|  |  |
| --- | --- |
| 3GPP TR 33.713 V0.4.0 (2024-10) | |
| Technical Report | |
| 3rd Generation Partnership Project;  Technical Specification Group Services and System Aspects;  Study on Security Aspect of Ambient IoT Services in 5G  (Release 19) | |
|  | |
|  |  |
| The present document has been developed within the 3rd Generation Partnership Project (3GPP TM) and may be further elaborated for the purposes of 3GPP. The present document has not been subject to any approval process by the 3GPPOrganizational Partners and shall not be implemented. This Specification is provided for future development work within 3GPPonly. The Organizational Partners accept no liability for any use of this Specification. Specifications and Reports for implementation of the 3GPP TM system should be obtained via the 3GPP Organizational Partners' Publications Offices. | |

|  |
| --- |
|  |
| ***3GPP***  Postal address  3GPP support office address  650 Route des Lucioles - Sophia Antipolis  Valbonne - FRANCE  Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16  Internet  http://www.3gpp.org |
| ***Copyright Notification***  No part may be reproduced except as authorized by written permission. The copyright and the foregoing restriction extend to reproduction in all media.  © 2024, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI, TTA, TTC).  All rights reserved.  UMTS™ is a Trade Mark of ETSI registered for the benefit of its members  3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners LTE™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners  GSM® and the GSM logo are registered and owned by the GSM Association |

Contents

Foreword 8

Introduction 9

1 Scope 10

2 References 10

3 Definitions of terms, symbols and abbreviations 10

3.1 Terms 10

3.2 Symbols 11

3.3 Abbreviations 11

4 Architecture and Security Assumptions 11

5 Key issues 11

5.1 Key Issue #1: Protection for disabling device operation 11

5.1.1 Key issue details 11

5.1.2 Threats 11

5.1.3 Potential security requirements 11

5.2 Key Issue #2: Authorization for 5G Ambient IoT services 12

5.2.1 Key issue details 12

5.2.2 Security threats 12

5.2.3 Potential security requirements 12

5.3 Key issue #3: Privacy by protecting AIoT device identifiers 12

5.3.1 Key issue details 12

5.3.2 Security Threats 12

5.3.3 Potential security requirements 13

5.4 Key issue #4: Protection of information during AIoT service communication 13

5.4.1 Key issue details 13

5.4.2 Security threats 13

5.4.3 Potential security requirements 13

5.5 Key Issue #5: Authentication in Ambient IoT service 13

5.5.1 Key issue details 13

5.5.2 Threats 14

5.5.3 Potential security requirements 14

5.6 Key issue #6: Exposure of Inventory Device Quantity 14

5.6.1 Key issue details 14

5.6.2 Security threats 14

5.6.3 Potential security requirements 15

5.X Key Issue #X: <Key Issue Name> 15

5.X.1 Key issue details 15

5.X.2 Security threats 15

5.X.3 Potential security requirements 15

6 Solutions 15

6.0 Mapping of solutions to key issues 16

6.1 Solution #1: Ambient IoT device disabling mechanism 16

6.1.1 Introduction 16

6.1.2 Solution details 17

6.1.3 Evaluation 18

6.2 Solution #2:PCF based Service Authorization and Provisioning to UE 19

6.2.1 Introduction 19

6.2.2 Solution details 19

6.2.3 Evaluation 19

6.3 Solution #3: Authorization of Intermediate UE for AIoT services 19

6.3.1 Introduction 19

6.3.2 Solution details 19

6.3.3 Evaluation 20

6.4 Solution #4: Protection for inventory and command procedure 21

6.4.1 Introduction 21

6.4.2 Solution details 21

6.4.2.1 Protection for inventory-only procedure 21

6.4.2.2 Protection for inventory and command procedure 22

6.4.2.3 Auth\_token and XAuth\_token derivation function 23

6.4.3 Evaluation 23

6.5 Solution #5: Disabling and Enabling AIoT Device 23

6.5.1 Introduction 23

6.5.2 Solution details 24

6.5.3 Evaluation 25

6.6 Solution #6: AIoT device authentication 25

6.6.1 Introduction 25

6.6.2 Solution details 25

6.6.3 Evaluation 27

6.7.1 Introduction 27

6.7.2 Details 27

6.7.3 Evaluation 30

6.8 Solution #8: Mutual authentication for AIoT system 30

6.8.1 Introduction 30

6.8.2 Details 30

6.8.3 Evaluation 32

6.9 Solution #9: Device authentication and data communication security 32

6.9.1 Introduction 32

6.9.2 Solution details 32

6.9.3 Evaluation 34

6.10.1 Introduction 35

6.10.2 Solution details 35

6.102.1 UE reader case 35

6.10.2.1.1 Alternative 1 – UE reader granularity 35

6.10.2.1.2 Alternative 2 – AIoT device granularity 36

6.10.2.2 RAN reader case 37

6.10.3 Evaluation 38

6.11 Solution #11: Authentication and ID Privacy of AIoT devices with USIM on AIoT AS Layer 39

6.11.1 Introduction 39

6.11.2 Solution details 39

6.11.3 Evaluation 41

6.12 Solution #12: Authentication and ID Privacy of AIoT devices with USIM on AIoT Layer 41

6.12.1 Introduction 41

6.12.2 Solution details 42

6.12.3 Evaluation 44

6.13 Solution #13: Authentication and ID privacy of AIoT devices without USIM 44

6.13.1 Introduction 44

6.13.2 Solution details 45

6.13.3 Evaluation 46

6.14 Solution #14: Information protection during AIoT service communication 46

6.14.1 Introduction 46

6.14.2 Solution details 47

6.14.2.1 Inventory Service information protection 47

6.14.2.2 Command Service information protection 48

6.14.3 Evaluation 49

6.15 Solution #15: End-to-end security protection of command procedure 49

6.15.1 Introduction 49

6.15.2 Solution details 50

6.15.3 Evaluation 51

6.16 Solution #16: Disabling operation procedure for Ambient IoT services 51

6.16.1 Introduction 51

6.16.2 Solution details 51

6.16.3 Evaluation 52

6.17 Solution #17: Disabling operation procedure for AIoT services 53

6.17.1 Introduction 53

6.17.2 Solution details 53

6.17.3 Evaluation 54

6.18 Solution #Y: Authorization procedure for AF-based intermediate node selection 54

6.18.1 Introduction 54

6.18.2 Solution details 55

6.18.3 Evaluation 55

6.19 Solution #19: Authorization of AIoT capable UE in topology 2 55

6.19.1 Introduction 55

6.19.2 Solution details 56

6.19.3 Evaluation 58

6.20 Solution #20: Lightweight AIOT ID privacy based on hashes 58

6.20.1 Introduction 58

6.20.2 Details 59

6.20.3 Evaluation 61

6.21 Solution #21: Ephemeral AIOT ID security context based on puzzles for privacy 61

6.21.1 Introduction 61

6.21.2 Details 62

6.21.3 Evaluation 64

6.22 Solution #22: Solution for protecting AIoT ID by using temporary ID 64

6.22.1 Introduction 64

6.22.2 Solution details 65

6.22.3 Evaluation 65

6.23 Solution #23: AIoT device ID privacy protection using anonymity key 66

6.23.1 Introduction 66

6.23.2 Solution details 66

6.23.3 Evaluation 67

6.24 Solution #24: temporary ID based AIoT device privacy protection 67

6.24.1 Introduction 67

6.24.2 Solution details 67

6.24.3 Evaluation 68

6.25 Solution #25: Use temporary identifier to protect the privacy of AIoT device identifiers. 69

6.25.1 Introduction 69

6.25.2 Solution details 69

6.25.3 Evaluation 70

6.26 Solution #26: Local generated Temporary ID to provide device privacy 70

6.26.1 Introduction 70

6.26.2 Solution details 71

6.26.2.1 Temporary ID generation. 71

6.26.3 Evaluation 72

6.27 Solution #27: Privacy protection of AIoT device identifier based on a temporary identifier 72

6.27.1 Introduction 72

6.27.2 Solution details 73

6.27.2.1 Procedures 73

6.27.2.2 Generation of a temporary identifier 74

6.27.3 Evaluation 74

6.28 Solution #29: Privacy protection on AIoT device IDs 74

6.28.1 Introduction 74

6.28.2 Details 74

6.28.3 Evaluation 75

6.29 Solution #29: Providing a network-computed AIoT concealed device identifier (AICI) to an AIoT device 75

6.29.1 Introduction 75

6.29.2 Solution details 75

6.29.3 Evaluation 76

6.30 Solution #30: Privacy protection for inventory operation 77

6.30.1 Introduction 77

6.30.2 Solution details 77

6.30.2.1 Inventory procedure with unprotected inventory request parameters 77

6.30.2.2 Inventory procedure with protected inventory request parameters 78

6.30.3 Evaluation 79

6.31 Solution #31: Ambient IoT ID privacy 79

6.31.1 Introduction 79

6.31.2 Solution details 80

6.31.3 Evaluation 81

6.Y Solution #Y: <Solution Name> 81

6.Y.1 Introduction 81

6.Y.2 Solution details 81

6.Y.3 Evaluation 81

7 Conclusions 81

Annex <X> (informative): Change history 82

# Foreword

This Technical Report has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

In the present document, modal verbs have the following meanings:

**shall** indicates a mandatory requirement to do something

**shall not** indicates an interdiction (prohibition) to do something

The constructions "shall" and "shall not" are confined to the context of normative provisions, and do not appear in Technical Reports.

The constructions "must" and "must not" are not used as substitutes for "shall" and "shall not". Their use is avoided insofar as possible, and they are not used in a normative context except in a direct citation from an external, referenced, non-3GPP document, or so as to maintain continuity of style when extending or modifying the provisions of such a referenced document.

**should** indicates a recommendation to do something

**should not** indicates a recommendation not to do something

**may** indicates permission to do something

**need not** indicates permission not to do something

The construction "may not" is ambiguous and is not used in normative elements. The unambiguous constructions "might not" or "shall not" are used instead, depending upon the meaning intended.

**can** indicates that something is possible

**cannot** indicates that something is impossible

The constructions "can" and "cannot" are not substitutes for "may" and "need not".

**will** indicates that something is certain or expected to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document

**will not** indicates that something is certain or expected not to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document

**might** indicates a likelihood that something will happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

**might not** indicates a likelihood that something will not happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

In addition:

**is** (or any other verb in the indicative mood) indicates a statement of fact

**is not** (or any other negative verb in the indicative mood) indicates a statement of fact

The constructions "is" and "is not" do not indicate requirements.

# Introduction

This clause is optional. If it exists, it shall be the second unnumbered clause.

Editor’s Note: This clause contains some background information for the study.

# 1 Scope

The present document identifies potential threats and security requirements to enable AIoT services for various use cases. Consideration for the energy and complexity constraints of AIoT devices is taken into account in identifying and developing potential security mechanisms to support AIoT services. Specifically, the present document focuses on the following:

1. Identify security and privacy and threats introduced by AIoT services for use cases captured in TS 22.369 [2], for topologies captured in RP-234058[3], and for architecture captured in TR 23-700-13[4].

2. Identify security requirements to address the identified threats.

3. Develop potential solutions that fulfil the security requirements, taking into account AIoT device constraints agreed upon in other 3GPP working groups.

NOTE 1: Enable/disable device operation is within the scope of the present document.

# 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 22.369: "Service Requirements for ambient power-enabled IoT".

[3] RP-234058, RAN New SID for Study on Solution for Ambient IoT in NR.

[4] 3GPP TR 23-700-13: "Study on Architecture Support of Ambient power-enabled Internet of Things".

[5] 3GPP TS 33.501: "Security Architecture and Procedures for 5G System".

[6] R2-2406202 RAN2#126 Meeting Report

[7] RFC 4739: "Multiple Authentication Exchanges in the Internet Key Exchange (IKEv2) Protocol".

[8] 3GPP TR 38.848: "Technical Specification Group Radio Access Network; Study on Ambient IoT (Internet of Things) in RAN".

# 3 Definitions of terms, symbols and abbreviations

## 3.1 Terms

For the purposes of the present document, the terms given in 3GPP TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in 3GPP TR 21.905 [1].

**example:** text used to clarify abstract rules by applying them literally.

## 3.2 Symbols

For the purposes of the present document, the following symbols apply:

<symbol> <Explanation>

## 3.3 Abbreviations

For the purposes of the present document, the abbreviations given in 3GPP TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in 3GPP TR 21.905 [1].

<ABBREVIATION> <Expansion>

# 4 Architecture and Security Assumptions

Editor’s Note: This clause contains security architecture and assumptions to be considered for the study (e.g., per work task/KI).

The following architecture and security assumptions are applied:

* The architecture assumptions and requirements for Ambient IoT services as defined in TR 23.700-13 [4] are used as architecture assumptions in this study.
* Two functional cases are considered as baseline: (1) inventory, (2) command.

# 5 Key issues

Editor’s Note: This clause contains all the key issues identified during the study.

## 5.1 Key Issue #1: Protection for disabling device operation

### 5.1.1 Key issue details

As specified in TS 22.369 [2], the enable/disable device operation is used for the operator to manage the Ambient IoT device, which can enable/disable the Ambient IoT device's capability to transmit RF signals. Based on operator policy, there are two categories of disabling device operations, i.e. permanent disabling of the capability and temporary disabling of the capability.

### 5.1.2 Threats

As a management operation, the availability of Ambient IoT devices will be impacted if the disabling device operation is not securely performed. For example, if the Ambient IoT device follows the spoofed permanent/temporary disable device operation from an attacker, the Ambient IoT devices will not respond to the network either permanently, or for a period of time, leading to the Denial of Service (DOS).

### 5.1.3 Potential security requirements

The means to securely disable the Ambient IoT device(s)’s capability to transmit RF signals shall be supported.

Editor’s Note: Whether the solutions for this key issue are the same or different from those for communication protection issue is FFS.

Editor’s Note: Security solutions for this Key Issue should be aligned with the conclusion of Ambient IoT system architecture in SA2..

## 5.2 Key Issue #2: Authorization for 5G Ambient IoT services

### 5.2.1 Key issue details

In TR 23.700-13 [4], Key Issues #1 and #3 describe the issues on the system architecture and procedure to support 5G Ambient IoT services.

In the Topology 2 as defined in TR 38.848 [8], the UE acting as the intermediate node is responsible for transferring the information between AIoT device and 5GS. If the authorization of intermediate node is not supported, the attacker can play the role of intermediate node and arbitrarily deny 5G AIoT service.

Therefore, it is necessary to study how to authorize the UE for acting as the intermediate node.

### 5.2.2 Security threats

If the 5GC cannot verify if the UE acting as an intermediate node is authorized, the attacker UE may impersonate the intermediate node. The attacker UE may then deny the 5G Ambient IoT services.

### 5.2.3 Potential security requirements

The 5GS shall be able to support the authorization of the AIoT capable UE as an intermediate node in 5G Ambient IoT services.

## 5.3 Key issue #3: Privacy by protecting AIoT device identifiers

### 5.3.1 Key issue details

5G Ambient IoT service is a type of cellular IoT communication system where Ambient IoT devices utilize harvested energy to generate RF signals for bi-directional information transmission. Ambient IoT devices are characterized by limited functions, requiring only small and infrequent data transfers.

TS 22.369 [2] clause 5.2.6 defines the following privacy-related requirements:

“The 5G system shall be able to provide a mechanism to protect the privacy of information (e.g., location and identity) exchanged during communication between an Ambient IoT device and the 5G network or an Ambient IoT capable UE.”

In AIoT services, identifiers of AIoT device are used to identify the device. If the identifiers associated with a device are not privacy protected (e.g., exposed over the air), an attacker (e.g., an over-the-air attacker) can identify and track an AIoT device based on the identifiers associated with the AIoT device. Thus, this key issue is to investigate potential mechanisms to privacy protect the AIoT device identifiers.

### 5.3.2 Security Threats

An attacker can identify, monitor and track an AIoT device based on the identifiers associated with the AIoT device if the identifiers are not privacy protected.

Editor’s Note: It is FFS how the above threat affects various use cases.

Editor’s Note: Security threat and requirement for potential exposure of quantity of devices after adversary broadcasts an inventory message is FFS.

### 5.3.3 Potential security requirements

Mechanisms for mitigating privacy threats (described above) by identifying, linking, and tracking the identifiers of AIoT Device(s) shall be supported.

Editor’s Note: AIoT use cases that do not need the above privacy protection mechanisms are FFS.

## 5.4 Key issue #4: Protection of information during AIoT service communication

### 5.4.1 Key issue details

As per TS 22.369 [2], Ambient power-enabled IoT (AIoT) services aim to support various use cases, including inventory taking, sensor data collection, asset tracking, and actuator control. These services intended to operate with lower power consumption and complexity than the existing IoT technologies such as eMTC, NB-IoT, and RedCap. To fulfil these requirements, AIoT devices require a communication capability.

Considering the ambient IoT device will be deployed in the indoor system, if the restricted access means (e.g., in factory) are provided, the possibility of attack is minimal. However, if the environment (e.g., in shopping mall) is open to the attacker, from a security perspective, security mechanisms to protect the information transmitted during AIoT service communication need to be supported. Failure to provide such security mechanisms will lead to various attacks such as eavesdropping, manipulation and/or unauthorized transmission of the information during AIoT service communication.

### 5.4.2 Security threats

For command operation (e.g., write, read), the following threats are applicable:

An attacker may acquire data transmitted to/from AIoT devices by eavesdropping messages if the communication of AIoT service is not confidentiality protected.

An attacker may manipulate information during communication of AIoT service if the communication of AIoT service is not integrity protected.

An attacker may replay a message if replay protection is not activated.

### 5.4.3 Potential security requirements

The Ambient IoT system shall support a means to ensure confidentiality, integrity and/or anti-replay of information for AIoT services.

Editor’s Note: Whether information protection between AIoT device and 5G core is mandatory is ffs.

Editor’s Note: Whether information protection can be done on application layer is ffs.

## 5.5 Key Issue #5: Authentication in Ambient IoT service

### 5.5.1 Key issue details

The TR 23.700-13 [4] studies the architecture support of Ambient Internet of Things (AIoT) device, considering the service requirements for ambient power-enabled IoT device. In TR 23.700-13 [4], the validation of the AIoT device identity and authentication are explicitly mentioned.

### 5.5.2 Threats

In the air interface, an attacker may impersonate the victim device and report fake identification to the network side. If the billing is based on per AIoT device’s identity, the fake identity may lead to charging problem. This can be used by an adversary to steal an AIoT device by replacing the AIoT device with a fake device, which might cause a loss to the owner of the device. Whether fake identities pose a threat in case of inventory use case, is up to the individual deployments risk profile.

Over the air interface, without authentication, an adversary can impersonate a legitimate network and send a command — e.g., disablement command will permanently deactivate a device.

### 5.5.3 Potential security requirements

The 5G system shall provide a means to perform mutual authentication between the AIoT device and the network.

NOTE: If solution reuses the existing authentication framework (e.g., 5G-AKA, EAP-AKA’, other EAP methods for SNPN), the existing UE security requirements in TS 33.501[5] apply. Otherwise, the core network entities, if any, that are used for authentication are dedicated for Ambient IoT service.

Editor’s Note: Secure storage and processing of credentials (on the AIoT device and network side) for solution not reusing existing authentication framework as per the note above is ffs.

Editor's Note: Whether performing authentication is to be mandated or optional, one way or mutual, or left to the application layer is to be discussed and decided in the conclusion.

## 5.6 Key issue #6: Exposure of Inventory Device Quantity

### 5.6.1 Key issue details

The inventory service is a fundamental process for AIoT devices, which includes both "inventory only" and "inventory and command" cases. In both scenarios, the mandatory steps involve AIoT paging and Device ID transmission. The AIoT paging message may contain an ID of a single A-IoT device, a group ID that maps to multiple A-IoT devices, or multiple IDs of A-IoT devices. If AIoT paging message does not contain an ID, it will map to all the A-IoT devices. After these steps, the network can calculate the quantity of device IDs for this inventory.

The inventory device quantity may contain business information, such as the quantity of stock in a shopping mall. If this information falls into the hands of competitors, they may adjust their sales strategy to attract more customers from that shopping mall.

### 5.6.2 Security threats

By broadcasting a fake inventory message with a group ID, an attacker could potentially calculate the quantity of devices in a group by observing the differences in reported device IDs, even if the IDs are encrypted. This could lead to the exposure of the inventory device quantity associated with the group ID. For example, in a shopping mall, assuming the attacker has knowledge of the link between the group ID and goods (such as knowledge of the link between SUPI and the real subscriber), the attacker could use a fake reader to broadcast this group ID. Subsequently, the attacker would receive multiple device IDs and calculate the device quantity for this group ID, allowing them to determine the number of specific goods.

An attacker could calculate the quantity of all devices by observing differences in reported device IDs, even if the IDs are encrypted, after sending a fake inventory message without any IDs. This could result in the exposure of the inventory device quantity within an area. For instance, in a shopping mall, if an attacker can control the broadcast scope into the shopping mall, they could utilize a fake reader to broadcast an inventory message without any ID. Subsequently, the attacker would receive multiple device IDs, enabling them to calculate the device quantity in this area and determine the stock levels of all the goods of this shopping mall.

### 5.6.3 Potential security requirements

TBA.

## 5.X Key Issue #X: <Key Issue Name>

### 5.X.1 Key issue details

### 5.X.2 Security threats

### 5.X.3 Potential security requirements

# 6 Solutions

Editor’s Note: This clause contains the proposed solutions addressing the identified key issues.

## 6.0 Mapping of solutions to key issues

Table 6.1-1: Mapping of solutions to key issues

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Solutions | KI#1 | KI#2 | KI#3 | KI#4 | KI#5 |
| **1** | X |  |  |  |  |
| **2** |  | X |  |  |  |
| **3** |  | X |  |  |  |
| **4** |  |  |  |  | X |
| **5** | X |  |  |  |  |
| **6** |  |  |  |  | X |
| **7** |  |  |  |  | X |
| **8** |  |  |  |  | X |
| **9** |  |  |  |  | X |
| **10** |  |  |  |  | X |
| **11** |  |  | X | X | X |
| **12** |  |  | X | X | X |
| **13** |  |  | X | X | X |
| **14** |  |  |  | X |  |
| **15** |  |  |  | X |  |
| **16** | X |  |  |  |  |
| **17** | X |  |  |  |  |
| **18** |  | X |  |  |  |
| **19** |  | X |  |  |  |
| **20** |  |  | X |  |  |
| **21** |  |  | X |  |  |
| **22** |  |  | X |  |  |
| **23** |  |  | X |  |  |
| **24** |  |  | X |  |  |
| **25** |  |  | X |  |  |
| **26** |  |  | X |  |  |
| **27** |  |  | X |  |  |
| **28** |  |  | X |  |  |
| **29** |  |  | X |  |  |
| **30** |  |  | X |  |  |
| **31** |  |  | X |  |  |
|  |  |  |  |  |  |

Editor’s Note: Each solution should be mapped here.

## 6.1 Solution #1: Ambient IoT device disabling mechanism

### 6.1.1 Introduction

This solution addresses KI#1.

According to TS 22.369, the enable/disable device operations are used by the network operator to manage the Ambient IoT device’s capability to transmit RF signals. As the disabling of RF transmission capability could, according to the operator’s policy, be temporary or permanent, it is paramount to ensure that the disabling, specifically of a permanent nature, is performed securely and in a manner that allows device recovery in case the system was compromised, and an attacker has managed to issue “disable” commands to one or multiple Ambient IoT devices.

### 6.1.2 Solution details

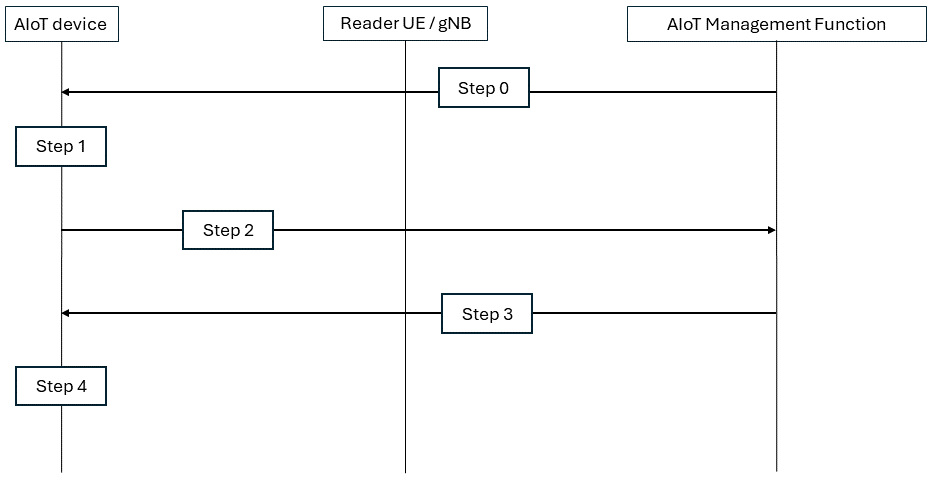


Figure 6.1.1 – Ambient IoT device disabling mechanism

The permanent disabling of an AIoT device is performed in a two-stage operation, where initially, the AIoT device is temporarily disabled, and then, following a cool-down period (i.e., recovery time window), the AIoT device could be disabled permanently. The two-stage permanent disabling operation is performed as follows:

In Step 0, the AIoT device is provisioned with a configuration determining how the device processes disabling operations. The configuration includes the required cool down period that needs to be met before a permanent “disable” command is allowed.

In Step 1, the AIoT managing function issues a temporary disable command to the Ambient IoT device. The command includes a counter T1.

Note 1: The AIoT Management Function (AIoT MF) depends on the entity that owns or manages the resource (i.e., AIoT device). If a device is owned or managed by the network, the disabling is triggered by a CN function, otherwise, it is triggered by a 3rd party Management Function managing the device.

Editor’s Note: Whether the solution aligns with SA2 system architecture and procedures is FFS.

In Step 2, The AIoT device, upon receiving the temporary disable command, retrieves and stores the counter T1, which will be used in subsequent processing.

In Step 3, the AIoT device sends an ACK to the AIoT MF, which may contain the counter received in Step 0. Then, the AIoT device temporarily disables its RF transmission capability..

In Step 4, depending on whether the AIoT MF intends to re-enable the AIoT device (i.e., recovery scenario), or permanently disable the AIoT device. The AIoT MF sends in:

- Step 4a: a message containing an enable command to recover the AIoT device, or

- Step 4b: a message containing a permanent disable command, in which a second counter T2 is included.

Note 2: The messages in step1 and step3 carrying the temporary and permanent disable commands are protected using the same means of protection applicable to other commands (e.g., write), and so is the verification that the disable commands are coming from a legitimate party.

Editor’s Note: How inventory/authentication is performed before the permanent disable command is FFS.

In Step 5, The AIoT device processes the received command. If the AIoT is requested to enable its RF transmission capability, the AIoT device recovers from the temporary disabled state and discards the stored counter T1. Otherwise, if the AIoT device is requested to perform a permanent disable command, it retrieves the second counter from the message, then checks whether the following conditions are met:

- The RF transmission capability is temporarily disabled.

- Whether the value T2 – T1 is greater or equal to the cool-down period configured in the AIoT device.

Editor’s Note: Whether the AIoT device can maintain an internal state is FFS.

If the checks succeed, the AIoT device temporarily enables its RF transmission capability to send an ACK message to the AIoT MF in step 6 confirming that the device is permanently disabling its RF transmission capability.

In Step 7, the AIoT device disables its RF transmission capability permanently.

### 6.1.3 Evaluation

Editor’s Note: How the solution addresses the security threats and requirement of KI#1 is FFS.

In the solution proposed, the protection of disable commands (temporary and permanent) re-uses the same means of protection applicable to other commands (e.g., write command).

Ambient IoT device(s) need to maintain an internal state to support the security mechanism proposed by the solution.

Editor’s Note: Further evaluation is FFS.

## 6.2 Solution #2:PCF based Service Authorization and Provisioning to UE

### 6.2.1 Introduction

This solution addresses the KI#2 Authorization for 5G Ambient IoT services.

### 6.2.2 Solution details

This solution propose to reuse the existing mechanism for 5G Prose U2N relay as specified in TS 23.304[x] with following changes:

A UE acting as an intermediate node is registered with 5GC using the existing mechanism, with some enhancements to indicate its capability of acting as an intermediate node regardless of for which AIoT device, and is authorized as an intermediate node (UE) during the registration procedure.

For PCF based Service Authorization and Provisioning to UE, the Registration procedures as defined in clause 4.2.2.2 of TS 23.502 [x], UE Policy Association Establishment procedure as defined in clause 4.16.11 of TS 23.502 [x] and UE Policy Association Modification procedure as defined in clause 4.16.12 of TS 23.502 [x] apply with the following additions:

- If the UE indicates AIoT Capability in the Registration Request message and if the UE is authorized to use 5G AIoT service based on subscription data, the AMF selects the PCF which supports AIoT Policy/Parameter provisioning and establishes a UE policy association with the PCF for AIoT Policy/Parameter delivery.The AMF reports the authorized AIoT Capability to the selected PCF, which may determine the AIoT Policy/Parameter based on the UE's authorized AIoT Capability.

This solution based on the assumption that PCF hold the Authorization information and the AIoT capabiltiy can be include in the Registration Request message.

### 6.2.3 Evaluation

TBD

## 6.3 Solution #3: Authorization of Intermediate UE for AIoT services

### 6.3.1 Introduction

The solution addresses the security requirement of KI#2: Authorization for 5G Ambient IoT services. Specifically, this solution proposes a method to authorize the UE as Intermediate UE in AF-initiated AIoT service procedure. The AMF/AIoT NF select the UE based on the information provided by AF, e.g., location information or external UE ID, and then interact with the UDM to obtain the selected UE’s subscription data and check whether it is allowed to act as Intermediate UE for AIoT secvice. Only after the UE is successfully authorized as intermediate UE, the network will then perform subsequent AIoT service procedure.

### 6.3.2 Solution details

Depicted in Figure 6.3.2-1 is the authorization procedure of Intermediate UE for AIoT Services.



**Figure** **6.3.2-1: Authorization of Intermediate UE for AIoT service**

1. The UE performs the registration procedure as specified in TS 23.502 [x] with the enhancement to indicate its AIoT Intermediate node capability, and is authorized as an intermediate UE during the registration procedure.
2. The AF sends the AIoT Service Request to the AMF/AIoT NF via the NEF, including the AIoT device ID, seivice type (e.g., Inventory, Command), location information, external UE ID (GPSI).
3. The AMF/AIoT NF selects the Intermediate UE based on the information provided by AF, e.g., location information and/or GPSI, etc.

NOTE1: The selection of Intermediate UE is up to SA2 WG decision.

1. The AMF/AIoT NF sends the UE Authorization Request to the UDM with the info of the selected UE.
2. The UDM checks whether the selected UE is allowed to act as Intermediate UE against the UE's subscription data for AIoT service.

NOTE2: The relevant subscription data could be configured offline in the UDM, or provided and updated in the UDM based on the AF-initiated AIoT service requests.

1. The UDM returns the UE Authorization Response to the AMF/AIoT NF.
2. The AMF/AIoT NF sends the AIoT Service Request to the Intermediate UE, including the AIoT device ID, service type, authorized result.
3. The inventory/Command procedure is carried out.

Editor’s Note: Whether the Intermediate UE is authorized during the registration or after the Intermediate UE selection is FFS.

Editor’s Note: Which entity performs the Intermediate UE authorization should be aligned with the AIoT system designed by SA2, which is FFS.

### 6.3.3 Evaluation

TBD

## 6.4 Solution #4: Protection for inventory and command procedure

### 6.4.1 Introduction

This solution addresses key issue on authentication, key issue on information protection and key issue on protection for disabling device operation.

The solution is assumed to be used in dedicated network for ambient IoT service. It is assumed that every AIoT Device is preconfigured with a pre-shared key, while the details of secure storage and processing of credentials e.g. pre-shared key is out of scope of this solution. Accordingly, it is assumed that such a key is also preconfigured/stored in the network side along-side the device ID.

### The solution provides a new AIoT device authentication procedure that could be used for both the inventory as well as the command procedures. 6.4.2 Solution details

### 6.4.2.1 Protection for inventory-only procedure

Figure 6.4.2-1: Information Flow for protection of inventory-only service



1. AF sends AIoT service operation request to NEF, including the AF ID, device information, and inventory operation.

2. NEF sends the AIoT service operation request to the selected AMF/AIoTMF.

3. The AMF/AIoTMF selects the Ambient IoT capable RANs. In addition, the Nonce1 is generated, which is used as fresh parameter for authentication. For each of the selected reader, the AMF/AIoTMF sends the request message with the generated Nonce1.

4. Upon reception of the request message, Reader executes inventory by triggering the Paging-like procedures with Nonce1 towards the AIoT devices.

5. The AIoT device performs random access like procedures to establish the connection with reader.

6. The AIoT device generates Nonce2 as the fresh parameter from device side. When deriving Auth\_token, Nonce1 and Nonce2 are used to form the input S to the KDF and the pre-shared key is used as input key. Auth\_token is used for network to authenticate device.

7-8. The AIoT Device sends the AIoT Device ID, Nonce2 and Auth\_token to Reader. In the following step, Reader reports the AIoT Device ID, Nonce2 and Auth\_token to AMF/AIoTMF.

AIoT Device uses registers for temporarily keeping the nonces required for command protection.

9. AMF/AIoTMF sends request message to authentication server for device verification, which includes AIoT Device ID, Nonce1 and Nonce2.

10. XAuth\_token is calculated with same input as in device side.

11. Authentication server sends response message to AMF/AIoTMF, which includes XAuth\_token.

12. AMF/AIoTMF compares Auth\_token and XAuth\_token.

NOTE: As an alternative, authentication server can verify the Auth\_token and send the authentication result to AMF/AIoTMF.

13-14. If the verification succeeds, the AMF/AIoTMF reports the AIoT Device ID to NEF by sending the AIoT\_ Notify message. The NEF forwards the received information to AF by sending the AIoT\_ Notify message.

### 6.4.2.2 Protection for inventory and command procedure

Figure 6.4.2-2: Information Flow for protection of information transfer for AIoT services



Compared with the inventory-only procedure in clause 6.4.2.1, the protection for command is enhanced with the following modification.

10. In addition to the XAuth\_token calculation, the session key Ks is derived if receiving additional indication from AIoT Function. The Nonce1 and Nonce2 is used to form the input S to the KDF and the pre-shared key is used as input key to derive the session key Ks. The Ks is different with XAuth\_token as different input (e.g., FC value).

11. Session key is also included. If the device security capability (e.g., AES-128 algorithm) is stored in AIoT\_ UDM, it will also be sent to AMF/AIoTMF.

12. The command message is protected with session key. For example, the MAC1 will be calculated to protect the integrity of the message. The algorithm will be selected based on device security capability.13-14. The AMF/AIoTMF sends the protected AIoT Command message to the AIoT Device, which includes encrypted data and MAC1. No AIoT device ID is required to be included in step 14, as other ID will be used between device and reader, which is discussed and decided in RAN groups. It is assumed that the device receives the command message while energy is available in energy storage.

15-16. The device will receive the message in step 14. Integrity verification and decryption will be performed. After successful verification, the device will follow the command operation. The uplink message will also be sent from device to network if needed. For example, if the command is “read” operation, data2 to be reported will be included after encryption. The message will also be integrity protected.

17-18. The AMF/AIoTMF reports the result of the AIoT service operation request to the NEF by sending the AIoT\_ notification message.

Editor’s Note: The information flow will be updated based on the conclusion regarding command operation procedure in SA2.

### 6.4.2.3 Auth\_token and XAuth\_token derivation function

The following parameters are used to form the input S to the KDF.

- FC = 0xaa,

- P0 = Nonce1,

- L0 = length of the Nonce1(i.e. 0x00 0x10),

- P1 = Nonce2,

- L1 = length of Nonce2 (i.e. 0x00 0x10),

The input key KEY shall be equal to the pre-shared key.

The (X)Auth\_token is identified with the 128 least significant bits of the output of the KDF.

### 6.4.3 Evaluation

Challenge-based symmetric algorithm is used for security protection. The privacy of device identifier in uplink and downlink is not addressed in this solution.

This solution assumes there is preconfigured pre-shared key as the authentication credential stored in AIoT device and network side securely.

For inventory-only case, one-way device authentication is provided. For inventory and command case, mutual authentication is supported, in which the network authentication in device is implicitly provided by MAC verification.

Editor’s Note: the impact to include nonce in paging message is FFS.

Editor’s Note: whether replay attacks are possible against the device or the network is FFS.

## Editor’s Note: whether additional authentication token is required for device to authenticate network is FFS.6.5 Solution #5: Disabling and Enabling AIoT Device

### 6.5.1 Introduction

This solution addresses KI#1 to disable or enable the RF transmission capabilities of an AIoT device according to operator’s security policy. Solution makes the following assumptions:

1. Disabling/enabling instruction is to be sent as part of inventory and command procedure if command only procedure is not supported. If command only procedure is supported, the solution can be part of the command only procedure.

2. Since part of the command case is to carry instructions for the AIoT device to perform some specific functions, it is also assumed that the AIoT device and the network has established security to protect the inventory and command procedure and that the command procedure is confidentiality-, integrity-, or replay-protected. The protection of either the inventory or command procedure is not in scope of this solution.

3. For a group of AIoT device, solution assume that the group can be paged (i.e., via inventory procedure) if the paing ID corresponds to a group ID and that disable/enable code is the same for the group.

NOTE 1: While AIoT device’s RF transmission capabilities are disabled, the device is still able to receive RF signals.

NOTE 2: Protection of command procedure is recommended.

### 6.5.2 Solution details



1. Inventory procedure is performed between the AIoT device and the network (CN or AF).

NOTE: Whether Command operation procedure is stand-alone or is coupled with inventory and command is to be aligned with RAN2.

2. Depending on the business model and security protection, the Command Operation Trigger containing Code1 and Code2 can come from third-party AF or within the 3GPP network. If third-party AF initiates the Command Operation Trigger, the request is routed through to the 3GPP core network, in which case, the AF is authorized by the 3GPP network.

3. The AIoT Function/AMF (either initiated by 3GPP network or as a result of request from third-party AF) sends a Command Operation Trigger with Code1 and Code2 to the AIoT Reader (RAN node or an AIoT-capable UE).

4. The AIoT Reader (RAN node or an AIoT-capable UE) sends the Command Operation Request to the AIoT device.

5. The AIoT Device checks Code1 and Code2 against the code stored in the AIoT device, using the following logic:

If Code1 and Code2 are the same and that they match the code stored in the AIoT device, the AIoT device permanently disables its RF transmission capabilities.

If Code1 and Code2 are different, and Code1 matches the code stored in the AIoT device, the AIoT device temporarily disables its RF transmission capabilities if its RF transmission capabilities are enabled, or if the AIoT device’s RF transmission capabilities are temporarily disabled, then the AIoT device re-enables its RF transmission capabilities. Furthermore, the AIoT device replaces its stored code with the value of Code2 for future instructions.

If Code1 does not match the code stroed in the AIoT device, the Command Operation Request is discarded.

Editor’s Note: How the disable/enable code between the AIoT device and the network is synchronized is FFS.

6, 7, 8. In the case when the AIoT device’s RF capabilities are re-enabled, the AIoT device may reply with a Command Operation Reply.

### 6.5.3 Evaluation

Solution fully addresses security requirement in KI#1. The solution does not address the security of the Command Operation procedure as it is part of communication security between AIoT device and the network.

The solution requires the AIoT device to be provisioned with a code to disable or re-enable RF transmission capabilities. The codes from the network (either CN or AF) are sent as part of secured Command Operation procedure and are secure from attackers. Replacing the code in the AIoT device after each temporary disabling/enabling RF transmission capabilities using the Command Operation procedure also prevents the code being replayed.

## 6.6 Solution #6: AIoT device authentication

### 6.6.1 Introduction

This solution addresses KI#5.

This solution propose an AIoT device Security Management and an AIoT controller. The AIoT device Security Management is a core network function, has the following security capabilities:  
 1.Stores the initial Kaiot together with the AIoT device. This key is used for MAC calculation.

1. Sends the Kaiot to AIoT controller to calculate the network MAC.

AIoT controller has capability about calculating the network MAC and verifying the device MAC.

AIoT device has capability about calculating the device MAC and verifying the network MAC.

The counter is maintained by the AIoT controller and the AIoT device.

### 6.6.2 Solution details

The following figure shows the call flow for AIoT device authentication.



Figure 6.6.2-1: AIoT device authentication

0. AIoT device and AIoT device Security management both store the Kaiot. AIoT device stores Device ID and initial counter. The AIoT device Security management stores the mapping relationship between the Device ID and reader ID.

1. AF sends Inventory(or command) request to AIoT controller, which includes Reader ID and Device ID. The Reader ID indicates the reader requested for Inventory(or command). Multiple Reader IDs can be included. The Device ID indicates the AIoT device for Inventory(or command). Multiple Device IDs can be included.

2. AIoT controller sends Device information request to AIoT device Security management, which includes Device ID received in step 1. If multiple Device IDs are received in step 1, AIoT controller can repeat this step per Device ID for multiple times, or AIoT controller can send the Device ID list to AIoT device Security management in the same message.

3. AIoT device Security management sends Device information response to AIoT controller, which includes the reader ID, and Kaiot. If Multiple Device IDs are received in step 2, this message may include all device information corresponding to the Device IDs.

4. The AIoT controller calculates the network MACn based on the Kaiot and counter, for example, MACn= HASH (device ID, reader ID, counter, Kaiot).

5. The AIoT controller sends the Inventory(or command) Request to the reader(s) base on reader ID in step 3, which includes Device ID and MACn. If the counter is initial value, which includes start indication.

6. The Reader(s) send Inventory(or command) Request to AIoT device, which includes Device ID and MACn, and which may includes start indication.

7. The AIoT device verifies the MACn, that is, the AIoT device calculates the local MACn', for example, MACn’=HMAC(device ID, reade ID, counter), the input key is Kaiot. The counter is the counter stored in the AIoT device. If start indication is valid, the the counter is initial counter value. If MACn and MACn' are the same, the verification succeeds. The AIoT device increases the counter by one, and store the new counter. The AIoT device can optionally calculate the MACu, for example, MACu= HMAC (device ID, reade ID, new counter), the input key is Kaiot, where Kaiot is the Kaiot stored on the AIoT device.

8. The AIoT device sends Inventory(or command) Response to the Reader with its Device ID, which may includes MACu.

9. The Reader sends Inventory(or command) Response to the AIoT controller with the Device ID and Reader ID, which may includes MACu.

10. The AIoT controller calculates the MACu' like the AIoT device, and verifies the MACu'. If the verification is successful, the Inventory(or command) is successful. The AIoT controller increases the counter by one. The AIoT controller stores the new counter.

11. AIoT controller stores the Device ID and the Reader ID, if multiple Inventory(or command) Responses are received from different Readers, the AIoT controller stores the Device ID and multiple Reader IDs. AIoT controller sends the Inventory(or command) Response to AF with the Device ID and Reader ID(s).

Editor’s Note: How the AIoT device get reader ID in step 7 is FFS.

Editor’s Note:if maintaining device ID and reader ID mapping is required or not, and if required how it is done is FFS.

Editor’s Note: It is FFS how synchronization of counter between devices and Controller is ensured and how to recover from the even of de-synchronization.

Editor’s Note: The impact of sharing key Kaiot with AIoT controller is FFS.

Editor’s Note: How the AIoT controller verifes the MAC if there is multiple response in step 10.

Editor’s Note: It is FFS how to protect privacy of device identifier in the step 6 is FFS.

Editor’s Note: It is FFS if replay attacks are possible against the device or the network.

Editor’s Note:It is FFS if K needs confidentialkity/integrity protection against physical attack in the device.

### 6.6.3 Evaluation

TBD.

6.7 Solution #7: Lightweight AIoT Authentication solution

### 6.7.1 Introduction

The assumption of this solution is AIoT device and network sharing a long-term root key K[x].

MAC/XMAC is used for device authenticating network. MAC is calculated by network side with the K, XMAC is calculated by AIoT device side with the K.

Optionally, RES/XRES is used for network authenticating device. XRES is calculated by network side with the K, RES is calculated by device side with the K.

To provide the freshness of the authentication vectors, RAND is also used to calculate MAC/XMAC and optionally RES/XRES.

### 6.7.2 Details

MAC/XMAC is used for device authenticating network. MAC is calculated by network side with the K, XMAC is calculated by AIoT device side with the K.

Optionally, RES/XRES is used for network authenticating device. XRES is calculated by network side with the K, RES is calculated by device side with the K.

To provide the freshness of the authentication vectors, RAND is also used to calculate MAC/XMAC and optionally RES/XRES.

Similar with 5G-AKA, and based on RAN2 agreement for “inventory and command” case[6], AIoT authentication solution has the following steps:



Figure 6.7.2-1 AIoT Authentication based on 5G-AKA

1. AIoT NF sends authentication vector request to UDM/ARPF for AIoT device.
2. UDM/ARPF calculates MAC with K (e.g. the root key of AIoT device) and RAND.
3. Optionally, UDM/ARPF calculates XRES with K and RAND, if network wants to authenticate AIoT device.
4. UDM/ARPF sends RAND, MAC, device ID and optionally XRES to AIoT Authentication Function.
5. AIoT Authentication Function sends Authentication Request including RAND, MAC, device ID to AIoT NF.
6. AIoT NF sends Authentication Request including RAND, MAC, device ID to Reader.
7. Reader sends Authentication Request including RAND and MAC to AIoT device. Authentication Request is carried by Step C in the “inventory and command” case.
8. AIoT device calculates XMAC with RAND and K.
9. Optionally, AIoT device calculates RES with K and RAND.
10. AIoT device verifies XMAC=MAC, then the network authentication is successful. If there is command, only after successful verification, the command can be proceeded.
11. AIoT device sends Authentication Reponses to Reader, this message optionally including RES if network wants to authenticate AIoT device. Authentication Reponses is carried by Step D in the “inventory and command” case.
12. Reader sends Authentication Reponses to AIoT NF.
13. Optionally, AIoT NF sends Uplink Authentication Request including RES to AIoT Authentication Function.
14. Optionally, AIoT Authentication Function verifies XRES=RES, then the AIoT device Authentication is successful.

As an example, the generation of MAC/XMAC and RES/XRES is described as below.



Figure 6.7.2-2 The generation of MAC and XRES at network side.



Figure 6.7.2-3 The generation of MAC and XRES at AIoT device side.

Editor's Note: Further evaluation of the usage of the crypto primitives is FFS.

Editor's Note: The impact of removing the use of SEQNO (e.g., replay against device or network) from AKA is FFS

Editor's Note: Anonymity key (AK) is used to protect SEQNO in AKA. If SEQNO is not used, the necessity of using AK to compute MAC and XRES is FFS.

Editor's Note: It is FFS if the shared key needs confidentiality/integrity protection against physical attack in the device

Editor's Note: When the AIoT NF triggers the authentication is FFS.

Editor's Note: The security level of MAC/RES calculation compare to 5G-AKA is FFS.

Editor's Note: The intermediate authentication parameter storage in AIoT device is FFS.

### 6.7.3 Evaluation

TBA.

## 6.8 Solution #8: Mutual authentication for AIoT system

### 6.8.1 Introduction

This solution addresses key issue#5: “Authentication in Ambient IoT service.”

### 6.8.2 Details

Preassumption:

1. AIoT AUSF and AIoT UDM are independent network entities for AIoT system, they can also be collocated with legacy AUSF and UDM, depending on operators’ deployment.
2. The Applacation Function is out of operator domain, which can be the AIoT device manufacture.
3. It is assumed AIoT UDM owns the AIoT device credentials, while Application Function has no access to those credentials.
4. Step 1-3 are following RAN2 procedure using 2-step RACH.

A screenshot of a computer

Description automatically generated

Step 0a. the AIoT device is configured with Device ID and K as the root key in the manufacturing time.

Step 0b. AIoT UDM is configured with K as the root key for the device. Every AIoT device owns a unique K.

Step 0c. The AF may sync up with AIoT UDM the device IDs after manufacturing.

Step 1. Reader sends the paging message including the device ID(s).

NOTE: what IDs to be included will be decided by RAN2.

Step 2-3. AIoT device sends random ID, device ID in Msg1. Reader echoes back the random ID in Msg2.

Step 4. AIoTF triggers the authentication through Reader.

Step 5. AIoT device calculates the RES using K and device ID and RAND, using HASH function.

Step 6. AIoT device sends the device ID, RES, and random number Counter to Reader in uplink AS message.

NOTE: The message details to be dependent on SA2 decision.

Step 7-8. Reader sends the Authentication Request to AIoT AUSF for authentication.

Step 9. AIoT AUSF sends authentication request to AIoT UDM including the device ID and Counter.

Step 10, AIoT UDM calculates the XRES and network authentication Token. AIoT UDM then sends both values back to AIoT AUSF.

Step 11. AIoT AUSF compares the RES and XRES, if they are equal, AIoT AUSF sends network authentication Token back to Reader. Otherwise, the authentication fails.

Step 12-13. the AIoT AUSF passes the authentication result and network authentication Token back to Reader.

Step 14. If the authentication success, the Reader continue with step 15, otherwise, the Reader ceases the authentication procedure.

Step 15. Reader sends the network authentication Token to AIoT devices in DL command message.

Step16. AIoT device verifies if the network authentication Token is correct. If yes, then the authentication of the network is successful, and AIoT device continue with the subsequent UL message. Otherwise, the AIoT device will cease the procedure.

Editor’s Note: The synchronization issue is FFS.

Editor’s Note: It is FFS when the network triggers the authentication.

Editor’s Note: It is FFS how AIoT device maintain the authentication status.

Editor’s Note: The call flow needs to be updated.

Editor’s Note: The architecture of AIoT is FFS, based on SA2 progress.

### 6.8.3 Evaluation

Editor’s Note: The impact on device and network is FFS.

## 6.9 Solution #9: Device authentication and data communication security

### 6.9.1 Introduction

This solution tries to address the Key issue #3, Key issue #4 and Key issue #5.

This solution protects the privacy of AIoT Device ID by encrypting AIoT Device ID or calculating temporary ID using AIoT device authentication keys.

For AIoT Device authentication, only entities (AIoT Device or network) that possess the AIoT Device authentication key can successfully decrypt or verify the protected AIoT Device ID.

For communication security, only entities (AIoT Device or network) that possess the AIoT Device authentication key can generate session key. For each data transmission, both network nonce and AIoT Device nonce are used to protect the communication content.

### 6.9.2 Solution details

The security procedure for command operations is shown in the following figure.

 Figure 6.9. 2-1: Security procedure for command operations

0. During the AIoT Device initialization phase, the AIoT Device Authentication Key is pre-configured in the AIoT Device. The AIoT Device Authentication Key is used to authenticate the device and secure the communication with the AIoT Device. Security policies for AIoT Device ID protection and communication may also be pre-configured.

1. The AF sends a Command Operation Request to the Ambient IoT Function (AIoTF). The request includes the AIoT Device ID and Payload.

2. To authenticate an AIoT Device, the AIoTF performs the following operations:

Generate a Network Nonce;

Use the Device Authentication Key and Network Nonce to derivate keys for confidentiality and/or integrity protection;

Protect the AIoT Device ID with the new derived keys to obtain the Protected AIoT Device ID; The Protected AIoT Device ID can also be a temporary ID generated using the authentication key and security parameters;

Send a Paging message to the AIoT Devices through the AIoT Reader. The message includes the Protected AIoT Device ID and Network Nonce.

3. The AIoT Device performs the following operations:

Use the same method as the AIoTF to derive the keys, and then decrypts and/or verifies the Protected AIoT Device ID or calculate the temporary ID;

Check if the Device ID carried in the paging message matches the locally stored Device ID or locally calculated temporary ID;

Generate a Device Nonce;

Use the Device Authentication Key, Network Nonce and Device Nonce to derivate keys for Device ID confidentiality and/or integrity protection;

Protect the AIoT Device ID with the new derived keys to obtain the Protected AIoT Device ID\*; The Protected AIoT Device ID\* can also be a temporary ID generated using the authentication key and security parameters;

The AIoT Device returns the Protected Device ID\* and Device Nonce to the AIoTF through the AIoT Reader. It may also return its Device capability.

4. The AIoTF performs the following operations:

Use the same method as the AIoT Device to derive the keys, and then decrypts and/or verifies the Protected AIoT Device ID\* or calculate the temporary ID;

Check if the Device ID carried in the authentication request matches the locally stored Device ID or locally calculated temporary ID; If they match, the authentication is successful.

5. To protect a command message, the AIoTF performs the following operations:

Generate a Network Nonce;

Use the Device Authentication Key and Network Nonce to derivate keys for confidentiality and/or integrity protection;

Protect the AIoT Device ID with the new derived keys to obtain the Protected AIoT Device ID; The Protected AIoT Device ID can also be a temporary ID generated using the authentication key and security parameters;

Protect the Payload and/or the entire message using the new derived keys according to the Security Policy that specifies how the communication message is protected;

Send Command message to the AIoT Device through the AIoT Reader. The request includes Protected AIoT Device ID, Security Policy, Network Nonce, Secured Payload and MAC.

6. The AIoT Device performs the following operations:

Use the same method as the AIoTF to derive the keys;

Use the same method as step 3 to check the protected AIoT device ID;

Decrypts and/or verifies the Secured Payload and/or the entire message according to the Security Policy in the message; If the verification is successful, the AIoT Device continues to perform the following operations;

Generate a Device Nonce;

Use the Device Authentication Key, Network Nonce and Device Nonce to derivate keys for confidentiality and/or integrity protection;

Protect the Payload and/or the entire message using the new derived keys according to the Security Policy (either from the command request message or from the pre-configured security policy);

Send Command Response Message to the AIoTF through the AIoT Reader. The response includes Security policy, Device Nonce, Secured payload and MAC.

7. The AIoTF performs the following operations:

Use the same method as the AIoT Device to derive the keys, and then decrypts and/or verifies the secured Payload and/or the entire message; If the verification is successful, the AIoTF continue to perform the following operations;

Return the Payload to the AF.

Editor's Note: It is FFS if generating device Nonce twice makes the protocol computationally correct.

Editor's Note: It is FFS if the shared key needs confidentiality/integrity protection against physical attack in the device.

Editor's Note: It is FFS how to prevent power and resource exhaustion in the AIoT devices when all of devices in the paging area always have to decrypt the Device ID.

### 6.9.3 Evaluation

TBD

6.10 Solution #10: Authentication for AIoT device

### 6.10.1 Introduction

This solution addresses KI#5 to propose 3 authentication procedures for AIoT devices including UE reader and RAN reader cases.

Editor’s Note: Clarification on using mechanism described in RFC 4739 is ffs.

Editor’s Note: Clarification on step 4 in 6.10.2.1.1 and step 5 in 6.10.2.1.2 and 6.10.2.2 is ffs.

Editor’s Note: Clarification on example for different cases is ffs.

Editor’s Note: Whether Auth Container can be sent to the device via paging is ffs.

### 6.10.2 Solution details

#### 6.102.1 UE reader case

##### 6.10.2.1.1 Alternative 1 – UE reader granularity

The following figure shows the call flow for AIoT device authentication via UE reader with UE reader granularity. This procedure has the following main points:

1. The operator is responsible for authorizing UE readers to provide AIoT services. This means that the operator does not manage individual AIoT subscriptions, but rather focuses on UE reader subscriptions. This allows the AIoT service to operate within licensed spectrum, with billing based on UE reader usage (e.g. per AIoT service) rather than the AIoT devices themselves.
2. As there is only SLA between the UE readers and operators, the operator does not require knowledge of AIoT authentication. Therefore, there is no specification of authentication between AAA and AIoT, and no need for SLA and trust establishment between AAA and 5GC. Instead, authentication is encapsulated in an Authentication Container.



Figure 6.102.1.1-1: Authentication for AIoT device via UE reader with UE reader granularity

0. The UE reader has established a PDU Session for authentication. The 5GC authorizes the PDU Session establishment procedure, e.g., based on subscription data.

1. The AIoT AF sends Inventory command to the AIoT NF/AMF via NEF.

2. The AIoT NF/AMF selects the UE reader for the inventory.

3. The AIoT NF/AMF interacts with UDM/UDR to authorize the UE reader for the inventory, e.g., whether the UE reader is allowed to serve the AF for the inventory for the AIoT devices. If authorization succeeds, the UDM/UDR returns authorization information (e.g. AAA address) to the UE reader.

4. In case the authorization succeeds, the AIoT NF/AMF responds to the AIoT AF via NEF.

5. In case the authorization succeeds, the AIoT NF/AMF sends AIoT paging command encapsulated in a DL NAS Transport message to the UE reader. The authorization information may be included in the DL NAS Transport message.

6. The UE reader may interact with AAA-S based on the authorization information over the PDU Session before paging AIoT devices, e.g., using mechanism described in RFC 4739 [7]. The AAA-S may return authentication information to the UE reader.

7. The UE reader broadcasts AIoT paging message, which may include an Auth Container that contains authentication information.

8. The AIoT device determines to responds to the AIoT paging message, it sends an AIoT message to the UE reader with Device ID and optional another Auth Container, which contains information for authentication.

9. The UE reader acts as proxy of the AIoT device to interact with AAA-S over the PDU Session for authentication between the AIoT device and the AAA-S, e.g., using EAP framework or mechanism described in RFC 4739 [7]. The UE reader may further interact with the AIoT device for authentication if required by AAA-S.

10. In case the authentication succeeds, the UE reader will receive authentication success indication from AAA-S. The UE reader reports the inventory result to the AIoT AF.

As an example, when AKA based authentication method is used, then step 6 is not performed, and there is no Auth Container in steps 7 and 8, and in step 9, the RAND and AUTN will be encapsulated into an Auth Container sent from the UE reader to the AIoT device, the RES will be encapsulated into another Auth Container sent from the AIoT device to the UE reader.

##### 6.10.2.1.2 Alternative 2 – AIoT device granularity

The following figure shows the call flow for AIoT device authentication via UE reader with AIoT device granularity. This procedure has the following main points:

1. The operator is responsible for managing AIoT subscriptions, but the AIoT credentials are stored in AAA. That means the operator needs to trust authentication result from AAA (i.e. there is SLA between operators and external AAA owner, or operator manage the AAA server). This allows operator to bill based on AIoT devices themselves.
2. It is also assumed that AIoT credentials are stored in AAA with specific authentication method, or AIoT device and AAA are within the same network domain or the same vendor. Therefore, there is no specification of authentication between AAA and AIoT. Instead, authentication is encapsulated in an Authentication Container.



Figure 6.10.2.1.2-1: Authentication for AIoT device via UE reader with AIoT device granularity

0. The UE reader has registered into 5G network.

1-3. The same as described in steps 1-3 of clause 6.10.2.1.1.

4. In case the authorization succeeds, the AIoT NF/AMF may interact with AAA-S before paging AIoT devices, e.g., using mechanism described in RFC 4739[7]. The AAA-S may return authentication information to the AIoT NF/AMF.

5. In case the authorization succeeds, the AIoT NF/AMF responds to the AIoT AF via NEF.

6. In case the authorization succeeds, the AIoT NF/AMF sends AIoT paging command encapsulated in a DL NAS Transport message to the UE reader. The authorization information may be included in the DL NAS Transport message. The paging command may include an Auth Container that contains authentication information.

7. The UE reader broadcasts AIoT paging message, which may include the Auth Container.

8. The AIoT device determines to responds to the AIoT paging message, it sends an AIoT message to the UE reader with Device ID and optional another Auth Container, which contains information for authentication. The UE reader forwards the AIoT message encapsulated in a UL NAS Transport message to the AIoT NF/AMF.

9. The AIoT NF/AMF acts as proxy of the AIoT device to interact with AAA-S for authentication between the AIoT device and the AAA-S, e.g., using EAP framework or mechanism described in RFC 4739 [7]. The AIoT NF/AMF may further interact with the AIoT device via the UE reader for authentication if required by AAA-S.

10. In case the authentication succeeds, the AIoT NF/AMF will receive authentication success indication from AAA-S. The AIoT NF/AMF reports the inventory result to the AIoT AF.

#### 6.10.2.2 RAN reader case

The following figure shows the call flow for AIoT device authentication via RAN reader. The main points are the same with alternative 2 of UE reader case as depicted in 6.10.2.1.2.



Figure 6.10.2.2-1: Authentication for AIoT device via RAN reader

1. The AIoT AF sends Inventory command to the AIoT NF/AMF via NEF.

2. The AIoT NF/AMF selects the RAN reader for the inventory.

3. The AIoT NF/AMF interacts with UDR to authorize the inventory, e.g., whether the AF is allowed to perform the inventory.

Editor’s Note: Clarification on inventory authorization is ffs.

4. In case the authorization succeeds, the AIoT NF/AMF may interact with AAA-S before paging AIoT devices, e.g., using mechanism described in RFC 4739 [7]. The AAA-S may return authentication information to the AIoT NF/AMF.

5. In case the authorization succeeds, the AIoT NF/AMF responds to the AIoT AF via NEF.

6. In case the authorization succeeds, the AIoT NF/AMF sends AIoT paging command to the RAN reader, which may include an Auth Container that contains authentication information.

7. The RAN reader broadcasts AIoT paging message, which may include the Auth Container.

8. The AIoT device determines to responds to the AIoT paging message, it sends an AIoT message to the AIoT NF/AMF via the RAN reader with Device ID and optional another Auth Container, which contains information for authentication.

9. The AIoT NF/AMF acts as proxy of the AIoT device to interact with AAA-S for authentication between the AIoT device and the AAA-S, e.g., using EAP framework or mechanism described in RFC 4739[7]. The AIoT NF/AMF may further interact with the AIoT device for authentication if required by AAA-S.

10. In case the authentication succeeds, the AIoT NF/AMF will receive authentication success indication from AAA-S. The AIoT NF/AMF reports the inventory result to the AIoT AF.

As an example, when AKA based authentication method is used, then step 4 is not performed, no Auth Container in steps 6-8, and in step 9, the RAND and AUTN will be encapsulated into an Auth Container sent from the AIoT NF/AMF to the AIoT device, the RES will be encapsulated into another Auth Container sent from the AIoT device to the AIoT NF/AMF.

### 6.10.3 Evaluation

TBA

## 6.11 Solution #11: Authentication and ID Privacy of AIoT devices with USIM on AIoT AS Layer

### 6.11.1 Introduction

This solution is addressing the Key Issue #5: Authentication and Key issue #3: Privacy by protecting AIoT device identifiers and Key issue #4: Protection of information during AIoT service communication.

Some solutions in 3GPP TR 23.700-13 [4] suggests the following simplified protocol stack for Ambient IoT as shown in Figure 6.11.1-1:



Figure 6.11.1-1: Potential AIoT protocol stack

This new architecture provides a control plane delivery of commands and instructions towards the AIoT device from the corresponding AF in charge. Since NAS protocol is not supported and the AIoT devices have a low complexity, it is assumed that only EAP-AKA’ is supported and the non-3GPP access procedures can be reused, comparable to N5CW or AUN3 devices. With that concept it is possible to either to protect the messages between AIoT device and AIoT function, similar to untrusted access, or between AIoT device and AIoT Reader similar to trusted access.

It is assumed that the AIoT device can use SUCI based on the presence of the USIM.

The security relationship is established between AIoT Device and the AIoT Reader, it is assumed that the communication between AIoT Reader and the AIoT Function is protected with NDS/IP.

### 6.11.2 Solution details

In this solution for adopting the trusted access principles, the AIoT Reader is taking the role as the TNAP and the AIoT Function the role as TWIF/TNGF and AMF.



Figure 6.11.2.2-1: AIoT Device authentication and ID Privacy as trusted non-3GPP access

1. The AF is authenticated based on the mechanisms described in TS 33.501, e.g. TLS, a token based authorization mechanism or local configuration at the NEF. The AF subscribes for device information of authenticated AIoT devices.

2. The AIoT device connects to an AIoT Reader as an access network. The AIoT device may be triggered by the AIoT Reader to send this message, e.g. based on the presence of a specific broadcast message from AIoT Reader.

3. The AIoT reader sends an L2 message with an EAP-Identity Request to the AIoT Device. This step may be part of step 5.

4. The AIoT device provide its unique AIoT Identity, e.g. SUCI or 5G-GUTI in an EAP-Identity Response to the AIoT Reader. The message may contain other information e.g. Device EPC.

5. The AIoT Reader selects a AIoT Function, e.g. based on the received realm, and sends an AAA request to the selected AIoT Function. The AIot Reader may include the received information from the AIoT Device, e.g. Device EPC, and the device location in the request.

6. The AIoT Function shall select an AUSF and sends Nausf\_UEAuthentication\_Authenticate Request message to the AUSF. The Nausf\_UEAuthentication\_Authenticate Request message contains SUCI or SUPI (in case of a valid 5G-GUTI is received by the AIoT Function acting as an AMF). The request message contains also an indication that the request is from an AIoT device.

7. The AUSF shall send Nudm\_UEAuthentication\_Get Request to the UDM including SUCI or SUPI and the AIoT indication. Upon reception of the Nudm\_UEAuthentication\_Get Request, the UDM shall invoke SIDF if a SUCI is received. SIDF shall de-conceal SUCI to gain SUPI before UDM can process the request. The UDM may select an authentication method based on the "realm" part of the SUPI, the AIoT device indicator, a combination of the "realm" part and the AIoT device indicator, or the UDM local policy. The UDM/ARPF shall first generate an authentication vector. The UDM shall subsequently send this transformed authentication vector AV' (RAND, AUTN, XRES, CK', IK') to the AUSF. The UDM shall also send the MSK indicator to the AUSF to indicate that the AIoT device does not support the 5G key hierarchy.

8. The AUSF shall send the EAP-Request/AKA'-Challenge message to the AIoT Function in a Nausf\_UEAuthentication\_Authenticate Response message.

9. The AIoT Function shall transparently forward the EAP-Request/AKA'-Challenge message to the AIoT Reader in an AAA response message.

10. The AIoT function forwards the EAP-Request/AKA’-Challenge message to the AIoT Device in a L2 message.

11. The AIoT device computes the authentication response message.

12. The AIoT device shall send the EAP-Response/AKA'-Challenge message to the AIoT Reader in a Auth-Resp message on L2.

13. The AIoT Reader shall send the EAP-Response/AKA'-Challenge message to the AIoT Function in a AAA request message.

14. The AIoT Function shall transparently forward the EAP-Response/AKA'-Challenge message to the AUSF in Nausf\_UEAuthentication\_Authenticate Request message.

15. The AUSF shall verify the message by comparing the XRES and RES. If successful, based on the MSK indicator received in step 11, the AUSF shall generate the MSK, the AUSF shall not generate the KAUSF.

16. The AUSF shall send to the AIoT Function an Nausf\_UEAuthentication\_Authenticate Response message including the EAP-Success, the MSK, and the SUPI and if available the GPSI.

17. AIoT Function sends an EAP-Success to the AIoT Reader upon reception of the MSK, and the SUPI and if available the GPSI.

18. The AIoT Reader stores the MSK and forwards the EAP-Success to the AIoT Device in a L2 message.

Editor’s Note: it is FFS whether the nr of interactions with the device are feasible for AIoT

### 6.11.3 Evaluation

Editor’s Note: The solution needs to be aligned with the final SA2 conclusions on the architecture.

## 6.12 Solution #12: Authentication and ID Privacy of AIoT devices with USIM on AIoT Layer

### 6.12.1 Introduction

This solution is addressing the Key Issue #5: Authentication and Key issue #3: Privacy by protecting AIoT device identifiers and Key issue #4: Protection of information during AIoT service communication.

Some solutions in 3GPP TR 23.700-13 [4] suggests the following simplified protocol stack for Ambient IoT as shown in Figure 6.12.1-1:



Figure 6.12.1-1: Potential AIoT protocol stack

This new architecture provides a control plane delivery of commands and instructions towards the AIoT device from the corresponding AF in charge. Since NAS protocol is not supported and the AIoT devices have a low complexity, it is assumed that only EAP-AKA’ is supported and the non-3GPP access procedures can be reused, comparable to N5CW or AUN3 devices. With that concept it is possible to either to protect the messages between AIoT device and AIoT function, similar to untrusted access, or between AIoT device and AIoT Reader similar to trusted access.

It is assumed that the AIoT device can use SUCI based on the presence of the USIM.

The security relationship is established between AIoT Device and the AIoT Function and adopts the untrusted access concepts.

### 6.12.2 Solution details

In this solution for adopting untrusted access principles, the AIoT Reader is taking the role as the Access Point and the AIoT Function the role as N3IWF and AMF.



Figure 6.12.2.1-1: AIoT Device authentication and ID Privacy as untrusted non-3GPP access

1. The AF is authenticated based on the mechanisms described in TS 33.501, e.g. TLS, a token based authorization mechanism or local configuration at the NEF. The AF subscribes for device information of authenticated AIoT devices.

2. The AIoT device connects to an AIoT Reader as an access network. The AIoT device may be triggered by the AIoT Reader to send this message, e.g. based on the presence of a specific broadcast message from AIoT Reader. When the AIoT device decides to attach to the AIoT Reader, the AIoT device selects an AIoT Function in a 5G PLMN. The AIoT Device may retrieve the address of the AIoT Function in a broadcast message of the AIoT Reader to which the AIoT Reader is connected to, or, the AIoT Function address may be preconfigured in the AIoT Device. The AioT Device retrieves limited IP connectivity from the AIoT Reader to perform the procedure.

3. The AIoT device proceeds with the establishment of an IPsec Security Association (SA) with the selected AIoT Function by initiating an IKE initial exchange.

4. The AIoT device shall initiate an IKE\_AUTH exchange by sending an IKE\_AUTH request message. The AUTH payload is not included in the IKE\_AUTH request message, which indicates that the IKE\_AUTH exchange shall use EAP signalling (in this case EAP-5G signalling).

5. The AIoT Function responds with an IKE\_AUTH response message which includes the AIoT Function identity, the AUTH payload to protect the previous message it sent to the AIoT device (in the IKE\_SA\_INIT exchange) and an EAP-Request/5G-Start packet.

6. The AIoT device shall validate the AIoT Function certificate and shall confirm that the N3IWF identity matches the AIoT Function selected by the AIoT device. The AIoT device shall send an IKE\_AUTH request which includes an EAP-Response/5G-NAS packet that contains a unique AIoT identifier, e.g. such as SUCI or 5G-GUTI and may contain the Electronic Product Code (EPC) of the AIoT device.

7. The AIoT Function shall select an AUSF and sends Nausf\_UEAuthentication\_Authenticate Request message to the AUSF. The Nausf\_UEAuthentication\_Authenticate Request message contains SUCI or SUPI (in case of a valid 5G-GUTI is received by the AIoT Function acting as an AMF). The request message contains also an indication that the request is from an AIoT device.

8. The AUSF shall send Nudm\_UEAuthentication\_Get Request to the UDM including SUCI or SUPI and the AIoT indication. Upon reception of the Nudm\_UEAuthentication\_Get Request, the UDM shall invoke SIDF if a SUCI is received. SIDF shall de-conceal SUCI to gain SUPI before UDM can process the request. The UDM may select an authentication method based on the "realm" part of the SUPI, the AIoT device indicator, a combination of the "realm" part and the AIoT device indicator, or the UDM local policy. The UDM/ARPF shall first generate an authentication vector. The UDM shall subsequently send this transformed authentication vector AV' (RAND, AUTN, XRES, CK', IK') to the AUSF. The UDM shall also send the MSK indicator to the AUSF to indicate that the AIoT device does not support the 5G key hierarchy.

9. The AUSF shall send the EAP-Request/AKA'-Challenge message to the AIoT Function in a Nausf\_UEAuthentication\_Authenticate Response message.

10. The AIoT Function shall transparently forward the EAP-Request/AKA'-Challenge message to the AIoT device in an IKE\_AUTH response message.

11. The AIoT device computes the authentication response message.

12. The AIoT device shall send the EAP-Response/AKA'-Challenge message to the AIoT Function in a Auth-Resp message.

13. The AIoT Function shall transparently forward the EAP-Response/AKA'-Challenge message to the AUSF in Nausf\_UEAuthentication\_Authenticate Request message.

14. The AUSF shall verify the message by comparing the XRES and RES. If successful, based on the MSK indicator received in step 11, the AUSF shall generates the MSK, the AUSF shall not generate the KAUSF.

15. The AUSF shall send to the AIoT Function an Nausf\_UEAuthentication\_Authenticate Response message including the EAP-Success, the MSK, and the SUPI and if available the GPSI.

16. AIoT Function sends an EAP-Success/EAP-5G to the AIoT device upon reception of the MSK, and the SUPI and if available the GPSI.

Editor’s Note: it is FFS whether the number of device interactions are feasible for AIoT

Editor’s Note: how USIM is supported is FFS

### 6.12.3 Evaluation

Editor’s Note: The solution needs to be aligned with the final SA2 conclusions on the architecture.

## 6.13 Solution #13: Authentication and ID privacy of AIoT devices without USIM

### 6.13.1 Introduction

This solution is addressing the the Key Issue #5: Authentication and Key issue #3: Privacy by protecting AIoT device identifiers and Key issue #4: Protection of information during AIoT service communication.

Some solutions in 3GPP TR 23.700-13 [4] suggests the following simplified protocol stack for Ambient IoT as shown in Figure 6.13.1-1:



Figure 6.13.1-1: Potential AIoT protocol stack

This new architecture provides a control plane delivery of commands and instructions towards the AIoT device from the corresponding AF in charge. Since NAS protocol is not supported and the AIoT devices have a low complexity, it is assumed that the devices in addition do not have a USIM for authentication and the security procedures. It is further assumed that the devices do have a simple security configuration which is shared with the AF. For ID privacy, the devices use a default ID for the onboarding to the AIoT network, after that only a temporary ID is used, derived from the security configuration. Confidentiality keys are derived from the security configuration, integrity protection may not be required based on the level of importance of the downlink messages.

### 6.13.2 Solution details



Figure 6.13.2.2-1: AIoT Device authentication and ID Privacy

1. The AF has a preshared configuration of the AIoT devices, which includes a unique Default ID of the device and respective security parameters for deriving a security key and temporary IDs for ID privacy.

2. The AF sends an AIoT Request to the NEF with the Default Id and the security parameters of the AIoT device.

3. The Nef forwards the AIoT Request to the selected AioT Function.

4. The AIoT Function generates a Nonce and uses it to derive an Encryption Key and a Temporary ID from the received security context from the NEF. The AIoT Function uses the Encryption Key to calculate an Expected Result.

5. The AIoT Function sends an AIoT Request to the AioT Reader, including the Default ID and the Nonce.

6. The AIoT Reader sends the AIoT Request to the AIoT Device, which is listening to requests with the Default ID for initial onboarding to the AIoT network.

7. The AIoT Device calculates the Encryption Key and the Temporary ID for the next usage in a similar way as the AIoT Function. The AIoT Device calculates the Result as a proof that it holds the security context.

8. The AIoT Device sends a AIoT response to the AIoT Reader, including the computed Result.

9. The AIoT Reader forwards the AIoT Response to the AiOT Function.

10. The AIoT Function compares the received result with the expected result and authenticates the AIoT Device if both are identical.

11. The AIoT Function sends a AIoT Response to the NEF, indicating the success of the authentication.

12. The Nef forwards the AioT Response to the AF.

All further requests from the AF are then encrypted by the AIoT Function and the AIoT Device, the AIoT device is addressed by the Temporary ID only. The AIoT Function may change the Encryption Key and the Temporary ID by providing a new Nonce in a protected downlink request.

Editor’s Note: Whether Nonce in Step 6 can be sent to the device depends on RAN paging message

Editor’s Note: How to address synchronization issues with the Temporary ID is FFS

Editor’s Note: How to generate the encryption key is FFS

Editor’s Note: It is FFS how to use the solution with a group of devices;

Editor’s Note: Command protection is FFS

Editor’s Note: how the result of the authentication is computed is FFS

### 6.13.3 Evaluation

Editor’s Note: The solution needs to be aligned with the final SA2 conclusions on the architecture.

## 6.14 Solution #14: Information protection during AIoT service communication

### 6.14.1 Introduction

The solution addresses the security requirement of KI#4: Protection of information during AIoT service communication.

The basic principle of this solution is to reuse principles of Integrity protection in 5GS [5] but simplify the tasks in a AIoT device. Furthermore, in this solution description also reuses the concept of locally generated Temporary ID (TempID) for privacy protection. However, this solution does not strictly depend on that solution, other means of handling temporary IDs are possible.

The solution assumes the following AIoT device capabilities:

- The AIoT device has higher complexity than a RFID tag that only reflects the same preconfigured device ID when excited by RF power, but significantly lower complexity than a 3GPP CIoT device.

- The AIoT device has a factory-encoded key and device ID, and the network also has these stored or can retrieve them.

- The AIoT device has a non-volatile storage capability.

As the available power in an AIoT device is very limited, the message exchange between the device and the network must be minimized and the computational requirement should be minimized. The solution is based on the following principle:

- The AIoT device can locally derive new temporary identifiers (TempID) to be used in future communication, a list of unused TempIDs.

NOTE: The solution requires the AIoT device to have a few TempIDs available that has not been used i.e., never sent in clear text over the radio interface.

Editor’s Note: The temporary ID derivation algorithm is FFS and as the temporary ID is derived locally the overall Temporary ID handling needs to address the case of TempID out-of-synchronization and re-synchronization between the device and 5GC.

- The AIoT device uses these identifiers together with the factory-encoded key to protect the information transmitted.

- The derivation of an MAC is used. The intention is to make the derivation with low complexity.

- The Inventory service and Command service would require different type of protection.

### 6.14.2 Solution details

#### 6.14.2.1 Inventory Service information protection

The information transmitted from the AIoT device is an ID triggered by an Inventory request targeting e.g. all AIoT devices, group, or type of AIoT devices. Inventory is not typically targeted for one individual device, but that case should still be considered.

Eavesdropping: To protect against eavesdropping the AIoT device shall transmit an ID that cannot be linked to specific AIoT device.

Manipulation/unauthorized transmission: To Protect against this, the system shall never send the same ID over the radio interface and have a method to check that the ID is authentic if needed.

The solution to protection against the listed attacks is the same. The UE shall always send an ID that has never been sent over the radio interface. For the listed cases the following procedure principle applies:

**Group/all device Inventory request:**

* The Reader transmit an Inventory request targeting a group of AIoT devices or all AIoT device.
* The AIoT device responds to the request using a TempID that has never been used before (first in the list)
* In case the Reader UE wants to verify that the ID received is authentic the Reader can perform an Individual device Inventory Request.

**Individual AIoT device Inventory request:**

* The Reader transmit an Inventory request targeting a specific of AIoT device by including the device TempID that has never been used before (first in the list).
* The AIoT device responds to the request using a TempID that has never been used before (second in the list, as the first TempID was used in the Inventory request).
* In case the Reader UE wants to verify that the ID received is authentic the Reader repeats this request.

#### 6.14.2.2 Command Service information protection

The information transmitted from the Reader is an ID and command. The information transmitted by the UE is an ID and a response. The Command is triggered by a Reader and targeting e.g. a group of AIoT devices or an individual AIoT device.

Eavesdropping: To protect against eavesdropping an ID that cannot be linked to specific device and optionally encrypt the command.

Manipulation: To protect against this, a MAC can be derived/verified using at least the following input parameters the command/response and a shared secrete.

unauthorized transmission: To protect against this, the system shall never send the same ID over the radio interface and have a method to check that the ID is authentic if needed. Additionally, deriving a MAC as discussed above can be added.

**MAC derivation**

The solution proposes to use available parameters and use these in a simplified "Integrity Algorithm" e.g. scramble the input with the key.



Figure 6.14.2.2-1: Derivation of MAC or XMAC.

As the TempID has never been sent over the radio interface and it is derived from the factory encoded key, it would be possible to further simplify the MAC derivation by reusing the TempID as a "derived Key" from the factory encoded key.



Figure 6.14.2.2-2: Alternative derivation of MAC or XMAC.

**Downlink (Reader -> AIoT device)**

The DL command message is sent to the AIoT device using the TempID (first in the list)

In addition to the Command, the Reader includes also a MAC derived as follows:

The KEY input is the factory-encoded key in the device.

The TempID input is the next TempID in the list (second ID).

The DIRECTION bit shall be set to 1 for downlink.

MESSAGE input is the Command.

Editor’s Note: Whether and how to derive a MAC for DL Group-Command message is FFS.

**Uplink (AIoT device -> Reader)**

The DL command message was sent to the AIoT device using a TempID (first in the list)

The AIoT device responds to the Reader by using the next TempID (second in the list) and includes the response (e.g., an ACK or Data) plus a MAC derived as follows:

The KEY input is the factory-encoded key in the device.

The TempID input is the next TempID in the list (third TempID).

The DIRECTION bit shall be set to 0 for uplink.

MESSAGE input is the UL response (e.g., ACK or data).

### 6.14.3 Evaluation

TBD.

## 6.15 Solution #15: End-to-end security protection of command procedure

### 6.15.1 Introduction

This solution addresses key issues #1 and #4. This solution provides a security mechanism for protecting the commands transmitted between an AIoT device and an Application Function (AF) for AIoT services. This solution assumes the AF manages the AIoT device identifier and the corresponding security protection profile. The security protection profile includes a device credential and an algorithm to use to protect the command.

NOTE 1: Device credential types are determined based on each AIoT service and device capability.

The proposed mechanism is also applied to protect the messages exchanged for enable/disable device operation. The security mechanism for protecting the commands is applied when the AIoT device and AF are provisioned with the security protection profile.

### 6.15.2 Solution details



Figure 6.15.2.1-1: End-to-end protection of messages during Command procedure

NOTE 2: The reference architecture and Command procedure described in clause 6.3 of TR 23.700-13 (i.e., solution #3) [4] are used to describe the end-to-end protection mechanism in this solution. The proposed mechanism can be applied to any reference architectures and procedures for AIoT services that require transmissions of messages between AIoT devices and an Application Function.

0. Each AIoT device is provisioned with its AIoT device identifier and security protection profile such as a device credential and an algorithm to use to protect the command. An Application Function (AF) manages the AIoT device identifier and the associated security protection profile.

1. When the AF triggers a Command procedure towards individual AIoT device(s), the AF protects a Command based on the Command Protection Key (CPK) and a freshness parameter. The AF, then, sends a message containing the protected Command to the AIoT Controller.

When the AF triggers a Command procedure towards a group of AIoT devices, it protects a Command based on the Group Command Protection Key (GCPK) and a freshness parameter. Then, the AF sends the protected Command to the AIoT Controller.

NOTE 3: CPK is either derived from the device credential or provisioned on the AIoT device by the AF.

NOTE 4: GCPK, if used, is provisioned on the AIoT devices by the AF.

NOTE 5: Freshness parameter can be a counter, time-based counter or a random number depending on the AIoT service and device capability.

Editor’s Note: whether device ID is included is FFS.

2. The AIoT Controller provides the Command to the selected Reader(s).

3. The Reader sends the Command to the AIoT device(s).

4. Upon receiving the Command, the AIoT device decrypts/verifies the received Command. If the verification is successful, the AIoT device processes the Command.

5. If the AIoT device needs to send a response, it generates a Command Response and protects it based on the CPK and a freshness parameter. The AIoT device, then, sends the protected Command Response.

In case of Group command, the AIoT device protects the Command response based on the GCPK and a freshness parameter.

6. Upon receiving the Command Response, the Reader sends it to the AIoT Controller with optional Enrichment data such as the location of the Reader if configured by AIoT Controller.

7. The AIoT Controller provides the protected Command Response to the AF.

8. The AF decrypts/verifies the received Command Response based on the CPK and the freshness parameter.

Editor’s Note: The procedure needs to align with SA2.

### 6.15.3 Evaluation

TBD

## 6.16 Solution #16: Disabling operation procedure for Ambient IoT services

### 6.16.1 Introduction

This solution is proposed to address Key Issue #1, which provides a method for permanently disabling the Ambient IoT device(s). This solution applies to Topology 1 and Topology 2.

Considering different roles are involved in the Ambient IoT services, it is assumed that only the owner of Ambient IoT device can authenticate the Ambient IoT device and initiate the disabling operation procedure to disable the Ambient IoT device’s capability to transmit RF signals.

In this solution, it is assumed that the Ambient IoT device is stateful. After the authentication is performed successfully, the Ambient IoT device enters the next state (e.g. secure state), in which it can execute the disabling operation.

For disabling the Ambient IoT device, the owner sends the Disabling request to the Ambient IoT device via the gNB/Intermediate node. If the owner is the application provider, the Disabling request is E2E protected between the AF and Ambient IoT device. If the owner is the operator, the Disabling request is protected by the AIoTF/AMF.

Once receiving the Disabling command, the Ambient IoT device determines whether to execute this operation based on authentication status and verification result. By maintaining the authentication status, the Ambient IoT device can ensure that the Disabling command is sent from the legitimate requester instead of the attacker. By maintaining the security material, the Ambient IoT device can ensure that the Disabling command is not tampered by the attacker.

NOTE 1: The states of Ambient IoT device are defined in TR 23.700-13 [4].

### 6.16.2 Solution details

This solution assumes that the authentication between the Ambient IoT device and its owner has been performed before disabling operation and the authentication status is stored in the Ambient IoT device. It is also assumed that the Ambient IoT device already share the security materials for communication protection with the AF or AIoTF/AMF before disabling operation.

Editor’s Note: The security materials between the Ambient IoT device and AF or AIoTF/AMF are FFS.

NOTE 2: The authentication and security context establishment between the Ambient IoT device and its owner is out of scope of this solution.



Figure 6.16.2-1: Disabling operation procedure for Ambient IoT services

1. The device owner sends the Disabling request to AIoTF/AMF. The Disabling request may include Ambient IoT device ID(s), disabling command, etc. If the owner of Ambient IoT device is application provider, the disabling command is E2E protected between the AF and Ambient IoT device.

2. The AIoTF/AMF determines the gNB/Intermediate node to transmit this Disabling request.

3. If the owner of Ambient IoT device is operator, the AIoTF/AMF protects the disabling command using the shared security materials. The AIoTF/AMF sends the Disabling request to the selected gNB/Intermediate node.

4. Upon reception of the request message, the selected gNB/Intermediate node executes inventory procedure with the Ambient IoT device.

5. The selected gNB/Intermediate node sends the protected Disabling Command message to the Ambient IoT device.

6. Once receiving the Disabling command, the Ambient IoT device firstly determines whether the authentication has been performed based on the authentication status. If the requester has been authenticated, the Ambient IoT device further verifies the received Disabling request by using the shared security materials. If the verification is successful, the Ambient IoT device can execute the disabling operation.

Note: Once the disabling operation is finished, the owner removes the security context of Ambient IoT device.

Editor’s Note: Whether the Disabling acknowledgement message is needed is FFS.

Editor’s Note: How to maintain the state in the Ambient IoT device is FFS.

Editor’s Note: Alignment with conclusion from TR 23.700-13 [4] is FFS.

### 6.16.3 Evaluation

TBD

## 6.17 Solution #17: Disabling operation procedure for AIoT services

### 6.17.1 Introduction

The solution addresses the security requirement of KI#1: Protection for disabling device operation. According to TS 22.369 [2], the network operator shall provide a suitable mechanism to temporarily/permanently disable the capability of an Ambient IoT device or a group of Ambient IoT devices to transmit RF signals. The solution assumes the network operator is responsible for managing the AIoT device using the temporarily/permanently disable operation.

Specifically, the AIoT NF sends the AIoT Disable Request to the AIoT device. The message includes the disable security parameter, which is calculated by the shared key between AIoT devices and AIoT NF. The AIoT device calculates and verifies the disable security parameter using the same shared key to decide whether to perform the disable operation.

### 6.17.2 Solution details

Depicted in Figure 6.17.2-1 is the disable operation procedure for AIoT Services.

**

**Figure 6.17.2-1: Disable operation procedure for AIoT service**

1. The AF sends the AIoT Disable Request to the AIoT NF via the NEF, including the AIoT device filter information, and disable type (i.e., temporarily/permanently disable).
2. The AIoT NF sends the AIoT Disable Trigger to the RAN/UE Reader, including the AIoT device filter information.

The AIoT device filter information is used to page an AIoT device or a group of AIoT device, which could be the AIoT device ID, the group ID, or the partial ID of AIoT device.

1. The RAN/UE Reader sends the AIoT Paging Request to the AIoT device, including the AIoT device filter information.
2. The AIoT device checks whether the ID matches according to the AIoT device filter information.
3. The AIoT device sends the AIoT Paging Response to the RAN/UE Reader, including the AIoT device ID.
4. The RAN/UE Reader sends the AIoT Disable Trigger Response to the AIoT NF, including the AIoT device ID.
5. The AIoT NF uses f function to calculate the disable security parameter with the input parameters key K, AIoT ID, disable type and counter, i.e., f (K, AIoT ID, disable type, counter)

NOTE: The f functions can be HMACfunctions, f1-f5 functions, AES functions, etc.

NOTE: The key K can be the root key or its derived key of the AIoT device.

1. The AIoT NF sends the AIoT Disable Request to the RAN/UE Reader. The message contains the AIoT ID, disable security parameter, disable type and counter.
2. The RAN/UE Reader forwards the AIoT Disable Request to the AIoT device.
3. The AIoT device uses the same shared key to verify the disable security parameters.
4. (a) The AIoT device performs the disable operation according to the disable type, if the calculated disable security parameter matches the received disable security parameter.

(b) The AIoT device sends the AIoT Disable Response to the RAN/UE Reader before disabling the RF capability, if the AF requires a feedback.

1. The RAN/UE Reader forwards the AIoT Disable Response to the AIoT NF.
2. The AIoT NF sends the AIoT Disable Response to the AF via the NEF.

### Editor's Note: How the AIoT device maintans the counter is FFS.6.17.3 Evaluation

TBD.

## 6.18 Solution #Y: Authorization procedure for AF-based intermediate node selection

### 6.18.1 Introduction

This solution is proposed to address Key Issue #2, supporting the authorization for AF-based intermediate node selection. This solution applies to Topology 2.

The authorization of intermediate node is based on the UE subscription data stored in the UDM. Once receiving the potential intermediate node information provided by the AF, the AMF/AIoTF interacts with its UDM to obtain the UE subscription data. The AMF/AIoTF determines whether the UE is authorized to provide the requested Ambient IoT service by using the Ambient IoT service information, Ambient IoT service area information, and/or Ambient IoT device information included in the UE subscription data.

### 6.18.2 Solution details



Figure 6.18.2-1: Authorization procedure for AF-based intermediate node selection

1. The AF sends the Inventory/Command request to AMF/AIoTF via NEF. The Inventory/Command request may include UE ID(s), Ambient IoT device ID, area information, command, etc.

2. For each UE as the potential intermediate node provided by the AF, the AMF/AIoTF interacts with its UDM to obtain the UE subscription data for UE authorization.

3. For each UE, the AMF/AIoTF checks the UE authorization as follows:

- If the Ambient IoT service information is included in UE subscription data, the AMF/AIoTF determines whether the UE is authorized to provide the requested Ambient IoT service.

- If the Ambient IoT service area information is included in UE subscription data, the AMF/AIoTF determines whether the UE is authorized to provide the requested Ambient IoT service.

- If the Ambient IoT device information is included in UE subscription data, the AMF/AIoTF determines whether the UE is authorized to provide the requested Ambient IoT service.

If multiple UEs are authorized, the AMF/AIoTF further decides to select one or more UE(s) as intermediate node(s).

4. For the selected UE, the AMF/AIoTF sends the inventory/command request to the UE.

Editor’s Note: The procedure of inventory/command is to be aligned with SA2.

Editor’s Note: The details of subscription information is to be aligned with SA2.

Editor’s Note: Whether the AF knows the UE ID is FFS.

### 6.18.3 Evaluation

Editor’s Note: Further evaluation is FFS.

## 6.19 Solution #19: Authorization of AIoT capable UE in topology 2

### 6.19.1 Introduction

This solution addresses key issue #2: Authorization for 5G Ambient IoT services.

In the Topology 2, as defined in TR 38.848, the AIoT capable UE acting as the intermediate node is responsible for transferring the Ambient IoT data and/or signalling between AIoT devices and 5GS and under network control.

A black background with a black square

Description automatically generated with medium confidence

Figure 6.19.1-1: Topology 2

In this solution, it is proposed that the AIoT device and the AIoT capable UE acting as the intermediate UE are authorized by the network and provisioned by the network with a common security material to protect the interface between the AIoT devices and the AIoT capable UE acting as the intermediate UE.

In this solution, the interface between the AIoT capable UE acting as the intermediate node and AIoT devices is called Ntopo2. Security material to protect Ntopo2 procedure (e.g. Inventory, Command) between the AIoT devices and the AIoT capable UE acting as the intermediate UE is called Ntopo2 security material, e.g. for integrity protection, confidentiality protection.

When the protected messages received from the peer side over Ntopo2 is successfully verified by the receiving side (the AIoT devices or the intermediate UE), it indicates that the sending side (the intermediate UE or the AIoT devices) has been authorized by the network.

### 6.19.2 Solution details

The procedure for authorization for AIoT capable UE acting as the intermediate node for 5G Ambient IoT is described as follows.



Figure 6.19.2-1: Authorization of Intermediate UE for topology 2

0. The AIoT device is provisioned with the Ntopo2 security material and its associated information from the network. During the provisioning procedure, the network checks whether the AIoT device is authorized to use Ntopo2 for Ambient IoT service. The Ntopo2 security material and its associated information is described in step 3.

Editor’s Note: It’s FFS how the Ntopo2 security material is provisioned to the AIoT device, and by which 5GC NF.

NOTE1: One or more AIoT devices (e.g. AIoT devices in the same group) can be associated with a common Ntopo2 security material.

1. The AIoT capable UE performs Registration procedure as defined in TS 23.502. The UE includes the AIoT Intermediate node capability indication in Registration Request message. The AMF determines whether the UE is authorized to work as Intermediate UE for AIoT based on the UE’s Subscription data. If the UE is authorised to work as Intermediate UE, then the AMF also includes the authorization information in NGAP message sent to NG-RAN.

2. AF sends AIoT service request (e.g. Inventory Request, Command Request) to the AIoT capable UE. The AIoT service request may be sent to the AIoT capable UE via 5GC or from AF directly over user plane with input parameters containing the area information, device information, optional inventory strategy information, and optional report aggregation info etc.

NOTE2: The detail of AIoT service request procedure is to be defined by SA2.

3. The AIoT capable UE sends a Key Request message to the network to get the Ntopo2 security material if the Ntopo2 security material is not available locally. In addition, the AIoT capable UE includes its Ntopo2 security capability to be used over the interface with AIoT devices, e.g. supported ciphering algorithms. The AIoT capable UE may also include the information that is associated with Ntopo2 security material e.g. the spatial information (e.g. geographical area), the temporal information (e.g. time period, expiry), and/or the AIoT device information (e.g. device ID, device group ID, device type, device mask info etc).

4. The 5G NF checks whether the AIoT capable UE is authorized to work as Intermediate UE for Ambient IoT service and provisions Ntopo2 security material. The 5G NF may check the Ambient IoT service authorization information for the AIoT capable UE with the AF.

Editor’s Note: It’s FFS which 5G NF performs the authorization and provisions the security material.

5. The 5G NF returns the Ntopo2 security material in Key Response Message. In addition, the 5G NF may include the chosen Ntopo2 ciphering algorithm, the Ntopo2 security policies, and the information that is associated with Ntopo2 security material as described in step 3.

6. The AIoT capable UE interacts with NG-RAN for radio resource allocation if the UE is authorized as in step 1.

NOTE3: The detail for radio resource allocation is assumed to be defined by RAN.

7. If the AIoT service request is Inventory Request, the AIoT capable UE initiates Inventory request over Ntopo2, selects Ntopo2 security material as received in step 5 and based on the associated security material information and input parameters in AIoT service request, and protects the Inventory request message with Ntopo2 security material.

8. If verification of the protected Inventory request message is successful based on the Ntopo2 security material received in step 0, the AIoT device reports the device ID and/or other info to the AIoT capable UE in the response message. The response message is protected by Ntopo2 security material. Successful verification of the protected Inventory request message assures the AIoT device that the AIoT capable UE is authorized to provide the AIoT service.

9. If the AIoT service request is Command Request, the AIoT capable UE initiates Command request over Ntopo2 and protects the Command request message with Ntopo2 security material simliar as step 7.

10. If verification of the protected Command request message is successful based on the Ntopo2 security material received in step 0, the AIoT device performs the Command and sends the response message. The response message is protected by Ntopo2 security material. Successful verification of the protected Command request message assures the AIoT device that the AIoT capable UE is authorized to provide the AIoT service.

11. The AIoT capable UE verifies the AIoT service response messange (Inventory response or Command response). Successful verification of the AIoT service response message assures the AIoT capable UE that the AIoT device is authorized for the AIoT service. The AIoT capable UE sends the AIoT service response to the AF. The AIoT service response may be sent to the AF via 5GC or directly over user plane.

Editor’s Note: What is the security material and the protocol betwen AIoT device and reader is FFS.

Editor’s Note: Clarify if ntopo2 key is long-term key or not. If long-term, the security impact is FFS.

### 6.19.3 Evaluation

Editor’s Note: Each solution should motivate how the potential security requirements of the key issues being addressed are fulfilled.

## 6.20 Solution #20: Lightweight AIOT ID privacy based on hashes

### 6.20.1 Introduction

The assumption of this solution is AIoT device can not support 5G-AKA due to power or computational resource limitation.

The existing Key issue #3, Privacy by protecting AIoT device identifiers, specifies the following requirement:

*Mechanisms for mitigating privacy threats (described above) by identifying, linking, and tracking the identifiers of AIoT Device(s) shall be supported*.

In addition, RAN2#125bis chairman notes [4] detail the following baseline procedure:

*…*

*3. RAN2 will support two use cases, “inventory” and “command”. The definition, detailed wording is FFS*

*4. Baseline:*

*Step A: Based on the service request, the reader sends the Initial Trigger Message indicating device(s) that need to respond; Details FFS*

*Step B: Triggered device(s) performs the random access-like procedure, if needed; Details FFS*

*Step C: The device may perform the data communication with the reader as needed,: Details FFS*

*…*

This solution aims to address both, the KI#3 of the present document and the baseline procedure described in the RAN2#125bis agreement.

Moreover, the proposed procedure for obfuscating the AIoT Device AIOT\_ID describes the “inventory” use case but is equally applicable to the “command” use case.

### 6.20.2 Details

The simplified call flow associated with the proposed solution is presented below.



Figure 6.20.2-1: Call flow for Lightweight AIOT ID privacy based on hashes

Steps associated with the call flow in Figure 6.20.2-1:

0. Configuration and provisioning step

0a. AIOT Device is configured/provisioned with AIOT\_ID

0b. RAN Reader/Intermediate Node is configured/provisioned with a list of AIOT\_IDs

1. AF sends an Inventory Operation Request with the following information: target area for the operation, client which requests the operation, and match information used to filter and discover the target AIoT devices for the operation.

2. The NEF authorizes the AF request. If the AF request is authorized, The NEF discovers the AIoT function using the information in the Inventory Operation Request, e.g. using the target area for the operation to discover the AIoT function from NRF. If the target area for the operation matches the AIoT service area of the AIoT Function, the NRF returns the information for the AIoT function to the NEF.

3. The NEF forwards the Inventory Operation information to each of the selected AIoT functions.

4. The AIoT function discovers and selects a reader or an Intermediate Node to perform Inventory Operation according to the Inventory Operation information. e.g. using the target area for the operation to discover the readers. If the target area for the operation matches the AIoT service area supported by the readers, those readers can be selected to execute the inventory operation.

5. The AIoT Function sends an Inventory Request with the Inventory Operation information for each selected reader.

6. The selected reader executes the inventory operation towards the target AIoT Devices.

Note that step 6 may be eavesdropped, manufactured, and replayed by an attacker. An appropriate authorisation of the Reader and freshness control to prevent message 6 replay will be needed to remedy such attacks.

7. The AIoT Device performs AIOT\_ID selection based on matching of hashed AIOT\_ID

a. The AIoT Device computes a hash using RAND\_READ as salt for each AIOT\_ID

b. The AIoT Device compares the hash values of its provisioned AIOT\_ID and AIOT\_ID hashes from the broadcast message to find at least one matching hashed AIOT\_ID.

c. The AIoT Device selects an AIOT\_ID that has a matching hash.

Note that step 7 may create an overhead for the AIOT that may lead to a resources depletion attack on AIoT Device. This overhead is either lighter or comparable with other methods for confidentiality protection of the AIoT Device identity.

Editor’s Note: The detailed explanation of the AIOT Device selection in this step is FFS.

8. The AIoT Device sends a Registration Request containing hashed obfuscated AIOT-ID

9. The RAN Reader or Intermediate Node performs AIOT\_ID selection based on matching of hashed AIOT\_ID

a. compute a hash using RAND\_READ for each AIOT\_ID

b. check that the hashed AIOT\_ID received from the AIOT Device matches one of the stored AIOT-ID from step 0b

10. The RAN Reader or Intermediate Node sends the Inventory Report with AIOT\_ID to AMF/AIOT AF

11-12.. AIoT Function reports the operation result to the AF vie NEF/AF.

### 6.20.3 Evaluation

This solution addresses the requirement of Key issue #3.

This solution proposes a lightweight privacy protection method for the AIoT Device identity. The proposed method provides lightweight confidentiality protection of AIoT Device identifiers.

Editor’s Note: The impact of the proposed solution on power and processing-constrained AIOT devices and network is FFS.

Editor’s Note: Further evaluation is FFS.

## 6.21 Solution #21: Ephemeral AIOT ID security context based on puzzles for privacy

### 6.21.1 Introduction

The assumption of this solution is AIoT device can not support 5G-AKA due to power or computation resource limitation.

The existing Key issue #3, Privacy by protecting AIoT device identifiers, specifies the following requirement:

*Mechanisms for mitigating privacy threats (described above) by identifying, linking, and tracking the identifiers of AIoT Device(s) shall be supported*.

This solution aims to address KI#3 of the present document.

The proposed procedure for obtaining the ephemeral security context is equally applicable to both, “inventory” and “command” use cases.

The agreed AIOT Random Access framework is presented below.



**Figure 6.21.1-1: AIOT Random Access framework**

The steps in Figue6.X.1-1 are described below.

1. The reader sends a paging message and a set of occasion synchronization messages which respectively provides the device IDs of the devices to respond and configures/delimits the random access occasions for transmissions by the AIOT devices

2. An AIOT device selects an occasion (using at least slotted ALOHA as the baseline), and transmits a random device ID in MSG1

3. The reader, upon successful reception of MSG1, transmits MSG2 by including the received random device ID in MSG2.

4. If the device receives the echoed random device ID in MSG2, it transmits MSG3 which contains upper layer data (e.g., an application layer device ID)

5. MSG4 may be transmitted by the reader (e.g., for subsequent command transmission), but the understanding is that contention is already resolved at MSG2 transmission.

### 6.21.2 Details

The simplified call flow based on the AIOT Random Access framework is presented below. This procedure is based on the security properties of cryptographic puzzles.



**Figure 6.21.2-1: Modified AIoT Random Access Procedure for establishing ephemeral security between AIOT Device and Reader**

The steps in Figure 6.21.2-1 are described below.

1. The Reader determines to prepare a set of cryptographic puzzles and the Reader will prepare a set of N tuples. Each tuple is comprised of a K-MACi (key) and corresponding K-MACi-IND (key index corresponding to key).

2. The Reader composes a set of N cryptographic puzzles each hiding a tuple comprising of the Ephemeral Key K-MACi, corresponding Ephemeral Key Index K-MACi-IND , and either partial key or partial hash function argument.

3. The reader transmits a paging message and a set of occasion synchronization messages. The combination of the paging message and synchronization message identifies which AIoT devices should respond to the paging message. In other words, the combination of the paging message and synchronization messages identifies which AIoT devices should perform a random access procedure. The paging message includes one or more of the puzzles that were constructed in step 2.

Editor’s Note: The feasibility of including multiple puzzles in paging messages is FFS.

4. The AIoT Device uses the information in the paging message and synchronization messages to determine that the AIoT Device needs to respond to the paging message. In other words, the AIoT Device determines that it needs to perform a RACH procedure. If the AIoT Device determines that it needs to perform a RACH procedure, then, a the AIOT Device randomly selects one puzzle from the set of N puzzles that were received in the paging message. If the AIoT Device does not determine that it needs to perform a RACH procedure, or the AIoT device determines that the strength of the puzzle, as determined by the puzzle number, does not satisfy certain requirements, then the procedure will stop in this step and the AIoT will not perform a RACH procedure. Determining that the strength of the puzzle does not satisfy the AIoT security requirements for the application means that the selected puzzle strength may compromise information used by the application after the security context is established with a puzzle of a certain strength.

5. The AIoT Device solves the puzzle that was selected in step 4 and recovers the security parameters.

Editor’s Note: The feasibility or using puzzle-based methods in power and processing-constrained AIOT devices is FFS.

6. An AIOT device selects an occasion (using at least slotted ALOHA as the baseline), and transmits a random device ID in a message, MSG1. The message also includes the K-MACi-IND that was recovered in step 5.

7. The Reader performs a lookup for the K-MACi from the corresponding K-MACi-IND received in step 6.

8. The AIOT Device and the Reader enter a state where they have established an ephemeral security context using K-MACi. The subsequent messages of this procedure may now be confidentiality and integrity protected using the ephemeral security context based on K-MACi. In other words, the AIOT Device and the Reader have established an ephemeral security context. The ephemeral security context is based on K-MACi. After this step, the AIoT Device may use the ephemeral security context to encrypt data that it sends to the reader and the reader may use the ephemeral security context to encrypt data that it sends to the AIoT Device. Thus, information can be sent more securely between the AIoT Device and the Reader. In addition, the ephemeral security context obtained in this step can be used to bootstrap more persistent security associations between AIOT functional entities.

9. Upon successful reception of MSG1, the reader transmits MSG2 by including the received random device ID in MSG2. The Reader may use K-MACi to encrypt some or all of the information in MSG2.

10. The AIoT Device uses K-MACi to decrypt some or all of the information in MSG2. A random device ID is an example of information that is carried in MSG2. If the AIoT Device determines that the random device ID that it transmitted in step 6 is included in MSG2, then the AIoT Device transmits MSG3 which contains upper-layer data (e.g., an application layer device ID). The AIoT Device may use K-MACi to encrypt some or all of the information in MSG3.

11. The Reader uses K-MACi to decrypt some or all of the information in MSG3. The reader may then Transmit MSG4 (e.g., for subsequent command transmission), but the understanding is that contention is already resolved at MSG2 transmission. The reader may use K-MACi to encrypt some or all of the information in MSG4.

### 6.21.3 Evaluation

TBD

## 6.22 Solution #22: Solution for protecting AIoT ID by using temporary ID

### 6.22.1 Introduction

This solution addresses key issue #3: Privacy by protecting AIoT device identifiers.

It is assumed that an AIoT device has power or computational resource limitation.

To avoid fake ID reporting, AIoTF checks the authenticity of the message by verifying the MAC which is generated using KAIoT and RANDAIoTF.

It is assumed that AIoT device and AF are provisioned with AIoT device identifier and a key (K).

Editor’s Note: Alignment with conclusion from TR 23.700-13 [4] is FFS.

### 6.22.2 Solution details



Figure 6.32.2-1 AIoT ID protection call flow

0. AIoT ID and a key (K) are provisioned to the AIoT device and AF. AF generates KAIoT from K and RANDAF.

1. AF sends AIoT service request message to AIoTF (AIoT Function). This message may be sent via NEF. AIoT service request message includes Temp AIoT ID #1, RANDAF, and KAIoT.

NOTE: If this message is sent for the first time, the Temp AIoT ID #1 is the AIoT ID.

Editor’s Note: Sending the AIoT device ID in plaintext is the first message is FFS.

2. AIoTF transfers AIoT service request message. This message includes RANDAIoTF, RANDAF, and Temp AIoT ID #1.

3. AIoT device derives KAIoT from K and RANDAF. After that, AIoT device generates MAC using Temp AIoT ID #1, RANDAIoTF, and KAIoT.

4. AIoT device responds with AIoT service response. The message includes RANDAIoT, Temp AIoT ID #1, and MAC.

5. After AIoTF finds KAIoT from Temp AIoT ID #1 received in step 4, AIoTF checks the authenticity of the message by verifying the MAC.

6. If the verification in step 5 is successful, AIoTF sends AIoT service response to AF. RANDAIoT and Temp AIoT ID #1 are included in this message.

7. AIoT device and AF generate Temp AIoT ID #2 from KAIoT, RANDAIoT, and Temp AIoT ID #1. The Temp AIoT ID #2 is used next time the AF requests a service to AIoT.

Editor’s Note: How to resolove syncronization issue on the temporary ID between AIoT device and network is FFS.

Editor’s Note: Key recovery attack is FFS.

### 6.22.3 Evaluation

This solution addresses the requirement of Key Issue #3 by using temporary AIoT ID.

This solution assumes that AIoT device and AF are provisioned with AIoT device ID and a key.

Editor’s Note: Further evaluation is FFS.

## 6.23 Solution #23: AIoT device ID privacy protection using anonymity key

### 6.23.1 Introduction

This solution addresses the security requirement of KI#3: Privacy by protecting AIoT device identifiers.

To prevent the AIoT deivce ID from being exposed in the air interface, the 5GC can trigger the reader(s) to pag an AIoT device or a group of AIoT Devices by broadcasting a partial AIoT Device ID. The AIoT device ID is consist of the common part (e.g., the Home Network identifier, or the 3rd party identifier) and the unique part (e.g., an identifier used to identify a specific Ambient IoT device). The partial AIoT Device ID is the common part of a group of AIoT device ID.

The AIoT deivce matches the partial AIoT deivce ID will perform random access responding to the pagging message and report its permanent ID to the network. The AIoT device encrypts its permanent ID with an anonymity key AK, which is a shared key between AIoT device and AIoTF. The AK and corresponding Key ID is provided to the Network by the AF. The AIoTF may receive the AK and Key ID from AF’s service request or retrieve such information from UDM (provided in advance by the AF).

### 6.23.2 Solution details

This solution describes the AIoT device ID protection as shown in the following figure 6.Y.2-1.



Figure 6.23.2-1: AIoT device ID privacy protection using anonymity key

1. The AF sends the AIoT Service Request to AIoTF, including the AIoT device ID, may include the anonymity key AK and the key ID. The anonymity key AK and the key ID can be provided by the AF to the Network, and pre-stored in UDM. The AK include two kinds of keys: (1) a specific key for each AIoT device; or (2)a group key shared by a group of AIoT devices.
2. The AIoTF requests the Reader to start Paging the AIoT device or a group of AIoT devices, using the partial AIoT Device ID.

The AIoT device ID is consist of the common part (e.g., the Home Network identifier, or the 3rd party identifier) and the unique part (e.g., an identifier used to identify a specific Ambient IoT device). The partial AIoT Device ID is the common part of a group of AIoT device ID.

1. The Reader pages the AIoT device or a group of AIoT devices using the partial AIoT Device ID.

For one or mautiple AIoT device(s):

1. The AIoT deivce matchs the partial AIoT deivce ID will perform random access responding to the message and report its permanent ID to the network. To prevent the AIoT device permanent ID being exposed, the AIoT device encrypts the AIoT device permanent ID with the anonymity key AK. The input parameters to the ciphering should include at least a fresh value, this fresh value changes to prevent the AIoT device from being linked or traced.
2. The AIoT device sends the Paging Response to the Reader, including the encrypted AIoT device permanent ID, the Key ID, and the fresh value.
3. The Reader sends the encrypted AIoT device permanent ID, the Key ID, and the fresh value to the AIoTF.
4. The AIoTF may retrieve the AK using the received Key ID from the UDM if needed. The AIoTF uses the anonymity key AK and the fresh value to decrypt the AIoT device permanent ID. Then the AIoTF matches the decrypted AIoT devices ID(s) with the received AIoT device ID in step 1.

Editor's Note: Where are Anonymity Keys stored in Network side is FFS.

Editor's Note: The privacy protection of using the individual AIoT device Key with exposed Key ID in step 5 is FFS.

### Editor's Note: Possible known ciphertext attack and remediation is FFS.6.23.3 Evaluation

This solution fully addresses security requirement in KI#3.

This solution requires the AIoT device to be provisioned with the anonymity key to encrypt the permanent device ID. And this solution is suitable for both Inventory-only case and Inventory and Command case.

## 6.24 Solution #24: temporary ID based AIoT device privacy protection

### 6.24.1 Introduction

This solution addresses the security requirement of KI#3: Privacy by protecting AIoT device identifiers. The purpose of the AIoT device temporary ID is to provide an unambiguous identification of the AIoT device that does not reveal the AIoT device permanent identity. The AIoT device temporary ID is allocated by the AIoTF after AIoT device initial registration.

### 6.24.2 Solution details

This solution describes the allocation and usage of the AIoT device temporary identifier, as shown in the following figure 6.24.2-1.



Figure 6.24.2-1: signalling flow of the AIoT device temporary ID allocation and usage

1. The AIoT device temporary ID is allocated by AIoTF after AIoT device initial registration. It is assumed that there are available security contexts between AIoT device and the Network after the AIoT device initial registration. Hence, the AIoT device temporary ID is protected by security contexts of the AIoT device, and transferred to the AIoT device.
2. The AF sends the AIoT Service Request to the AIoTF via the NEF, including the AIoT device ID. The NEF maps the external AIoT device ID to internal AIoT device ID.

Editor’s Note: whether there is AIoT device external ID is based on SA2’s decision, which is FFS.

1. The AIoTF maps the received AIoT device ID to the AIoT device temporary ID, and use the AIoT device temporary ID to indicate the specific AIoT device. The AIoTF sends the AIoT Service Request (Inventory or Command) via the reader.

After the AIoT device temporary ID allocation, the AIoT device can be paged by its temporary ID, or partial of the temporary ID to enable more efficient radio signalling procedures.

1. The AIoT device sends the AIoT Service Response to the AIoTF via the reader, including the AIoT device temporary ID.
2. The AIoTF maps the AIoT device temporary ID to AIoT device ID to response the AF via the NEF. The NEF maps the internal AIoT device ID to external AIoT device ID.

Editor's Note: How to synchronize and reallocate the temporary ID is FFS.

### 6.24.3 Evaluation

This solution fully addresses security requirement in KI#3.

This solution requires the AIoT device to store the temporary ID allocated by the Network

Editor's Note: More Evaluation is FFS.

## 6.25 Solution #25: Use temporary identifier to protect the privacy of AIoT device identifiers.

### 6.25.1 Introduction

This solution addresses key issue #3: Privacy by protecting AIoT device identifiers.

### 6.25.2 Solution details

****

**Figure 1. AIoT Temporary Identifier configuration procedure.**

AIoT is an Ambient IoT Device which has ultra-low complexity power, cost and resource-constrained. The reader refers to the base station or UE based on the connectivity topologies. The core network NF includes independent or co-located network functions such as AMF/AUSF/UDM/Authentication Function of AIoT.

1. Initial temporary identifier (TempID) and crenditials such as TempID generation key are known by both the CN NF and the AIoT device through onboarding or registration procedure of AIoT device or other NAS/AS procedure.
2. The network sends AIoT request such as inventory request or command request to AIoT device.
3. Once triggered by AIoT request, the AIoT sends information including TempID to the reader.
4. The reader transfer the AIoT information to the core network.
5. The network use tempID to identify devices and perform operations. For example, the core network verifies the validity of the AIoT device based on the TempID and other subscription data of the device.
6. The network returns an acknowledgement to the reader. And a freshness parameter is included in the response.
7. The reader transfers the acknowledgement to the AIoT device.
8. Both the AIoT device and the core network function generate a new TempID according to the TempID derivation function for later use. The freshness parameter will be used in both sides.

Editor’s Note: How to deal with the TempID synchronization issue is FFS.

Editor’s Note: The corenetwork NF needs to be specified.

TempID derivation function

When deriving the TempID from TempID generation key, the following parameters shall be used to form the input S to the KDF:

- FC = 0xxx;

- P0 = "TempID";

- L0 = length of "TempID"; (i.e. 0x00 0x06)

- P1 = device ID;

- L1 = length of device ID.

- P2 = freshness parameter;

- L2 = length of freshness parameter.

The input key KEY (i.e. TempID generation key) is long term key pre-configured both in AIoT device and core network function.

Device ID is the fixed identifier of AIoT device.

### 6.25.3 Evaluation

Edtor’s Note: evaluation is FFS.

## 6.26 Solution #26: Local generated Temporary ID to provide device privacy

### 6.26.1 Introduction

The solution addresses the security requirement of KI#3: Privacy by protecting AIoT device identifiers. Specifically, this solution proposes a method for the Ambient IoT system to generate and use Temporary IDs.

The basic principle of this solution is that a Temporary ID (TempID) is locally generated both by the CN NF and the AIoT device after every time the TempID has been sent over the radio interface as a response to an Inventory request or a Command request.

The solution assumes the following AIoT device capabilities:

- The AIoT device has higher complexity than a RFID tag that only reflects the same preconfigured device ID when excited by RF power, but significantly lower complexity than a 3GPP CIoT device.

- The AIoT device has a factory-encoded key and device ID.

- The AIoT device has a non-volatile storage capability.

- The TempID generation algorithm is light weight and enough complex to avoid unauthorized AIoT device tracking.

As the available power in an AIoT device is very limited, the message exchange between the device and the network must be minimized. The solution is based on the following principle:

- The initial temporary identifier (TempID) is known by both the CN NF and the AIoT device. After the AIoT device has been onboarded to the network the CN NF provision the AIoT device with the initial TempID and/or parameters to derive the initial TempID i.e., parameters for the TempID generation algorithm.

NOTE: It is assumed that during the onboarding procedure the CN NF can retrieve information from another NF or Application Function (AF) to onboard the AIoT device. The initial message from the UE during onboarding could e.g. include the device owner ID, URL, FQDN or other info that enables the CN NF to establish an IP connection with the AF that holds additional onboarding information needed.

- Every time the TempID has been sent over the radio interface as a response to an Inventory request or a Command request, both CN NF and AIoT device locally generate a new TempID. The exact algorithm used can be decided during the normative phase, but the assumption is that it at least uses the factory-encoded key and a seed.

- If the CN NF detects that the TempID is out of sync, i.e., the CN NF expected a different TempID than received during an Inventory or Command. The CN NF has two options to re-synchronize 1) send a new seed to restart the algorithm or 2) find the received TempID in the sequence of TempIDs and continue from there.

### 6.26.2 Solution details

#### 6.26.2.1 Temporary ID generation.

In the procedure below CN NF is used as a generic name for the 5GC Network Function that supports the Ambient IoT functionality. It is expected that SA2 will define and specify CN NF as part of the 5GC architecture.



Figure 6.26.2-1 Local temporary ID control

0. It is assumed that the AIoT device is pre-provisioned by the device owner with information to be used when onboarding to a network. The information includes a device unique identifier (unique at owner level, not necessary globally unique), device owner ID and security Key.

It is assumed that the device owner and the MNO has a Service Level Agreement (SLA) and the network is provisioned with information needed to onboard an AIoT device(s) or a URL or FQDN to establishing IP connection to an AF that holds the device unique onboarding information.

1. The AIoT device is triggered to onboard and sends an onboarding request that includes (device owner ID, device ID). The network either holds the necessary information to continue the onboarding or connects to the device owners AF and retrieves the necessary information from the AF to continue the onboarding.

NOTE: How the AIoT device is triggered to request onboarding is up to SA2 WG to specify.

The CN NF sends a response message (Accept/Reject, selected TempID algorithm, seed) to the AIoT device.

Editor’s Note: It is FFS whether the device ID must be concealed.

1. The CN NF creates a AIoT device context and generates locally the TempID to be used when triggering the AIoT device next time e.g. when sending a Command to the AIoT device. To generate the TempID the CN NF uses the device security key received by the device owner AF, selected algorithm, and seed.
2. The AIoT device generates locally the TempID to be used next time. To generate the TempID the AIoT device uses its pre-provisioned security key and the selected algorithm and seed received in the response message.
3. The CN NF sends a command message or Inventory request, that may be triggered by 3rd-part AF. The Command message includes the AIoT device TempID, PDU.
4. The AIoT device only considers Command message that includes its expected TempID and check the command PDU. In case of Inventory request the AIoT device check whether the Inventory is for the device.
5. The AIoT device responds to the Command or Inventory request.
6. Both the AIoT device and CN NF generates the next temporary ID.

### 6.26.3 Evaluation

TBD.

## 6.27 Solution #27: Privacy protection of AIoT device identifier based on a temporary identifier

### 6.27.1 Introduction

This solution addresses key issue #3.

This solution provides a privacy protection mechanism by introducing a temporary identifier associated with an AIoT device. In the proposed mechanism, an AIoT device generates a temporary device identifier (T-ID) when it needs to transmit its device identifier towards the network (e.g., for the purpose of inventory management). This solution assumes that an Application Function (AF) manages the AIoT device identifier and the corresponding security protection profile. The security protection profile includes device credential and an algorithm to use to generate a T-ID.

NOTE 1: The privacy protection mechanism based on a temporary identifier is applied when the AIoT device and AF are provisioned with the security protection profile.

NOTE 2: Device credential types are determined based on each AIoT service and device capability.

### 6.27.2 Solution details

#### 6.27.2.1 Procedures



Figure 6.27.2.1-1: Privacy protection based on a temporary device identifier (T-ID) during Inventory procedure

NOTE 3: The reference architecture and Inventory procedure described in clause 6.3 of TR 23.700-13 (i.e., solution #3) [4] are used to describe the privacy protection mechanism in this solution. The proposed mechanism can be applied to any reference architectures and procedures for AIoT services that require transmission of AIoT device identifier by AIoT device.

0. Each AIoT device is provisioned with its AIoT device identifier and security protection profile such as a device credential and an algorithm to use to generate a T-ID. An Application Function (AF) manages the AIoT device identifier and the associated security protection profile.

1-3. The AF triggers an Inventory procedure towards AIoT devices.

4. Upon receiving the Inventory Request from the Reader, the AIoT device replies to the Reader with an Inventory Response containing a temporary device identifier (T-ID). The AIoT device generates a new T-ID as described in clause 6.27.2.2.

5. Upon receiving a T-ID, the Reader sends the received T-ID to the AIoT Controller. Additionally, the Reader sends Enrichment data (e.g., the Reader location) along with the T-ID if configured by the AIoT Controller.

6. The AIoT Controller stores the received T-ID and Enrichment data.

7. There are two methods to provide Inventory Response to AF as follows:

- Pull-based procedure: When the AIoT Controller receives a Data Request from the AF (step 7a in Figure 6.27.2.1-1), it sends an Inventory Response containing the stored T-IDs along with the Enrichment data if exist (step 7b in Figure 6.27.2.1-1). If a list of T-ID(s) was included in the Data Request, the AIoT Controller only contains the information associated with the requested T-IDs in the Inventory Response.

- Push-based procedure: the AIoT Controller sends an Inventory Response (step 7b in Figure 6.Y.2.1-1) containing the information received from previous steps to the corresponding AF if T-ID includes the service identifier (e.g., AF Identity).NOTE 4: In case of pull-based procedure in step 7, a freshness parameter needs to be synchronized between the AIoT device and AF.

Editor’s Note: How an AF identifies the key to generate expected T-ID is FFS.

#### 6.27.2.2 Generation of a temporary identifier

A new temporary device identifier (T-ID) is generated as follows:

T-ID = F(K, freshness parameter, AIoT device identifier), where F is a service specific function that generates a temporary ID. The K is the key provisioned at the AIoT device and AF for temporary ID generation. The K is either a device credential or derived from the device credential. The freshness parameter is determined based on the device capability. For example, an index can be used as a freshness parameter along with a refresh timer. In this case, the AIoT device increments the index if a refresh timer has expired.

### 6.27.3 Evaluation

This solution addresses the security requirements in key issue #3 by introducing a temporary identity based on symmetric key provisioned at an AIoT device and AF.

## 6.28 Solution #29: Privacy protection on AIoT device IDs

### 6.28.1 Introduction

This solution addresses key issue#3: “Mechanisms for mitigating privacy threats (described above) by identifying, linking, and tracking the identifiers of AIoT Device(s) shall be supported.”

### 6.28.2 Details

Preassumption: the AIoT device has one ID preconfigured by the manufacture (Application Function).

A screenshot of a computer

Description automatically generated

Step 0a. the device is configured with one Device ID in the manufacturing time. Every ID is configured corresponding

to an Index.

Editor’s Notes: Clarifications of the configuration of device ID and the Index and the mapping is FFS.

Editor’s Notes: How to address the synchronizaiton issue is FFS.

Editor’s Notes: if the ID is sending in the plaintext in paging message, the attacker can track the HASH chain, how to address this issue is FFS.

Editor’s Notes: It is FFS how the procedure is integrated into inventory procedure, e.g. how the Reader report the device ID to the AF, when the inventory procedure is triggered.

Editor’s Notes: It is FFs what is the impact on the device and network.

Step 0b. the Application Function sends the Device IDs and Indexes to the AIoTF through NEF.

Step 0c. AIoTF sends the corresponding Device IDs to each Reader based on distribution policy from Application Function or local policy from MNOs.

Step 0d. Reader stores the Devices IDs and the corresponding Indexes under this Reader.

Step 1. Reader sends the paging message to the AIoT devices. Reader sends ID1-1, ID2-1, … in the paging message.

NOTE 1: According to RAN2 agreement in #116, the paging message may contain one ID (one Device ID or one group ID) or more (multiple IDs FFS in RAN2) IDs. The procedure may be updated based on RAN2 progress.

NOTE 2: IDm-n indicates the n-th ID for Device m.

Step 2: AIoT device replies with ID1-1||Index to the Reader.

NOTE 3: Index shall indicate which ID of which device.

Step 3: Reader compares the ID1-1 with its data base, then confirm this ID1-1 is in its data base.

Step 4: Reader echoes back the ID1-1 to device, following RAN2 procedure.

Step 5: Device checks ID1-1 is correct, use ID1-2 in next message, in which ID1-2 = HASH (ID1-1)

NOTE 4: Reader shall use unused ID for each device for the next paging if there is any.

### 6.28.3 Evaluation

TBD

## 6.29 Solution #29: Providing a network-computed AIoT concealed device identifier (AICI) to an AIoT device

### 6.29.1 Introduction

This solution addresses KI#3: Privacy by protecting AIoT device identifiers.

### 6.29.2 Solution details

The solution proposes a method for the 5G network to compute a AIoT Concealed Device Identifier (AICI) and provide the SUCI to the AIoT device in a command message. Once an AIoT device is identified, e.g., after a successful completion of inventory procedure, the network can send a command message on the downlink channel to the AIoT device. In the downlink command message, the network includes an AICI, which is computed based on the long-term identifier of the AIoT device using the public key of the network. The downlink command message is both confidentiality and integrity protected using keys derived from a shared key between the network and the AIoT device. Figure 6.29.2-1 presents a high-level message flow of the solution.



Figure 6.29.2-1: Procedure for delivering a AICI to an AIoT Device

The figure is self-explanatory, therefore, the steps are not explained step-by-step. Exact content of the messages exchanged, and details about authentication challenge, computing response to the challenge, and deriving the keys Kenc and Kint are not described because these details have to be adjusted with the authentication protocol that is finally agreed.

NOTE: If the AIoT device does not have a network-computed AICI, for example, in the very first time of the device’s life cycle, then the AIoT device computes AICI using null scheme. This happens only in the beginning. To avoid using null scheme in the first time, a network can choose to provision every AIoT device with a network-computed AICI before they are handed out to their users.

Editor’s Note: The content of the paging message and how the device decides to respond to the paging message based on the content of the paging message is FFS.

Editor’s Note: Synchronization of AICI is FFS

Editor’s Note: Whether AICI is to be computed and decrypted in SIDF/UDM is FFS

Editor’s Note: Whether AIoT device can update and store AICI is FFS

### 6.29.3 Evaluation

TBD

## 6.30 Solution #30: Privacy protection for inventory operation

### 6.30.1 Introduction

This solution addresses Key issue #3: Privacy by protecting AIoT device identifiers.

The principle of this solution is that the key used to protect inventory operations is pre-configured in AIoT devices. These keys are shared among multiple AIoT devices.

This solution provides two procedures:

- Inventory procedure with unprotected inventory request parameters, and

- Inventory procedure with protected inventory request parameters, which can address the privacy issues related to groups of devices.

### 6.30.2 Solution details

#### 6.30.2.1 Inventory procedure with unprotected inventory request parameters

Editor’s Note: What is the Device ID Matching Info. needs clarification.

Editor’s Note: The feasibility of UDM participating in security computing is FFS.

Editor’s Note: The need for transmitting security policy in message is FFS.

Editor’s Note: How to decrypt data if there is no Key ID in the message is FFS.

The inventory procedure with unprotected inventory request parameters is shown in the following figure.

 Figure 6.30.2-1: Inventory procedure with unprotected inventory request parameters

0. During the device initialization phase, the Device ID protection keys are pre-configured in the AIoT device. The Device ID protection keys are used to protect Device ID during inventory process. A key ID for these keys may also be pre-configured in order to facilitate key management. A security policy for AIoT Device ID protection may also be pre-configured.

1. The AF sends an inventory operation request to the Ambient IoT Function (AIoTF). The AF may also provide Device ID Matching Information to address only specified AIoT Devices.

2. The AIoTF sends Inventory parameter request to the UDM.

3. The UDM generates a Network Nonce and sends it to the AIoTF.

4. The AIoTF sends Inventory request to the AIoT Devices through the AIoT Reader. The Inventory request includes the Device ID Matching Information and Network Nonce.

5. The AIoT Device that matches the Device ID Matching Information performs the following operations:

- Generate a Device Nonce;

- Use the pre-configured Device ID protection keys, Network Nonce and Device Nonce to derivate keys for Device ID confidentiality and/or integrity protection;

- Use new derived keys to protect the AIoT Device ID according to the AIoT Device ID protection policy (security policy);

- The AIoT Device sends Inventory response to the AIoTF through the AIoT Reader. The response includes Security policy, Device Nonce, protected AIoT Device ID and the key ID (if any).

6. The AIoTF sends Authentication request to the UDM. The Authentication request includes Security policy, Device Nonce, Protected AIoT Device ID and the key ID (if any).

7. The UDM uses the same method as the AIoT Device to derive the keys, and then decrypts and/or verifies the protected AIoT Device ID.

The UDM checks whether the AIoT Device ID is valid. If the verification is successful, the UDM return the Device ID to the AIoTF.

8. The AIoTF, AIoT Devices and UDM continue to perform steps 4-7 on the remaining AIoT Devices.

9. The AIoTF returns the Inventory result to the AF.

#### 6.30.2.2 Inventory procedure with protected inventory request parameters

The inventory procedure with protected inventory request parameters is shown in the following figure.

 Figure 6.30.2-2: Inventory procedure with protected inventory request parameters

0. Same as step 0 in clause 6.30.2.1.

1. Same as step 1 in clause 6.Y.2.1.

2. The AIoTF sends Inventory parameter request to the UDM. The request includes the Device ID Matching Information.

3. The UDM performs the following operations:

- Generate a Network Nonce;

- Use the Device ID protection keys and Network Nonce to derivate keys for confidentiality and/or integrity protection;

- Use new derived keys to protect the AIoTF Device ID Matching Information according to the AIoT Device ID protection policy (security policy);

The protected AIoT device ID matching information may also be a temporary ID generated using security key and security parameters.

- Send Inventory parameter response to the AIoTF. The response includes protected AIoT Device ID Matching Information, Security policy, Network Nonce, and the key ID (if any).

4. The AIoTF sends Inventory request to AIoT Devices through AIoT Reader. The Inventory request includes Protected AIoT Device ID Matching Information, Security policy, Network Nonce, and the key ID (if any).

5. The AIoT Device performs the following operations:

- If there is a key ID in the request, check if it matches one of the locally stored keys.

- Use the same method as the UDM to derive the keys, and then decrypts and/or verifies the protected AIoT Device ID Matching Information;

If temporary ID is used, the locally stored key and parameters received from the request are used to calculate the local temporary ID.

- Check if its Device ID/Group ID matches the Device ID Matching Information;

If temporary ID is used, check if the local temporary ID is equal to the temporary ID in the request.

The other operations are the same as step 5 in clause 6.Y.2.1.

6-9. The remaining steps and operations are the same as steps 6-9 in clause 6.Y.2.1.

### 6.30.3 Evaluation

This solution addresses the Key Issue #3.

This solution uses pre-configured keys to protect AIoT Device IDs during inventory operations. The pre-configured key can be shared among all AIoT Devices or a group of AIoT devices. The pre-configured keys can also be used on the network side to protect the privacy of Device ID matching information.

Editor’s Note: Further evaluation is FFS.

## 6.31 Solution #31: Ambient IoT ID privacy

### 6.31.1 Introduction

This solution addresses KI#3 and proposes a solution to protect the privacy of the identifier of an Ambient IoT (AIoT) device. The solution focuses on the functionality in the AIoT device:

- verifying a request to share the identity of the AIoT device,

- protecting the identity of the AIoT device, when sharing it.

The procedure fits the baseline RAN procedure agreed in RAN2#125bis:

*Step A: Based on the service request, the reader sends the Initial Trigger Message indicating device(s) that need to respond;*

*Step B: Triggered device(s) performs the random access-like procedure, if needed;*

*Step C: The device may perform the data communication with the reader as needed.*

### 6.31.2 Solution details

A diagram of a company

Description automatically generated

Figure 6.31.2-1: Ambient IoT ID privacy

In step 0, AIoT device(s) are configured with identity(ies) and device specific parameters, such as security credentials including device- and group-specific credentials.

In step 1, the AIoT Management Function (AIoT MF) sends a message to the reader to trigger an inventory procedure including an inventory request message and a configuration associated with the inventory procedure. The configuration includes timers (e.g., to broadcast the request and to collect responses from AIoT device(s)), response aggregation requirements and forwarding criteria, etc.

Note 1: The AIoT Management Function (AIoT MF) depends on the entity that owns or manages the resource (i.e., AIoT device). If a device is owned or managed by the network, the disabling is triggered by a CN function, otherwise, it is triggered by a 3rd party Management Function managing the device.

In step 2, the reader UE/gNB sends the inventory request message, as received in step 1, to AIoT device(s). The inventory request includes a device (or group) identifier and a device-specific (or group-specific) challenge. Based on the configuration received, the reader UE/gNB may start a timer T1 to collect responses.

In step 3, The AIoT device processes the received request and checks whether the message:

3.1 is addressed to the AIoT device by matching its identifier (i.e., device or group identifier) against the received identifier, and

3.2 is sent by a trusted party by verifying the challenge value using its security credentials and identifiers.

If the checks 3.1 and 3.2 succeed, the AIoT computes a pseudonym based on its identifier and the received challenge value. This can be implemented by means of a lightweight function, e.g., a hash function.In step 4, the AIoT device sends its pseudonym in a response message (e.g., an inventory response message) including a device-specific key identifier.

Editor’s Note: Whether the lightweight function is a keyed hash function and how the device-specific key identifier is used are FFS.

In step5, based on the configuration received in step 1, the reader UE/gNB aggregates the response message(s) received, until the time T1 runs out and/or other forwarding criteria (e.g., all device responses are received) are met.

In step 6, the reader UE/gNB forwards the response(s) to the AIoT MF.

### 6.31.3 Evaluation

This solution describes a lightweight solution for privacy protection of AIoT device(s) identity(ies).

Editor’s Note: Further evaluation is FFS.

## 6.Y Solution #Y: <Solution Name>

### 6.Y.1 Introduction

Editor’s Note: Each solution should list the key issues being addressed.

### 6.Y.2 Solution details

### 6.Y.3 Evaluation

Editor’s Note: Each solution should motivate how the potential security requirements of the key issues being addressed are fulfilled.

# 7 Conclusions

Editor’s Note: This clause contains the agreed conclusions that will form the basis for any normative work.

Annex <X> (informative):  
Change history

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Change history** | | | | | | | |
| **Date** | **Meeting** | **TDoc** | **CR** | **Rev** | **Cat** | **Subject/Comment** | **New version** |
| 04/2024 | SA3#115Adhoc-e | S3-241476 |  |  |  | Initial draft TR | 0.0.0 |
| 04/2024 | SA3#115Adhoc-e | S3-241648 |  |  |  | Incorporated accepted contributions S3-241477, S3-241622, S3-241630, S3-241636 | 0.1.0 |
| 05/2024 | SA3#116 | S3-242536 |  |  |  | Incorporated accepted contributions S3-242649, S3-242534, S3-242535, S3-242539, S3-242540, S3-242541 | 0.2.0 |
| 08/2024 | SA3#117 | S3-243699 |  |  |  | Incorporated accepted contributions S3-243492, S3-243680 | 0.3.0 |
| 10/2024 | SA3#118 | S3-243828 |  |  |  | Incorporated accepted contributions S#-244124, S3-244366, S3-234460, S3-244367, S3-244368. S3-244369, S3-244370, S3-244449, S3-244450, S3-244451, S3-244452, S3-244453, S3-244454, S3-244455, S3-244456, S3-244457, S3-244458, S3-244459, S3-244508, S3-244509, S3-244467, S3-244468, S3-244469, S3-244470, S3-244477, S3-244478, S3-244479, S3-244480, S3-244483, S3-244484, S3-244485, S3-244486, S3-244487, S3-244488, S3-244489 | 0.4.0 |