.*

Editor's Note: The onboarding part of API invoker being part of UE is ffs.

# 4 Detailed proposal

*\*\*\*\*\*\*\*\*\*\* START OF CHANGES*

## 6.15 Solution #15: Authorization token request handling in CAPIF interconnect

### 6.15.1 Introduction

This solution is addressing KI#2 on security aspects for CAPIF interconnect, specifically the authorization aspect that an access token needs to be requested by the CCF from the other CCF to which the AEF is associated.

### 6.15.2 Solution details

#### 6.15.2.1 Summary

An API invoker is onboarded at CCF-B, i.e. in the same trust domain, and wants to consume application services offered by an AEF outside of its trust domain. The AEF services are therefore not registered at the same CCF (CCF-B) as the API invoker but at a CCF-A. Hence, in interconnect an access token needs to be requested from CCF-A in a different trust domain. This is done by the CCF-B where the API invoker belongs to.

The API invoker includes onboarding secret and possibly also a client credential assertion (CCA) information into its access token request. CCF-B processes the information and forwards to CCF-A. Note, the onboard secret is not needed at CCF-A, hence it will be removed.

CCF-A verifies the access token request and CCA, if available. If successfully verified, CCF-A provides back to CCF-B an access token, with the client identifier in the access token claims set to the source API invoker ID as verified before.

Note: The assumption is that cross-domain certification is enabled, which allows CCF-A to verify the signature of the requesting API invoker known in CCF-B before creating the access token.

CCF-B provides the access token to the API invoker, which then can establish a TLS connection with AEF and invoke the northbound API with an OAuth 2.0 Access Token.

#### 6.15.2.2 Information flow



**Figure 6.15.2.2-1: Information flow to retrieve security method in interconnect**

**Step 1**: CAPIF-1/e authentication and secure session establishment is performed.

NOTE 0: APIInvoker can be residing on a UE or can be any outside the UE (e.g., an Application Function).

**Step 2:** After successful establishment of TLS session over CAPIF-1e, the API invoker sends to the CCF-B an Access Token Request message and optionally CCA token signed (using the APIInvoker’s private key) as per the OAuth 2.0 specification.

NOTE 1: The API invoker may include the CAPIF core function assigned API invoker ID and the Onboard\_Secret in the OAuth access token request message for the CAPIF core function to validate the access token request.

NOTE 2: The CCA token will be compliant to Json Web Token IETC RFC 7519 and includes the APIInvokerID, CCF ID, timestamp (iat) and the expiry time (exp). The lifetime of the CCA token is expected to be smaller than the expiry time associated with the OAuth 2.0 access token. The CCA token when received at CCF-A ensures the request is originated by APIInvoker.

**Step 3:** CCF-B verifies the Access Token Request message per OAuth 2.0 specification. In the above figure, Resource Owner is shown as part of CCF-A’s domain. Instead, if Resource Owner is part of CCF-B’s domain, then CCF-B can fetch the resource owner consent using RNAA (e.g. if consent is the applicable legal basis).

Editor’s Note: How CCF-B gets resource owner authorization information using RNAA should be clarified.

**Step 4:** If CCF-B cannot successfully verify the Access Token Request message from APIInvoker, it provies an error message back.

If successfully verified, CCF-B creates a new Access Token request using the Access Token Request it received from APIInvoker. The new Access Token Request does not include the Onboard\_Secret as received in the access token request in step 2) anymore. The remaining parameters in step 2 are reused. In addition, the source, e.g. "sourceAPIInvokerID” is added and is set to the value of client\_id received in step 2 (APIinvokerID). client\_id is set to CCF-B identifier.

If the Resource Owner is part of CCF-B’s domain then and consent was retrieved, then CCF-B can also include the consent information.

If the Resource Owner is within CCF-A’s domain, and the consent needs to be captured, then CCF-B includes information about the APIInvoker identity (sourceAPIInvokerID) as well as consent-specific parameters (e.g., purpose of the data processing). By this, the RO can identify the party requesting access to the protected resources and the reason for that. The APIInvoker certificate, which is retrieved locally, is added as additional IE.

**Step 5:** CCF-B sends the newly generated OAuth access token request to CCF-A along with the CCA token, if available and as provided by the APIInvoker) and the APIInvoker certificate.

**Step 6:** CCF-A verifies the Access Token Request as per OAuth2.0 specification. (more details are below)

**Step 7:** CCF-A verifies if CCF-B is authorized for the service. It validates the CCA token, if available, with the received APIInvoker certificate. It validates whether the sourceAPIInvokerID and CCF-B id in CCA token are matching with the Access Token Request received. CCA token verification ensures the Access Token Request is originated from APIInvoker and also to authenticate APIInvoker at CCF-A.

CCF-A verifies if the APIInvoker is authorized for the service if RO is part of CCF-B and consent information if available in the token is valid, or if RO is part of CCF-A’s domain, and consent is applicable. In the latter case, it gets the consent from resource owner using RNAA.

**Step 8:** After successful validation, CCF-A generates an access token response with a token including client\_id in AccessTokenClaims set to sourceAPIInvokerID present in step 4.

**Step 9:** CCF-A sends the access token response to CCF-B.

**Step 10:** CCF-B forwards the access token response to APIInvoker.

**Step 11-12:** After successful authentication to the AEF on CAPIF-2e, the API invoker shall initiate invocation of a 3GPP northbound API with the AEF. The access token received in step 10 is included along with the northbound API invocation request.

**Step 13:** The API exposing function shall validate the access token. The AEF verifies the integrity of the access token by verifying the CAPIF core function signature. If validation of the access token is successful, the AEF verifies the API invoker's Northbound API invocation request against the authorization claims in access token, ensuring that the API Invoker has access permission for the requested service API.

**Step 14:** After successful verification of the access token and authorization claims of the API invoker, the requested northbound API is invoked, and the appropriate response is returned to the API invoker.

### 6.15.3 Evaluation

The solution partly addresses the 3rd requirement of KI#2 by enabling the API invoker to gain an access token issued by the CCF-A to which the AEF is associated from the CCF-B it registers with.

The solution is presenting a mechanism for API invoker to retrieve an access token which is understood by the AEF in a different domain.

The solution proposes that the CCF-B sends a new OAuth access token request to CCF-A along with the APIInvoker certificate and optional the CCA token. The new OAuth access token request is created using the Access Token Request received from APIInvoker, but with a new parameter indicating the API invoker ID and setting the client\_id to the CCF-B identifier.

The solution has an impact on the CCF functionalities of the Access token request procedure, and introduces additional communications between interconnected CCFs.

As for access token generation, firstly the solution recommends that the CCF-A validates the CCA token, if available, with the received APIInvoker certificate to ensure the Access Token Request is originated from APIInvoker and also to authenticate APIInvoker. Secondly, the solution proposes that the CCF-A verifies the consent information in the new OAuth access token request if the Resource Owner is part of CCF-B’s domain, or gets the consent information from the resource owner based on consent-specific parameters if the Resource Owner is within CCF-A’s domain.The revocation of the API invoker in CAPIF interconnection scenarios is not addressed in the solution.

The solution has also impact on the API invoker for creating the CCA.

*\*\*\*\*\*\*\*\*\*\* END OF CHANGES*