**3GPP TSG-SA3 Meeting #119AdHoc-e S3-250079-r2**

Online, Electronic meeting, 13 -16 January 2025

**Source: Ericsson**

**Title: New Solution to KI#5**

**Document for: Approval**

**Agenda Item: 5.9**

# 1 Decision/action requested

***This document proposes a new solution to KI#5 in TR 33.713***

# 2 References

# 3 Rationale

This solution addresses KI#5 (Authentication in Ambient IoT service)

# 4 Detailed proposal

### 6.Y.1 Introduction

This solution addresses KI#5 (Authentication in Ambient IoT service)

### 6.Y.2 Solution details

It is assumed that Device and 5GC share a cryptographic key K, called the primary key. The figure below outlines a mutual authentication protocol between the AIoT device and 5GC.



Figure Z: Authentication between AIoT device and 5GC

The figure is explained step-by-step in the following:

1. Once the 5G core network decides to reach an AIoT device, it generates a random authentication challenge Rand1.
2. The 5GC sends a service request to the AIoT reader/gNB. The message includes Rand1 and a paging identifier. The paging identifier may target a group of devices or a single device. If the paging identifier targets a single AIoT device, then the paging identifier can be a Temporary ID (TID) of the target the AIoT device if such a TID was previously assigned to the target AIoT device. If a TID was not previously assigned to the target AIoT device, then the paging identifier is a concealed version of the permanent identifier of the target AIoT device — the concealment is done using the primary key K.
3. The AIoT reader/gNB broadcasts the paging identifier and challenge Rand1.
4. The device checks if the paging identifier is meant for itself. If the paging identifier targets a group of devices, then the device checks if the device belongs to the group or not. If the paging identifier is a TID and the device has a previously assigned TID, then the device checks if the TID matches with the device’s TID. If the paging identifier is a concealed permanent identifier, then the device deconceals the permanent identifier and checks if it matches with the deivce’s permanent identifier.
5. If the paging is meant for the device, then the device starts the random access procedure.
6. The device sends random access message 1 to the AIoT reader/gNB
7. The reader checks if it sent a paging message recently or not. If not paging message was sent recently, the reader abandon the procedure.
8. AIoT reader/gNB sends random access message 2 to the AIoT device.
9. AIoT device randomly generate Rand2, compute MAC tag T over Rand1, and Rand2 using primary key K.
10. The AIoT device sends its identity, tag T, and the randomly generated Rand2 to the AIoT reader/gNB. If the paging identifier Step 2 included a TID of the device, then the device also sends the TID as the identiy of the device. If the paging identifier received in Step 2 was targeted to a group of devices or was a concealed permanent identifier, then the device sends its primary identifier in a concealed manner, e.g., by sending an AICI that the device obtained earlier from the network, or by encrypting the primary identifier using the primary key K.
11. The AIoT reader/gNB forwards the Device ID, T, and Rand2 to the 5GC.
12. The 5GC resolves the permanent identity and primary key of the device based on the received device identifier. 5GC verifies T by performing the same computation as on device side. If T is verified successfully, then the AIoT device is considered authenticated. 5GC Computes Kenc, Kint using a KDF function taking inputs Rand1, Rand2, H, and key K.   
    The 5GC generates a command message including a fresh temporary ID (TID) and optionally paging identifier and command counter, encrypts it using Kenc, and computes a MAC on the ciphertext using Kint. The TID can be an encryption of the device’s permanent identity using the public key of the network — such a TID can also be called an AICI (AIoT Concealed Identifier).
13. The 5GC sends the encrypted command message, (optionally paging identifier and command counter), MAC T´ to the AIoT reader/gNB.
14. The AIoT reader/gNB forwards the encrypted command message, (optionally paging identifier and command counter), and the MAC T´ to the AIoT device.
15. The AIoT device derives keys Kenc and Kint from the shared key K in the same manner as in 5GC. The AIoT device validates MAC T´ and decrypts command message. If the MAC T´ is valid, then the network is considered authenticated. The the AIoT device extracts and store (or replace previously stored) TID. Finally, the device executes the command.
16. The AIoT device sends command execution acknowledgement to the AIoT reader/gNB
17. The AIoT reader/gNB forwards the command execution acknowledgement to the 5GC.
18. The 5GC stores the TID mapped with the permanent identity of the device.
19. Network and device store context info e.g., TID, Kenc, Kint. At this stage the 5GC considers the device to be registered in the network.

NOTE 1: Exact non-security-related content of the messages (e.g., the paging message) exchanged and 5GC NFs involved depend on SA2 and RAN specifications. Cryptographic details about computing cryptographic hash H, cryptographic keys Kenc, Kint, and MAC tags T and T´ can be figured out during conclusion phase.

NOTE 2: Device includes the hash H in the computation of the MAC tag T to bind the authentication to the service context (e.g., when group identifier used for group-based paging). Rand2 which is a device-side generated authentication challenge is also included in the MAC computation.

NOTE 3: If a device is already registered in the network, the 5GC may decide to send the command directly starting from Step 11 without running the authencation protocol from the beginning.

### 6.Y.3 Evaluation

The solution fulfils the potential security requirement in KI#5.

If both inventory and command is run, it takes three round trips (six messages) between the AIoT device and the reader. However, it takes two round trips (four messages) between the AIoT device and the reader if only inventory is performed.

Editor’s Note: Futher evaluation is FFS