**3GPP TSG-SA3 Meeting #109AdHoc-e *draft\_S3-230339-r2***

**Electronic meeting, 16 - 20 January 2023**

**Source: Samsung, Lenovo**

**Title: New Solution on Resource owner Authorization in API Invocation**

**Document for: Approval**

**Agenda Item: 5.11**

# 1 Decision/action requested

***The contribution proposes a new solution for key issue #2 of TR 33.884.***

# 2 References

NA

# 3 Rationale

This contribution proposes a new solution for a resource owner to provide/revoke the resource owner’s authorization to API Invoker.

# 4 Detailed proposal

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Start of 1st Change \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# 2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non‑specific.

- For a specific reference, subsequent revisions do not apply.

- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] 3GPP TS 22.261: "Service requirements for the 5G system".

[3] 3GPP TR 23.700-95: “Study on application enablement aspects for subscriber-aware northbound API access”.

[4] IETF RFC 6749: “The OAuth 2.0 Authorization Framework”.

[5] 3GPP TS 33.122: “Security aspects of Common API Framework (CAPIF) for 3GPP northbound APIs”.

[6] openID.net: " OpenID Connect Core 1.0 incorporating errata set 1". Available at: <https://openid.net/specs/openid-connect-core-1_0.html>

[7] IETF RFC 7009: “OAuth 2.0 Token Revocation”.

[8] IETF RFC 7515: “JSON Web Signature (JWS)”.

[xx] IETF RFC 7542: “The Network Access Identifier".

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* End of 1st Change \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

## 6.X Solution #X: Providing and Revoking Resource Owner Authorization

### 6.X.1 Introduction

This solution addresses Key Issue #2 "Checking authorization before allowing access".

This solution proposes to use a resourse owner’s 3GPP credentials for UE to generate a token, which is used to validate an API Invoker accessing the resource owner’s resources.

A resource owner in this solution is an end-user who is using the UE.

### 6.X.2 Solution details

### 6.X.2.1 Architecture



Figure 6.X.2.1-1 architecture for CAPIF with SNA enhancement

This solution uses an architecture proposed in solution #2 of TR 23.700-95 [3]. As defined in TR 23.700-95 [3], the Resource owner client is an application client used by end-user or subscriber of the API provider domain's service provider.

### 6.X.2.2 Procedure

Pre-requisite:

* During the primary authentication, AUSF receives Routing Indicator for Authorization Function from UDM.
* After the primary authentication, UE and AUSF generate S-KID (SNAAPPY Key Identifier) and KAuz from KAUSF as detailed in 6.X.2.3 and 6.X.2.4 of this document, respectively. After the key material is generated, AUSF selects the Authorization Function based on the Routing Indicator and sends the KAuz, S-KID, and SUPI to the Authorization Function. The Authorization Function stores the latest information sent by the AUSF.
* Information on Authorization Function (e.g. address, Routing Indicator, etc.) which holds the KAuz is provisioned to the UE (e.g. during registration procedure).
* API Invoker knows which APIs require a resource owner's authorization.



Figure 6.X.2.2-1 Procedure for resource owner authorization based API invocation

1. If API Invoker does not have TokenAuz for service API invocation which requires the resource owner’s authorization, API Invoker needs to request resource owner’s authorization for the API invocation even if the API invocation is authorized from API exposing function as defined in TS 33.122 [5].
2. API Invoker requests resource owner’s authorization for the API invocation. The request message includes Service API name (e.g. QoS API, location API, etc.) and API Invoker Information (e.g. API Invoker identity which is provided from CAPIF Core Function).
3. When the resource owner decides to give an authorization on the Service API name to the API Invoker (e.g. using GUI), UE generates an authorization token (TokenAuz). The claims of the TokenAuz include service API name, S-KID (SNAAPPY Key Identifier), API Invoker Information, “Authorized”, generated time, and validity time. TokenAuz contains the claims (TokenAuz, claim) and the verification information (TokenAuz, verify). Details of S-KID and the corresponding key KAuz are specified in 6.X.2.3 and 6.X.2.4 of this document, respectively. TokenAuz, verify is generated as detailed in 6.X.2.5 by using the claims and the key KAuz.
4. If the resource owner gives the authorization for the API invocation, UE responds with the TokenAuz. Upon receving the response, the API Invoker stores the TokenAuz with UE ID (e.g. application layer ID or GPSI or SUPI). The API Invoker can use the TokenAuz for the API invocation until the TokenAuz is expired by an expiration time or revoked by the resource owner, even when there is no online connection between the API Invoker and the UE.
5. API Invoker performs the service API invocation with the TokenAuz.
6. API exposing function requests token verification to Authorization Function, via CAPIF-9 interface.
7. Authorization Function finds KAuz matched to S-KID which is included in TokenAuz, claim and verifies the TokenAuz using KAuz.
8. Authorization Function responds with the verification result and UE ID (SUPI or GPSI).
9. If the verification result of the TokenAuz is successful, API exposing function stores the TokenAuz with UE ID (SUPI or GPSI). Untill API exposing function receives a revocation notification for the service API invocation or the TokenAuz is expired by an expiration time, API exposing function uses the stored TokenAuz for authorizing the API Invoker without performing token verification request to Authorization Function.
10. API Invoker receives the service API invocation response.
11. If the resource owner does not want for the API Invoker to invoke the service API, the resource owner can revoke the TokenAuz anytime before the validity time of the TokenAuz by using resource owner client. When the resource owner decides to revoke the TokenAuz for the service API, UE generates a revocation token (TokenRev). The claims of TokenRev include service API name, A-KID, API Invoker information, “Not authorized”, generated time. TokenRev contains the claims (TokenRev, claim) and the verification information (TokenRev, verify). TokenRev, verify is generated as detailed in 6.X.2.5 by using the claims and the key KAuz.
12. UE and Authorization function perform mutual authentication based on TLS-PSK as specified in clause 6.5.2.1 in TS 33.122, where PSK can be derived from KAuz. The TokenRev is transmitted to Authorization Function via CAPIF-8 interface with revocation request message.
13. Authorization Function finds KAuz by using S-KID which is included in the TokenRev, claim. Authorization Function verifies the TokenRev using KAuz.
14. Authorization Function notifies the revocation on the API Invoker’s service API invocation. The revocation notification includes the TokenRev and UE ID.
15. For the same UE ID stored at step 9 and received at step 14, API exposing function finds the TokenAuz which has same service API name and API Invoker information as the TokenRev. If the generated time of the TokenAuz is prior to that of the TokenRev, API exposing function revokes the TokenAuz and stores the TokenRev. After this, if the API Invoker performs service API invocation using the revoked TokenAuz, API exposing function shall reject the API invocation request by noticing that the generation time in the TokenAuz is prior to the generation time in the TokenRev.
16. API exposing function notifies the revocation of the token.

### 6.X.2.3 S-KID

S-KID is in NAI format as specified in clause 2.2 of IETF RFC 7542 [xx], i.e. username@realm. The username part includes the Routing Indicator for Authorization Function and S-TID (SNAAPPY Temporary UE Identifier), and the realm part includes Home Network Identifier.

When deriving S-TID from KAUSF, the following parameters shall be used to form the input S to the KDF:

* FC = 0xXX;
* P0 = "S-TID";
* L0 = length of "S-TID";
* P1 = SUPI;
* L1 = length of SUPI.

The input key KEY shall be KAUSF.

NOTE: FC value to be determined during normative phase.

### 6.X.2.4 KAuz derivation function

When deriving KAuz from KAUSF, the following parameters shall be used to form the input S to the KDF:

* FC = 0xYY;
* P0 = “Authorization”;
* L0 = length of “Authorization”;

The input key KEY shall be the KAUSF.

NOTE: FC value to be determined during normative phase.

### 6.X.2.5 Verification information derivation

When deriving the verificiation information (TokenAuz, verify or TokenRev, verify) from KAuz, the following parameters shall be used to form the input S to the KDF.

* FC = 0xZZ;
* P0 = TokenAuz, claim or TokenRev, claim;
* L0 = length of TokenAuz, claim or TokenRev, claim;

The input key KEY shall be KAuz.

The verification information is identified with the 128 least significant bits of the output of the KDF.

NOTE: FC value to be determined during normative phase.

### 6.X.3 Evaluation

TBD

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