



## eSIM IoT Technical Specification

Version 1.3

22<sup>nd</sup> May 2026

---

### Security Classification: Non-Confidential

Access to and distribution of this document is restricted to the persons permitted by the security classification. This document is subject to copyright protection. This document is to be used only for the purposes for which it has been supplied and information contained in it must not be disclosed or in any other way made available, in whole or in part, to persons other than those permitted under the security classification without the prior written approval of the Association.

### Copyright Notice

Copyright © 2026 GSM Association

### Disclaimer

The GSMA makes no representation, warranty or undertaking (express or implied) with respect to and does not accept any responsibility for, and hereby disclaims liability for the accuracy or completeness or timeliness of the information contained in this document. The information contained in this document may be subject to change without prior notice.

### Compliance Notice

The information contain herein is in full compliance with the GSMA Antitrust Compliance Policy.

This Permanent Reference Document is classified by GSMA as an Industry Specification, as such it has been developed and is maintained by GSMA in accordance with the provisions set out GSMA AA.35 - Procedures for Industry Specifications.

## Table of Contents

<b>1</b>	<b>Introduction</b>	<b>8</b>
1.1	Overview	8
1.2	Scope	8
1.3	Document Purpose	8
1.4	Intended Audience	8
1.5	Definition of Terms	8
1.6	Abbreviations	10
1.7	References	11
1.8	Conventions	13
1.9	References to SGP.22	13
<b>2</b>	<b>General Architecture</b>	<b>13</b>
2.1	General Architecture	13
2.1.1	Architecture Diagram with IPA in the IoT Device	13
2.1.2	Architecture Diagram with IPA in the eUICC	14
2.1.3	ASN.1	15
2.2	Roles	16
2.3	Interfaces	16
2.4	eUICC Architecture	17
2.4.1	eUICC Architecture Diagram	17
2.4.1.1	eUICC Architecture Diagram (IPA in the IoT Device)	17
2.4.1.2	eUICC Architecture Diagram (IPA in the eUICC)	17
2.4.2	ECASD	18
2.4.3	ISD-R	18
2.4.4	ISD-P	18
2.4.5	Profile	19
2.4.5.1	Operational Profile	19
2.4.5.2	Provisioning Profile	19
2.4.5.3	Test Profile	19
2.4.6	Telecom Framework	19
2.4.7	Profile Package Interpreter	19
2.4.8	IPA in the eUICC	19
2.4.9	IPA Services	19
2.4.10	Hardware Characteristics of the eUICC	20
2.4.11	Platform Characteristics of the eUICC	20
2.4.12	Profile Policy Enabler	20
2.4.13	eUICC OS Update	20
2.4.14	eUICC Minimum Security Level Check	20
2.4.15	RSP Server Test Profile Allow-list Check	20
2.5	Profile Protection and Delivery	20
2.6	Security Overview	20
2.6.1	Relation to SGP.22	20
2.6.2	IoT Device Security	20
2.6.3	TLS Requirements	20

2.6.3.1	TLS Requirements for communication to the RSP Servers	21
2.6.3.2	TLS/DTLS Requirements for communication to the eIM	21
2.6.4	eIM Keys and eIM Certificates	22
2.6.4.1	eIM Keys and eIM Certificates Naming Conventions	22
2.6.4.2	Algorithms and Parameters for eIM Signing Key	22
2.6.4.3	eIM Signing Certificate (CERT.EIM.ECDSA)	23
2.6.4.4	eIM TLS/DTLS Certificate (CERT.EIM.TLS)	24
2.6.5	Use of Server Name indication extension (SNI) for ESipa.	26
2.7	Certificate Management	27
2.7.1	RSP Server Certificates	27
2.7.2	eUICC/EUM Certificates	27
2.7.3	eIM Certificates	27
2.8	Profile Policy Management	28
2.9	Profile State Management	28
2.9.1	Profile State Management by the eIM	28
2.9.2	Immediate Profile enabling of an IPA initiated Profile download from default SM-DP+	28
2.10	eIM Configuration	29
2.10.1	eIM Configuration by eIM	29
2.10.2	eIM configuration by IPA	29
2.11	eIM Package Structures	30
2.11.1	eIM Package Request	30
2.11.1.1	EuiccPackageRequest	30
2.11.1.2	IpaEuiccDataRequest	36
2.11.1.3	ProfileDownloadTriggerRequest	38
2.11.1.4	EimAcknowledgements	39
2.11.2	eIM Package Result	39
2.11.2.1	EuiccPackageResult	39
2.11.2.2	IpaEuiccDataResponse	43
2.11.2.3	ProfileDownloadTriggerResult	44
<b>3</b>	<b>Procedures</b>	<b>45</b>
3.1	eIM and IPA Communication	45
3.1.1	Transferring eIM Package	45
3.1.1.1	eIM Package Retrieval	45
3.1.1.2	eIM Package Injection	47
3.1.2	Secure Connection Establishment	48
3.1.2.1	Secure Connection using HTTP over TCP	49
3.1.2.2	Secure Connection over CoAP	50
3.1.2.3	Secure Connection by underlying transport layer	50
3.2	Profile Download and Installation	50
3.2.1	Profile Download Initiation	50
3.2.2	Common Mutual Authentication	50
3.2.3	Profile Download	51
3.2.3.1	Direct Profile Download	51
3.2.3.2	Indirect Profile Download	57

3.2.3.3	Cancel Session Procedure for Indirect Profile Download	65
3.2.3.4	Cancel Session Procedure for Direct Profile Download	68
3.3	eUICC Package Handling	68
3.3.1	Generic eUICC Package Download and Execution	68
3.3.2	Profile Rollback Procedure	74
3.4	Profile State Management	75
3.4.1	Enable Profile	75
3.4.2	Disable Profile	76
3.4.3	Delete Profile	78
3.4.4	Configure by eIM of Immediate Profile Enabling	80
3.4.5	Configure by IPA of Immediate Profile Enabling	82
3.4.6	Setting the Fallback Attribute for the target Profile	83
3.4.7	Unsetting the Fallback Attribute for the target Profile	86
3.5	eIM Configuration at eUICC	88
3.5.1	eIM Configuration Data managed by eIM	88
3.5.1.1	Addition of eIM Configuration Data	88
3.5.1.2	Deletion of eIM Configuration Data	89
3.5.1.3	Update of eIM Configuration Data	90
3.5.1.4	Request for a list of Associated eIMs	92
3.5.2	eIM Configuration Data managed by IPA	93
3.5.2.1	Addition of initial eIM Configuration Data	93
3.5.2.2	Deletion of all eIM Configuration Data	94
3.5.2.3	Reading eIM Configuration Data	95
3.6	Error handling within an RSP session	96
3.7	Notification Delivery to Notification Receivers	96
3.8	IoT Device and eUICC Initialisation	98
3.8.1	eUICC Initialisation	98
3.8.2	IoT Device Capabilities	98
3.8.3	eUICC File Structure	99
3.8.4	ISD-R Selection and IP Ae Activation	99
3.8.5	Triggering of the IP Ae for eIM Package retrieval	100
3.9	SM-DS Use	100
3.9.1	Event Registration	100
3.9.2	Event Retrieval	100
3.9.2.1	Event Retrieval by the IPA	100
3.9.2.2	Event Retrieval by the eIM	100
3.9.3	Event Deletion	104
<b>4</b>	<b>Data Elements</b>	<b>104</b>
4.1	IPA Capabilities	104
4.2	eIM Supported Protocol	107
4.3	eIM Identifier	107
4.4	Profile MetaData	108
<b>5</b>	<b>Functions</b>	<b>109</b>
5.1	Overview of Functions per Interface	109

5.2	Server to Server Function Commonalities	112
5.2.1	Common Data Types	112
5.2.2	Request-Response Function	112
5.2.3	Notification Handler Function	112
5.2.4	Functions Input Header	112
5.2.5	Functions Output Header	112
5.2.6	Status Code	112
5.2.6.1	Subject Code	112
5.2.6.2	Reason Code	114
5.2.6.3	Common Function Status Code	115
5.3	ES2+ (Operator -- SM-DP+)	115
5.4	ES6 (Operator -- eUICC)	115
5.4.1	Function: ConnectivityParametersUpdateByOperator	115
5.4.2	Function: UpdateMetadata	116
5.5	ES8+ (SM-DP+ -- eUICC)	118
5.6	ES9+ (IPA -- SM-DP+)	120
5.6.1	Function (ES9+): AuthenticateClient	120
5.7	ES9+' (eIM -- SM-DP+)	121
5.7.1	Function (ES9+'): InitiateAuthentication	121
5.7.2	Function (ES9+'): GetBoundProfilePackage	121
5.7.3	Function (ES9+'): AuthenticateClient	122
5.7.4	Function (ES9+'): HandleNotification	122
5.7.5	Function (ES9+'): CancelSession	122
5.8	ES10a (IPA -- eUICC)	123
5.8.1	Function (ES10a): GetEuiccConfiguredAddresses	123
5.9	ES10b (IPA -- eUICC)	123
5.9.1	Function (ES10b): LoadEuiccPackage	123
5.9.2	Function (ES10b): GetEUICCInfo	125
5.9.3	Function (ES10b): GetEUICCChallenge	127
5.9.4	Function (ES10b): AddInitialEim	127
5.9.5	Function (ES10b): eUICCMemoryReset	129
5.9.6	Function (ES10b): AuthenticateServer	132
5.9.7	Function (ES10b): PrepareDownload	132
5.9.8	Function (ES10b): LoadBoundProfilePackage	132
5.9.9	Function (ES10b): CancelSession	132
5.9.10	Function (ES10b): GetCerts	133
5.9.11	Function (ES10b): RetrieveNotificationsList	133
5.9.12	Function (ES10b): RemoveNotificationFromList	135
5.9.13	Function (ES10b): GetRAT	135
5.9.14	Function (ES10b): GetProfilesInfo	135
5.9.15	Function (ES10b): ImmediateEnable	135
5.9.16	Function (ES10b): ProfileRollback	137
5.9.17	Function (ES10b): ConfigureImmediateProfileEnabling	140
5.9.18	Function (ES10b): GetEimConfigurationData	141
5.9.19	Function (ES10b): GetEID	142

5.9.20	Function (ES10b): ExecuteFallbackMechanism	142
5.9.21	Function (ES10b): ReturnFromFallback	144
5.9.22	Function (ES10b): EnableEmergencyProfile	146
5.9.23	Function (ES10b): DisableEmergencyProfile	148
5.9.24	Function (ES10b): GetConnectivityParameters	150
5.9.25	Function (ES10b): SetDefaultDpAddress	151
5.10	ES11 (IPA -- SM-DS)	152
5.11	ES11' (eIM -- SM-DS)	152
5.11.1	Function (ES11'): InitiateAuthentication	152
5.11.2	Function (ES11'): AuthenticateClient	153
5.12	ES12 (SM-DS -- SM-DP+)	153
5.13	ESep (eIM -- eUICC)	153
5.13.1	Function (ESep): Enable	153
5.13.2	Function (ESep): Disable	154
5.13.3	Function (ESep): Delete	154
5.13.4	Function (ESep): ListProfileInfo	154
5.13.5	Function (ESep): GetRAT	154
5.13.6	Function (ESep): ConfigureImmediateEnable	155
5.13.7	Function (ESep): AddEim	155
5.13.8	Function (ESep): UpdateEim	155
5.13.9	Function (ESep): DeleteEim	155
5.13.10	Function (ESep): ListEim	156
5.13.11	Function (ESep): SetDefaultDpAddress	156
5.14	ESipa (eIM -- IPA)	156
5.14.1	Function (ESipa): InitiateAuthentication	156
5.14.2	Function (ESipa): GetBoundProfilePackage	159
5.14.3	Function (ESipa): AuthenticateClient	162
5.14.4	Function (ESipa): TransferEimPackage	165
5.14.5	Function (ESipa): GetEimPackage	168
5.14.6	Function (ESipa): ProvideEimPackageResult	170
5.14.7	Function (ESipa): HandleNotification	172
5.14.8	Function (ESipa): CancelSession	175
<b>6</b>	<b>Interface Binding</b>	<b>176</b>
6.1	ESipa interface binding over HTTP	177
6.1.1	ASN.1 message definition	177
6.1.2	JSON message definition	178
6.2	ESipa interface binding over CoAP	178
6.3	ESipa function binding in ASN.1	179
6.3.1	ASN.1 structures for ESipa messages between IPA and eIM	180
6.3.2	ESipa functions provided by eIM	180
6.3.2.1	ESipa.InitiateAuthentication	180
6.3.2.2	ESipa.AuthenticateClient	181
6.3.2.3	ESipa.GetBoundProfilePackage	182
6.3.2.4	ESipa.HandleNotification	182
6.3.2.5	ESipa.CancelSession	182

6.3.2.6	ESipa.GetEimPackage	183
6.3.2.7	ESipa.ProvideEimPackageResult	183
6.3.3	ESipa functions provided by IPA	184
6.3.3.1	ESipa.TransferEimPackage	184
6.4	ESipa function binding in JSON	185
6.4.1	List of functions	185
6.4.1.1	ESipa.InitiateAuthentication	185
6.4.1.2	ESipa.AuthenticateClient	187
6.4.1.3	ESipa.GetBoundProfilePackage	188
6.4.1.4	ESipa.TransferEimPackage	189
6.4.1.5	ESipa.GetEimPackage	191
6.4.1.6	ESipa.ProvideEimPackageResult	192
6.4.1.7	ESipa.HandleNotification	193
6.4.1.8	ESipa.CancelSession	194
<b>Annex A</b>	<b>IoT Device Requirements (Normative)</b>	<b>194</b>
A.1	Functional IoT Device Requirements	194
A.2	Support for CAT Mechanisms	194
<b>Annex B</b>	<b>Mapping of procedures to IoT protocols (Informative)</b>	<b>195</b>
B.1	LwM2M	195
B.1.1	Secure connection	195
B.1.2	LwM2M object	195
B.1.3	Procedures	196
B.1.3.1	Transferring messages between eIM and IPA (LwM2M)	196
B.2	MQTT	197
B.2.1	Mapping of MQTT Roles	197
B.2.2	Secure connection	197
B.2.3	Description	198
B.2.4	Procedures	199
B.2.4.1	Connection	199
B.2.4.2	Communication (Retrieval)	200
B.2.4.3	Communication (Injection)	202
<b>Annex C</b>	<b>ASN.1 Definitions (Normative)</b>	<b>203</b>
<b>Annex D</b>	<b>Tag allocation (Normative)</b>	<b>218</b>
<b>Annex E</b>	<b>Document Management (Informative)</b>	<b>221</b>
E.1	Document History	221

# 1 Introduction

## 1.1 Overview

This document provides a technical description of the GSMA's eSIM IoT Architecture and Requirements SGP.31 [1] specification.

## 1.2 Scope

This specification provides a technical description of:

- The remote provisioning and management of the eUICC in IoT Network Constrained and/or User Interface Constrained Devices.
- The eUICC architecture for IoT Devices.
- The interfaces used within the eSIM IoT architecture; and
- The security functions used within the eSIM IoT architecture.

## 1.3 Document Purpose

This document defines a technical solution for the remote provisioning and management of the eUICC in IoT Devices as defined in eSIM IoT Architecture and Requirements SGP.31 [1]. The adoption of this technical solution will provide the basis for global interoperability between different deployment scenarios supported by the eSIM IoT Architecture and Requirements SGP.31 [1].

## 1.4 Intended Audience

Technical experts working for Operators, eUICC solution providers, IoT Device vendors, standards organisations, network infrastructure vendors, Mobile Service Providers and IoT service providers and other impacted industry bodies.

## 1.5 Definition of Terms

Term	Description
Activation Code	As defined in SGP.22 [4].
Activation Code Token	As defined in SGP.22 [4].
Associated eIM	As defined in SGP.31 [1].
Bound Profile Package	As defined in SGP.31 [1].
Certificate	Public key certificate.
Certification Authority	An authority which issues public key certificates.
Connectivity Parameters	A set of data (e.g. APN, login, password, ...) required by the IPA to open a communication channel (e.g. HTTPS, CoAP) on a dedicated network.
Device Baseband	A device component that manages the cellular radio functions.
eIM Configuration Data	As defined in SGP.31 [1].
eIM Configuration Operation	As defined in SGP.31 [1].

eIM Package	Contains an eUICC Package, a request for IPA and/or eUICC data, a Profile download trigger (e.g., containing an Activation Code) or acknowledgements of eUICC Package Results and Notifications.
eIM Package Result	Execution result of an eIM Package.
Emergency Mechanism	As defined in SGP.31 [1].
Emergency Profile	As defined in SGP.31 [1].
eUICC	As defined in SGP.22 [4].
eUICC Minimum Security Level Check	As defined in SGP.22 [4].
eUICC OS Update	As defined in SGP.22 [4].
eUICC Package	A cryptographically signed package prepared by the eIM that contains either PSMO(s) or eCO(s). The package is verified and executed by the eUICC.
eUICC Package Result	A cryptographically signed package prepared by the eUICC that contains the execution result of the eUICC Package.
Event	As defined in SGP.22 [4].
EventID	As defined in SGP.22 [4].
Event Deletion	A process of deleting an Event from an SM-DS by the SM-DP+.
Event Record	As defined in SGP.22 [4].
Event Registration	As defined in SGP.22 [4].
Event Retrieval	A process of retrieving from the SM-DS any available information for a specific eUICC registered by a specific SM-DP+ or a specific SM-DS.
Fallback Attribute	This is an attribute of a Profile which, when set, identifies the Profile to be enabled by the Fallback Mechanism.
Fallback Mechanism	eUICC-based mechanism which enables and disables the Profile with Fallback Attribute set.
Fallback Profile	An Operational Profile having the Fallback Attribute set.
Field-Test eUICC	As defined in SGP.22 [4]
IoT	As defined in SGP.31 [1].
IoT Device	As defined in SGP.31 [1].
IPA Capabilities	Indicates to the eIM what the IPA is capable of related to the functional split of Profile download between the eIM and the IPA and the use of compact data object structures in ESipa functions.
MatchingId	As defined in SGP.22 [4].
Mobile Service Provider	As defined in SGP.31 [1].
Network Constrained Device	As defined in SGP.31 [1].
Notification	As defined in SGP.22 [4].
Notification Receivers	As defined in SGP.31 [1].
Operational Profile	As defined in SGP.22 [4].

Operator	As defined in SGP.22 [4].
Profile	As defined in SGP.22 [4].
Profile Metadata	As defined in SGP.22 [4].
Profile Package	As defined in SGP.31 [1].
Profile Package Interpreter	An eUICC Operating System service that translates the Profile Package data into an installed Profile using the specific internal format of the target eUICC.
Profile State Management Operation	As defined in SGP.31 [1].
Provisioning Profile	As defined in SGP.22 [4].
Registered PLMN	As defined in 3GPP TS 23.122 [19].
Rollback Mechanism	As defined in SGP.31 [1].
Root SM-DS	As defined in SGP.22 [4].
RSP Server	As defined in SGP.22 [4].
Rules Authorisation Table	As defined in SGP.22 [4].
Signed eCO	eIM Configuration Operation performed by an Associated eIM using cryptographically signed eIM Configuration Operation that contains the eIM Configuration Data.
Subscription Manager Data Preparation	As defined in SGP.22 [4].
Subscription Manager Discovery Server	As defined in SGP.22 [4].
Telecom Framework	An eUICC Operating System service that provides standardised network authentication algorithms to the network authentication applications hosted in the ISD-Ps. Furthermore, it provides capabilities to configure the algorithm with necessary parameters in the Enabled Profile.
Test Profile	As defined in SGP.22 [4].
Unsigned eCO	Non-cryptographically signed eIM Configuration Operation of the initial eIM.
User Interface Constrained Device	As defined in SGP.31 [1].

## 1.6 Abbreviations

Abbreviation	Description
CA	Certification Authority
CoAP	Constrained Application Protocol.
CR	Comprehension Required
CRL	Certificate Revocation List
DTLS	Datagram Transport Layer Security.
ECASD	eUICC Controlling Authority Security Domain
eCO	eIM Configuration Operation

EID	<b>eUICC-ID</b> entifier as defined in SGP.22 [4]
eIM	<b>eSIM IoT Remote M</b> anager
eIMID	<b>eIM I</b> dentifier
EUM	<b>eUICC M</b> anufacturer
GSMA CI	<b>GSMA C</b> ertificate Issuer
IoT	<b>I</b> nternet of Things
IPA	<b>IoT P</b> rofile Assistant
IPAd	<b>IoT P</b> rofile Assistant in the IoT <b>D</b> evice
IP Ae	<b>IoT P</b> rofile Assistant in the <b>eUICC</b>
ISD-P	<b>I</b> ssuer <b>S</b> ecurity <b>D</b> omain – <b>P</b> rofile
ISD-R	<b>I</b> ssuer <b>S</b> ecurity <b>D</b> omain – <b>R</b> oot
NCD	<b>N</b> etwork <b>C</b> onstrained <b>D</b> evice
OCSP	<b>O</b> nline <b>C</b> ertificate <b>S</b> tatus <b>P</b> rotocol
OTA	<b>O</b> ver <b>T</b> he <b>A</b> ir
PSMO	<b>P</b> rofile <b>S</b> tate <b>M</b> anagement <b>O</b> peration
RAT	<b>R</b> ules <b>A</b> uthorisation <b>T</b> able
RPLMN	<b>R</b> egistered <b>P</b> ublic <b>L</b> and <b>M</b> obile <b>N</b> etwork
SM-DP+	<b>S</b> ubscription <b>M</b> anager <b>D</b> ata <b>P</b> reparation +
SM-DS	<b>S</b> ubscription <b>M</b> anager <b>D</b> iscovery <b>S</b> erver
TLS	<b>T</b> ransport <b>L</b> ayer <b>S</b> ecurity.
UICD	<b>U</b> ser <b>I</b> nterface <b>C</b> onstrained <b>D</b> evice

## 1.7 References

Ref	Document Number	Title
[1]	SGP.31	eSIM IoT Architecture and Requirements, version 1.3
[2]	RFC 2119	Key words for use in RFCs to Indicate Requirement Levels, S. Bradner
[3]	RFC 8174	Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words
[4]	SGP.22	RSP Technical Specification, version 2.7
[5]	ETSI TS 102 223	Smart Cards; Card Application Toolkit (CAT)
[6]	SGP.02	GSMA Remote Provisioning of Embedded UICC Technical specification version 4.3
[7]	RFC 7252	The Constrained Application Protocol (CoAP)
[8]	TLS 1.2 RFC 5246	The Transport Layer Security (TLS) Protocol Version 1.2
[9]	TLS 1.3 RFC 8446	The Transport Layer Security (TLS) Protocol Version 1.3
[10]	RFC 6347	Datagram Transport Layer Security Version 1.2

[11]	RFC 9147	The Datagram Transport Layer Security (DTLS) Protocol Version 1.3
[12]	RFC 7959	Block-Wise Transfers in the Constrained Application Protocol (CoAP)
[13]	RFC 9175	Constrained Application Protocol (CoAP): Echo, Request-Tag, and Token Processing
[14]	RFC 9146	Connection Identifier for DTLS 1.2
[15]	RFC 8422	Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and earlier
[16]	RFC 7027	Elliptic Curve Cryptography (ECC) Brainpool Curves for Transport Layer Security (TLS)
[17]	RFC 5289	TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)
[18]	RFC 8410	Algorithm Identifiers for Ed25519, Ed448, X25519, and X448 for Use in the Internet X.509 Public Key Infrastructure
[19]	3GPP TS 23.122	3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; Non-Access-Stratum (NAS) functions related to Mobile Station (MS) in idle mode
[20]	RFC 5077	Transport Layer Security (TLS) Session Resumption without Server-Side State
[21]	RFC 6960	X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP
[22]	3GPP TS 24.008	3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; Mobile radio interface Layer 3 specification; Core network protocols; Stage 3
[23]	MQTTs	MQTTs version 5.0 based on TLS
[24]	RFC 5639	Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation
[25]	RFC 5280	Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile
[26]	ANSSI ECC	Avis relatif aux paramètres de courbes elliptiques définis par l'Etat français. JORF n°0241 du 16 octobre 2011 page 17533. texte n° 30. 2011
[27]	RFC 5480	Elliptic Curve Cryptography Subject Public Key Information
[28]	ISO 14888-3	IT Security techniques — Digital signatures with appendix — Part 3: Discrete logarithm-based mechanisms
[29]	TCA eUICC Profile Package	eUICC Profile Package: Interoperable Format Technical Specification
[30]	OMA LwM2M Object and Resource Registry	OMA LwM2M Object and Resource Registry : <a href="https://technical.openmobilealliance.org/OMNA/LwM2M/LwM2MRegistry.html">https://technical.openmobilealliance.org/OMNA/LwM2M/LwM2MRegistry.html</a>

[31]	RFC 2818	HTTP Over TLS.
[32]	GPC_SPE_034	GlobalPlatform Technology Card Specification v2.3.1
[33]	GPD_SPE_075	GlobalPlatform Technology Open Mobile API Specification v3.3
[34]	ETSI 102 221	Smart Cards; UICC-Terminal interface; Physical and logical characteristics
[35]	RFC 6066	Transport Layer Security (TLS) Extensions: Extension Definitions
[36]	RFC 7159	The JavaScript Object Notation (JSON) Data Interchange Format

## 1.8 Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [2] and clarified by RFC 8174 [3], when, and only when, they appear in all capitals, as shown here.

## 1.9 References to SGP.22

The present document contains several references to SGP.22 [4].

The following list maps the terms used by SGP.22 [4] to the terms used in the present document:

- LPA (Local Profile Assistant) refers to IPA.

The following features defined in SGP.22 [4] are not applicable in the scope of this specification:

- Activation Code Retrieval, including Activation Code Retrieval availability check

## 2 General Architecture

This section contains a technical description and architecture of the eSIM IoT system for IoT Devices. The statements in this section define the basic characteristics that need to be taken into account when reviewing this specification.

### 2.1 General Architecture

This section further specifies the roles and interfaces associated with the Remote SIM Provisioning and Management of the eUICC for IoT Devices.

#### 2.1.1 Architecture Diagram with IPA in the IoT Device

This section further specifies the roles and interfaces associated with the Remote SIM Provisioning and Management with the IPA in the IoT Device (Figure 1).

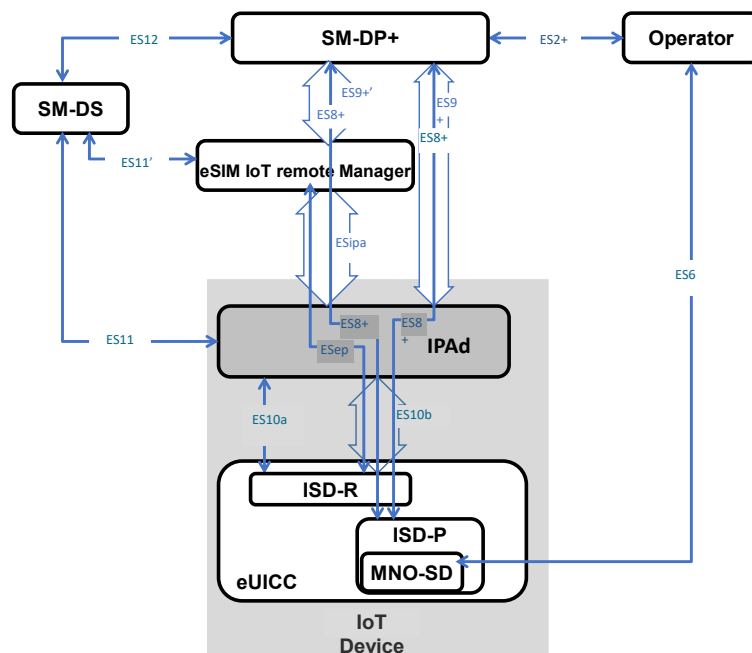


Figure 1 eSIM IoT Functional Architecture (IPA in the IoT Device)

### 2.1.2 Architecture Diagram with IPA in the eUICC

This section further specifies the roles and interfaces associated with the Remote SIM Provisioning and Management with the IPA in the eUICC (Figure 2).

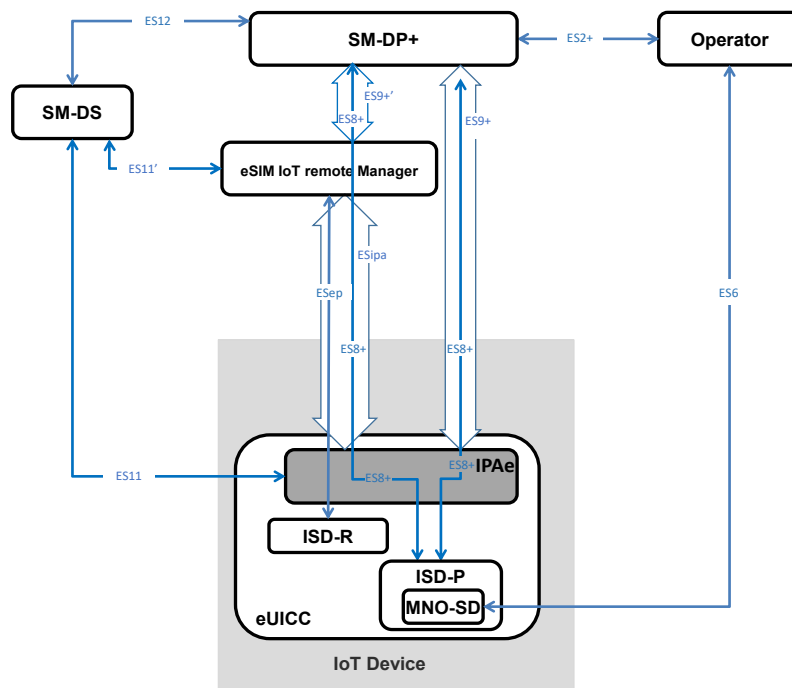


Figure 2 eSIM IoT Functional Architecture (IPA in the eUICC)

### 2.1.3 ASN.1

The description of some data objects in this specification is based on ASN.1 specified in ITU-T X.680 [49] and encoded in TLV structures using DER (Distinguished Encoding Rule) encoding as specified in ITU-T X.690 [50]. This provides a flexible description of those data objects.

Data structures in these specifications are defined in a single, self-contained, ASN.1 definition module called `SGP32Definitions`, with an ISO Object Identifier in the GSMA namespace. A number of definitions are imported from the `RSPDefinitions` module defined in SGP.22 [4].

```
-- ASN1START
SGP32Definitions {joint-iso-itu-t(2) international-organizations(23) gsma(146) rsp(1)
asn1modules(1) sgp32v1(31)}
DEFINITIONS
AUTOMATIC TAGS
EXTENSIBILITY IMPLIED ::=
BEGIN

IMPORTS Certificate, SubjectPublicKeyInfo FROM PKIX1Explicit88 {iso(1) identified-
organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-
pkix1-explicit(18)}
SubjectKeyIdentifier FROM PKIX1Implicit88 {iso(1) identified-organization(3) dod(6)
internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-pkix1-implicit(19)}
CancelSessionReason, ServerSigned1, RspCapability, Iccid, TransactionId,
ProfileInfoListRequest, RulesAuthorisationTable, EUICCInfo1, DeviceInfo,
VersionType, UICCCapability, PprIds, CertificationDataObject, Octet1, Octet16,
Octet32, PrepareDownloadResponseOk, PrepareDownloadResponseError,
AuthenticateResponseError, CtxParams1, ProfileInstallationResultData,
OtherSignedNotification, EuiccSignPIR, ErrorResult, NotificationMetadata,
CancelSessionResponseOk, SmdpSigned2, BoundProfilePackage, NotificationEvent,
VendorSpecificExtension, DpProprietaryData, OperatorId,
NotificationConfigurationInformation, ProfileClass, IconType, ProfileState,
OctetTo16 FROM RSPDefinitions {joint-iso-itu-t(2) international-organizations(23)
gsma(146) rsp(1) asn1modules(1) sgp22v2(2)};
-- ASN1STOP
```

Two encoding/decoding attributes are defined:

- **AUTOMATIC TAGS** means that the tags are defined automatically using the encoding rules unless a tag notation is present in a data object format definition.
- **EXTENSIBILITY IMPLIED** means that types MAY have elements that are not defined in this specification. This means that decoders SHALL be able to handle values with unspecified tags, either by processing them if they know their value content or ignoring them silently (without reporting an error) if they do not know them. This is useful when processing data definitions from a newer specification and to handle proprietary tag values.

As the eUICC cannot implement an off-the-shelf standard decoder, the requirement on extensibility SHALL NOT apply to the eUICC.

NOTE 1: As specified in ITU-T X.690 [50], the sender omits the encoding of fields marked as "DEFAULT" when its value equals the default value.

NOTE 2: DER encoding of a BIT STRING removes all trailing bits that are set to '0'.

## 2.2 Roles

Roles are defined within SGP.31 [1] section 3.

## 2.3 Interfaces

The following table provides information about the interfaces within the architecture.

Interface	Between		Description
ES2+	Operator	SM-DP+	Used by the Operator to order Profiles for specific eUICCs as well as other administrative functions as defined in SGP.31 [1].
ES6	Operator	eUICC	Used by the Operator for the management of Operator services via OTA services as defined in SGP.31 [1].
ES8+	SM-DP+	eUICC	Provides a secure end-to-end channel between the SM-DP+ and the eUICC for the administration of the ISD-P and the associated Profile during download and installation. It provides Perfect Forward Secrecy as defined in SGP.31 [1].
ES9+	SM-DP+	IPA	Used to provide a secure transport between the SM-DP+ and the IPA for the delivery of the Bound Profile Package as defined in SGP.31 [1].
ES9+'	SM-DP+	eIM	Used to provide a secure transport between the SM-DP+ and the eIM for the delivery of the Bound Profile Package as defined in SGP.31 [1].
ES10a	IPA	eUICC	Used between the IPA (in the IoT Device) and the eUICC to handle a Profile discovery as defined in SGP.31 [1].
ES10b	IPA	eUICC	Used between the IPA (in the IoT Device) and the IPA Services to transfer a Bound Profile Package to the eUICC as defined in SGP.31 [1]. This interface plays no role in the decryption of Profile Packages.
ES11	IPA	SM-DS	Used by the IPA to retrieve Event Records for the respective eUICC as defined in SGP.31 [1].
ES11'	eIM	SM-DS	Used by the eIM to retrieve Event Records for the respective eUICC as defined in SGP.31 [1].
ES12	SM-DP+	SM-DS	Used by the SM-DP+ to issue or remove Event Registrations on the SM-DS as defined in SGP.31 [1].
ESep	eIM	eUICC	Logical end-to-end interface between the eIM and the eUICC used to transfer eUICC Packages for Profile State management and eIM configuration by eIM, as defined in SGP.31 [1].
ESipa	eIM	IPA	Logical interface between an eIM and an IPA, as defined in SGP.31 [1], used to trigger a Profile download at the IPA and to provide a secure transport for the delivery of eUICC Packages, unless the underlying transport provides necessary security.

**Table 1 Interfaces**

NOTE: Support of the ES10c interface as defined in SGP.22 [4] is out of scope of this specification.

## 2.4 eUICC Architecture

This section describes the internal high-level architecture of the eUICC.

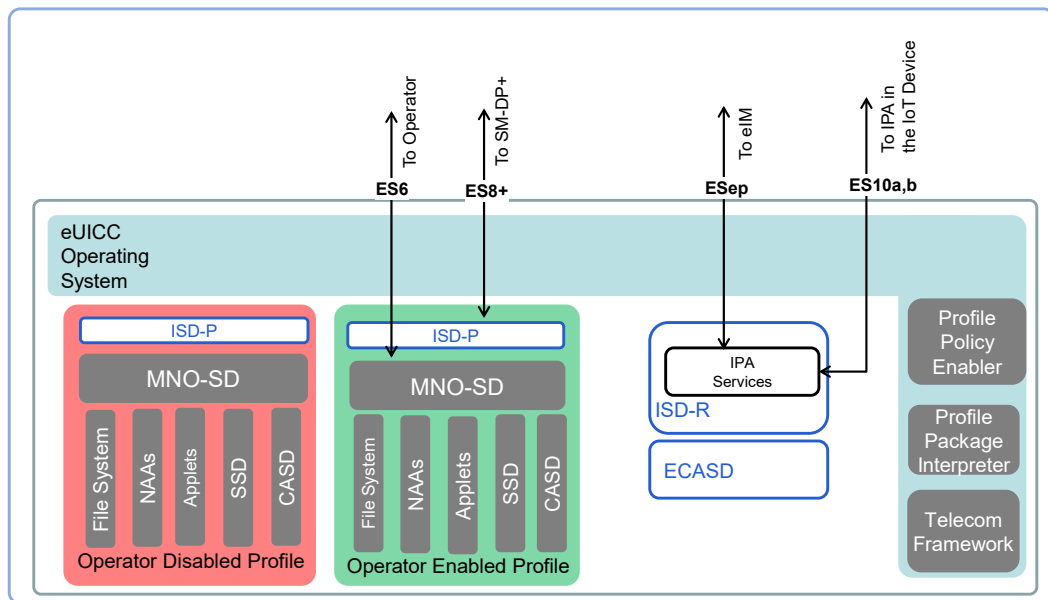
NOTE: The eUICC architecture is very similar to that used in SGP.22 [4].

### 2.4.1 eUICC Architecture Diagram

The IPA SHALL be located either in the IoT Device (IPAd) or in the eUICC (IPAE).

#### 2.4.1.1 eUICC Architecture Diagram (IPA in the IoT Device)

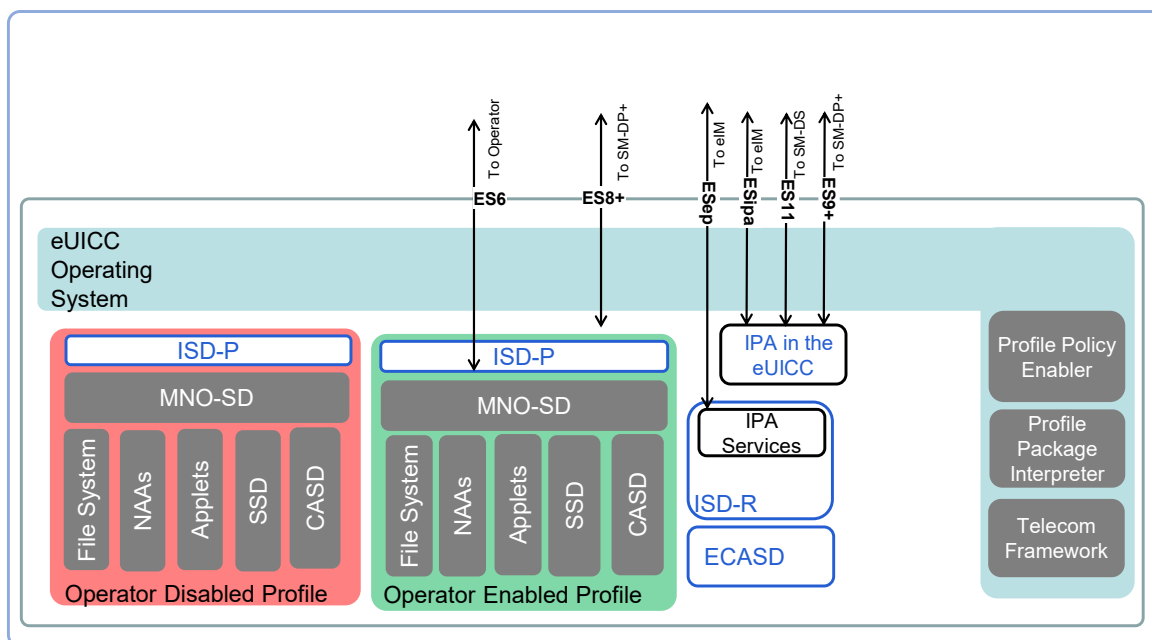
The following diagram represents the eUICC architecture when the IPA is located in the IoT Device.



**Figure 3 Schematic Representation of the eUICC (IPA in the Device)**

#### 2.4.1.2 eUICC Architecture Diagram (IPA in the eUICC)

The following diagram represents the eUICC architecture when the IPA is located in the eUICC.



**Figure 4 Schematic Representation of the eUICC (IPA in the eUICC)**

### 2.4.2 ECASD

The eUICC Controlling Authority Security Domain (ECASD) as defined in SGP.22 [4].

### 2.4.3 ISD-R

The ISD-R as defined in SGP.22 [4].

### 2.4.4 ISD-P

The ISD-P entity as defined in SGP.22 [4].

In addition to the current ISD-P as defined in SGP.22, the Connectivity Parameters MAY be included in the Profile as per the following tables:

DGI	Length	Value Description		MOC	
'3A07'	Var	Connectivity Parameters		M	
		Tag	Length	Value Description	MOC
		'A1'	n	Connectivity parameters for HTTP and CoAP protocol	C

**Table 2 Connectivity Parameters Data Field**

The coding of Connectivity Parameters is defined as follows. Comprehension TLVs as defined in ETSI TS 102 223 [5]. The Comprehension Required (CR) bit of the tags SHALL be set to zero:

Description
Bearer description
Network Access Name (NAN)
User Login
User Password

**Table 3 Coding of Connectivity Parameters for HTTP and CoAP**

If the Connectivity Parameters are set in the enabled Profile, the IPA MAY use these to setup a communication channel with eIM.

NOTE: If the Connectivity Parameters were not set at the installation of the profile, it will be possible to update the Connectivity Parameters through ES6.

## 2.4.5 Profile

The Profile as defined in SGP.22 [4].

### 2.4.5.1 Operational Profile

The Operational Profile as defined in SGP.22 [4].

### 2.4.5.2 Provisioning Profile

The Provisioning Profile as defined in SGP.22 [4].

### 2.4.5.3 Test Profile

The Test Profile as defined in SGP.22 [4].

## 2.4.6 Telecom Framework

The Telecom Framework as defined in SGP.22 [4].

## 2.4.7 Profile Package Interpreter

The Profile Package Interpreter as defined in SGP.22 [4].

## 2.4.8 IPA in the eUICC

The IPAe is a functional element which provides similar features as those provided by the IPA in the IoT Device. The implementation of IPAe is OPTIONAL.

The technical implementation of the IPAe is EUM-specific.

## 2.4.9 IPA Services

This role provides the necessary access to the services and data required by functions of the IPA. These services include:

- Provide the address of the Root SM-DS and (if configured) the default SM-DP+
- Transfer Bound Profile Package from the IPAd to the ISD-P
- Provide list of installed Profiles and their Profile Metadata
- Retrieve EID
- Provide Profile State Management Operations

- Provide eUICC execution results and Notifications.

#### **2.4.10 Hardware Characteristics of the eUICC**

The Hardware Characteristics of the eUICC are defined in section 2.4.10 of SGP.22 [4].

#### **2.4.11 Platform Characteristics of the eUICC**

The Platform Characteristics of the eUICC are defined in section 2.4.11 of SGP.22 [4].

#### **2.4.12 Profile Policy Enabler**

The Profile Policy Enabler is defined in section 2.4.12 of SGP.22 [4].

#### **2.4.13 eUICC OS Update**

The eUICC OS Update is defined in section 2.4.13 of SGP.22 [4].

#### **2.4.14 eUICC Minimum Security Level Check**

The eUICC Minimum Security Level Check is defined in section 2.4.14 of SGP.22 [4].

#### **2.4.15 RSP Server Test Profile Allow-list Check**

The RSP Server Test Profile Allow-list Check is defined in section 2.4.15 of SGP.22 [4].

### **2.5 Profile Protection and Delivery**

An Operator's Profile is protected within a Profile Package prior to being downloaded to the eUICC as defined in section 2.5 of SGP.22 [4].

### **2.6 Security Overview**

#### **2.6.1 Relation to SGP.22**

This specification SHALL re-use the overall security features specified in SGP.22 [4] section 2.6, unless otherwise specified.

#### **2.6.2 IoT Device Security**

IoT Devices will be located in a wide variety of environments. It is a basic assumption that in addition to the measures described in this specification, these IoT Devices will have adequate security measures applied in order to provide whatever protection is required to satisfy the security policy for the use-case.

Descriptions of these additional measures are out of scope of this specification.

#### **2.6.3 TLS Requirements**

TLS v1.2 as defined in RFC 5246 [8] SHALL be the minimal version for any TLS connection in this specification.

DTLS v1.2 as defined in RFC 6347 [10] SHALL be the minimal version for any DTLS connection in this specification.

### 2.6.3.1 TLS Requirements for communication to the RSP Servers

For communication with SM-DP+ and SM-DS, all the TLS requirements defined in section 2.6.6 and 6.1 (and its subsections) of SGP.22 [4] SHALL apply, where the IPA plays the role of the LPA. In addition, the following requirements apply:

- TLS with server authentication SHALL be used over ES9+' (eIM with SM-DP+), and ES11' (eIM with SM-DS).

NOTE: Here the eIM is playing the role of the LPA.

- TLS with mutual authentication MAY be used over ES2+. If TLS with mutual authentication is not used over ES2+, ES2+ SHALL be protected with level of security equivalent to TLS.
- When using TLS 1.2, an RSP Server (TLS server) designed to support NCD (TLS client), that receives an extended ClientHello containing a "max\_fragment\_length" extension, SHOULD accept the requested maximum fragment length by including an extension of type "max\_fragment\_length" in the (extended) ServerHello (see RFC 6066 [35]).

NOTE: To avoid interoperability issues caused by the lack of support for the max\_fragment\_length extension in RSP Servers, Indirect Profile Download SHOULD be used for NCD.

### 2.6.3.2 TLS/DTLS Requirements for communication to the eIM

NOTE: The following TLS/DTLS requirements for communication with the eIM applies to section 3.1.2.1 and 3.1.2.2.

TLS v1.3 as defined in RFC 8446 [9] and DTLS v1.3 as defined in RFC 9147 [11] MAY be supported for the communication over ESipa (IPA with eIM).

TLS with server authentication SHALL be used. The client (e.g., IPA) either directly trusts the server (i.e., eIM) Certificate or public key, or trusts the CA to which the server Certificate chains back or trusts an intermediate CA along the path.

The following elliptic curve(s) SHALL be supported:

- NIST P-256 with the namedCurve secp256r1 (23) as defined in RFC 8422 [15]

The following elliptic curves MAY be supported:

- brainpoolP256r1(26) as defined in RFC 7027 [16],
- NIST P-384 with the namedCurve secp384r1 (24) as defined in RFC 8422 [15],
- Ed25519 as defined in RFC 8410 [18], and x25519 with the namedCurve x25519 (29) as defined in RFC 8422 [15]

The following cipher suites SHALL be supported:

- For (D)TLS 1.2
  - TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_GCM\_SHA256 as defined in RFC 5289 [17]

- TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_GCM\_SHA384 as defined in RFC 5289 [17]
- For (D)TLS 1.3
  - TLS\_AES\_128\_GCM\_SHA256 as defined in TLS 1.3 (RFC 8446 [9])
  - TLS\_AES\_256\_GCM\_SHA384 as defined in TLS 1.3 (RFC 8446 [9])

The following cipher suites MAY be supported:

- For (D)TLS 1.2
  - TLS\_ECDHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384 as defined in RFC 5289 [17]
- For (D)TLS 1.3
  - TLS\_CHACHA20\_POLY1305\_SHA256 as defined in TLS 1.3 (RFC 8446 [9])

## 2.6.4 eIM Keys and eIM Certificates

### 2.6.4.1 eIM Keys and eIM Certificates Naming Conventions

The eIM keys and eIM Certificates used in this specification are named according to the conventions described in this section.

The general name structure is: <XX>. <YY>. <ZZ>

Where:

- <XX> designates the nature of the element, the following values are defined:
  - PK: the public key of an asymmetric key pair
  - SK: the private key of an asymmetric key pair
  - CERT: a Certificate containing a public key
- <YY> designates the owner of the element, the following values are defined:
  - EIM: an eIM
- <ZZ> designates the usage of the element, the following values are defined:
  - ECDSA: for a digital signature
  - TLS: for TLS connection establishment
  - DTLS: for DTLS connection establishment

Examples:

- PK.EIM.ECDSA: Public key of an eIM, used to verify an eIM signature.
- CERT.EIM.TLS: Certificate of the eIM, used to establish TLS connection.

### 2.6.4.2 Algorithms and Parameters for eIM Signing Key

This section provides the values to be set in 'AlgorithmIdentifier.algorithm' and 'AlgorithmIdentifier.parameters' fields of the eIM signing key for each of the algorithms used in this specification.

For section 'subjectPublicKeyInfo' the following settings SHALL apply:

'AlgorithmIdentifier.algorithm' field SHALL be set to: "iso (1) member-body (2) us (840) ansi-X9-62(10045) keyType (2) ecPublicKey (1)" as defined in RFC 5480 [27], or

'AlgorithmIdentifier.parameters' field SHALL be set to:

- for BrainpoolP256r1: "iso (1) identified-organization (3) teletrust (36) algorithm (3) signatureAlgorithm (3) ecSign (2) ecStdCurvesAndGeneration (8) ellipticCurve (1) versionOne (1) brainpoolP256r1 (7)" as defined in RFC 5639 [24]
- for NIST P-256: "iso (1) member-body (2) us (840) ansi-X9-62 (10045) curves (3) prime (1) prime256v1(7)" as defined in RFC 5480 [27]

### 2.6.4.3 eIM Signing Certificate (CERT.EIM.ECDSA)

This specification supports X.509 certificate format as defined in RFC 5280 [25].

The table below describes the common fields that a CERT.EIM.ECDSA SHALL contain.

Certificate common fields		
Field	Value Description	
tbsCertificate	Data to be signed	
	Field	Value Description
	version	Version SHALL be 3.
	serialNumber	Certificate serial number.
	signature	Contains the algorithm identifier used to compute the value of the field 'signatureValue'. NOTE: The algorithm identifier value SHALL be the same as the one of the field 'signatureAlgorithm'.
	issuer	Distinguished Name of the entity that has signed the certificate.
	validity	Validity period of the eIM Certificate.
	subject	Distinguished Name of the entity owning the certificate. The following DN components SHALL be used: CN=<eIMiD> -- mandatory C=<country code> -- mandatory O=<company> -- optional OU=<organizational unit> -- optional L=<locality> -- optional
	subjectPublicKeyInfo	Contains the algorithm identifier, parameters, and public key value. Algorithm identifier and parameters SHALL be set according to section 2.6.4.2 subjectPublicKeyInfo.subjectPublicKey contains the public key value and SHALL be coded as defined in RFC 5480 [27].

	Extension for Authority Key Identifier (RFC 5280 [25] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier critical = false extnValue = <Identifier of the public key to verify this certificate>  To identify the CA public key that has to be used to verify this certificate.
	Extensions for Key Usage (RFC 5280 [25] section 4.2.1.3)	Extension for Key usage (RFC 5280 [25] section 4.2.1.3): extnID = id-ce-keyUsage critical = true extnValue = digitalSignature (0)
	Other extensions	CRL and OCSP SHOULD NOT be defined, since the eUICC will have no means to access those.  Policies SHOULD NOT be defined since they will increase the certificate size.  Subject Alternative Names SHOULD NOT be specified.
signatureAlgorithm	Section 2.6.4.2	
signatureValue	Signature computed accordingly to one of the possible algorithms listed in section 2.6.4.2	

**Table 4 eIM Certificate fields**

#### 2.6.4.4 eIM TLS/DTLS Certificate (CERT.EIM.TLS)

This specification supports X.509 certificate format as defined in RFC 5280 [25].

The table below describes the common fields that a CERT.EIM.TLS or CERT.EIM.DTLS SHALL contain.

NOTE: If the eIM supports both TLS and DTLS, it may use the same certificate.

Certificate common fields		
Field	Value Description	
tbsCertificate	Data to be signed	
	Field	Value Description
	version	Version SHALL be 3.
	serialNumber	Certificate serial number.
	signature	Contains the algorithm identifier used to compute the value of the field 'signatureValue'. NOTE: The algorithm identifier value SHALL be the same as the one of the field 'signatureAlgorithm'.
	issuer	Distinguished Name of the entity that has signed the certificate.
	validity	Validity period of the eIM TLS/DTLS Certificate.

	subject	Distinguished Name of the entity owning the certificate. The following DN components SHALL be used: CN=<eIMiD> -- mandatory C=<country code> -- mandatory O=<company> -- optional OU=<organizational unit> -- optional L=<locality> -- optional
	subjectPublicKeyInfo	Contains the algorithm identifier, parameters, and public key value. Algorithm identifier and parameters SHALL be set according to section 2.6.4.2 subjectPublicKeyInfo.subjectPublicKey contains the public key value and SHALL be coded as defined in RFC 5480 [27].
	Extension for Authority Key Identifier (RFC 5280 [25] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier critical = false extnValue = <Identifier of the public key to verify this certificate> To identify the CA public key that has to be used to verify this certificate.
	Extensions for Key Usage (RFC 5280 [25] section 4.2.1.3)	Extension for Key usage (RFC 5280 [25] section 4.2.1.3): extnID = id-ce-keyUsage critical = true extnValue = digitalSignature (0)
	Extensions for subjectAltName (RFC 5280 [25] section 4.2.1.6):	extnID = id-ce-subjectAltName critical = false extnValue = GeneralNames ::= { { dNSName '<eIM hostname (FQDN) value>' }, { registeredID '<eIM OID value>' } } } This extension SHALL contain at least one dNSName entry, and only one registeredID entry. A dNSName entry MAY contain the wildcard character * as allowed by RFC 2818 [31].

	Extension for extendedKeyUsage (RFC 5280 [25] section 4.2.1.12)	Extension for Extended Key usage (RFC 5280 [25] section 4.2.1.12): extnID = id-ce-extKeyUsage critical = true extnValue = ExtKeyUsageSyntax ::= { { id-kp-serverAuth }, { id-kp-clientAuth } }  Certificate SHALL not contain any other extended key usage.
	Other extensions	Extension for CRL Distribution Points (RFC 5280 [25] section 4.2.1.13).  Policies SHOULD NOT be defined since they will increase the certificate size.
signatureAlgorithm	Section 2.6.4.2	
signatureValue	Signature computed accordingly to one of the possible algorithms listed in section 2.6.4.2	

**Table 5 eIM TLS/DTLS Certificate fields**

**2.6.5 Use of Server Name indication extension (SNI) for ESipa.**

Server Name Indication, often abbreviated SNI, is an extension for TLS protocol. This extension allows multiple hostnames and consequently multiple TLS certificates to be served over HTTPS from the same IP address.

This capacity could be useful for the eIM server to select the right TLS certificate chain that IPA is able to verify.

This section provides to the IPA and the eIM Server a mechanism to guide the eIM Server selection of an appropriate TLS certificate chain.

The eIM Server and the IPA SHOULD support this mechanism. An eIM Server and an IPA that support this mechanism SHALL comply with this section.

**IPA/Device side:**

An IPA SHALL perform a DNS resolution using the eIM FQDN registered during the eIM association, then IPA SHALL proceed with the TLS handshake including the "server\_name" extension defined in RFC 6066 [35] in the ClientHello containing the eIM FQDN. If the eIM Server replies with a fatal-level alert different from unrecognized\_name(112), the IPA SHALL retry the TLS handshake without sending the Server Name indication extension.

NOTE: This section is also applicable to the IP Ae.

**eIM Server side:**

The eIM Server SHALL take care that its domain names in their final encoding do not violate the length limits of domain names.

The eIM Server SHALL support the "server\_name" extension to determine the TLS certificate to use.

- If the received ClientHello message contains the "server\_name" extension with an unrecognized FQDN, the eIM Server SHALL abort the handshake by sending a fatal-level unrecognized\_name(112) alert.
- If the received ClientHello message doesn't contain the "server\_name" extension, the eIM Server SHOULD use a TLS certificate associated with a default domain.

## 2.7 Certificate Management

### 2.7.1 RSP Server Certificates

The RSP Server Certificate management and verification SHALL follow what is specified in SGP.22 [4].

### 2.7.2 eUICC/EUM Certificates

The RSP Certificate revocation management as defined in SGP.22 [4] does not apply to the eUICC. Therefore, the revocation management is not supported by the eUICC and the IPA or eIM SHALL NOT pass any revocation information (CRL, OCSP) to the eUICC.

The eUICC Certificates (i.e., CERT.EUICC.ECDSA) according to SGP.22 [4] SHALL be used. A Field-Test eUICC MAY contain certificates that chain up to the GSMA CI for the purpose of integration and/or end-to-end testing, as defined in SGP.22 [4].

### 2.7.3 eIM Certificates

The following Certificates MAY be supported, and MAY be revoked at any time:

- eIM Certificate for signing eUICC Packages (if any)
- eIM DTLS/TLS Certificate (if any)

The issuance, verification and revocation of eIM Certificates for signing eUICC Packages are implementation specific and out of scope of this specification.

If an eIM utilizes DTLS/TLS for ESipa protection, the eIM SHALL use a Certificate or Public Key. The eIM DTLS/TLS Certificate (if any) or the CA that issues the eIM DTLS/TLS Certificate SHALL be trusted by the IPA. This CA MAY be public or private. In case of TLS, there SHALL be means for the IPAd to check the revocation status of this Certificate either through CRL or OCSP [21] made available by the issuing CA. The eIM MAY provide OCSP stapling. In case of DTLS, revocation status SHALL NOT be checked.

NOTE: For constrained IoT Devices, a private CA allows for flexibility in configuring how many sub CA layers are utilized or in defining the CA lifecycle.

A Certificate listed in a CRL SHALL be considered as definitively revoked (i.e., the 'Hold' state is not considered).

## 2.8 Profile Policy Management

Profile Policy Management provides mechanisms by which Profile Owners can enforce the conditions of use under which services are provided. Please refer to section 2.9 of SGP.22 [4] for details on this.

## 2.9 Profile State Management

Profile State Management Operation is related to the state update of a Profile in a dedicated ISD-P on the eUICC (e.g.: enable Profile, disable Profile, delete Profile, list Profile information).

### 2.9.1 Profile State Management by the eIM

Profile State Management is performed by an eIM. For this purpose, the eIM SHALL prepare a signed eUICC Package. For Profile State Management, an eUICC Package contains one or more PSMOs.

The eUICC Package is provided from the eIM to the IPA via ESipa. IPA transfers the eUICC Package to the eUICC via ES10b. The eUICC verifies the signature of the eUICC Package and, upon successful verification, executes the eUICC Package and generates a signed eUICC Package Result. The eUICC Package Result is provided to the IPA that provides it to the eIM. The eIM verifies the signature of the eUICC Package Result before processing the execution result of the eUICC Package.

All the procedure of Profile State Management by the eIM are described in section 3.4.

### 2.9.2 Immediate Profile enabling of an IPA initiated Profile download from default SM-DP+

The IPA MAY initiate a Profile download request towards the eUICC by instructing the use of the default SM-DP+. Subsequent to successful download and installation, the IPA MAY also request the eUICC to enable this newly downloaded Profile. The eUICC MAY be configured to support the immediate enabling:

- by EUMs during eUICC manufacturing, or
- by an Associated eIM via PSMO, or
- by the IPA via ES10b only if there is no Associated eIM.

If immediate enabling is not configured within the eUICC, the request for immediate Profile enabling SHALL be rejected.

An eUICC configured to support the immediate enabling and requested by IPA to immediately enable a particular Profile, SHALL verify that the OID and FQDN of the SM-DP+ from where the Profile is downloaded match to the default SM-DP+ data (OID and FQDN) stored in the eUICC (see section 3.2.3.1). The storage of the default SM-DP+ data MAY be performed according to the following options:

- by EUMs during eUICC manufacturing, or
- by an Associated eIM via PSMO, or
- by the IPA via ES10b only if there is no Associated eIM.

NOTE: SM-DS triggered Profile download and immediate enabling without signed PSMO from any potentially Associated eIM is not specified in this version of the specification.

## 2.10 eIM Configuration

In order to perform Profile State Management Operation on a given eUICC, an eIM SHALL be associated with this eUICC. For every Associated eIM the eUICC stores the eIM Configuration Data (see section 2.11.1.1.1) that is used by the eUICC for verification of signed PSMOs and Signed eIM Configuration Operations as part of eUICC Package(s).

eIM Configuration Operations MAY be performed at different stages within the eUICC or IoT Device life cycle as described in SGP.31 [1].

NOTE: Annex B of SGP.31 [1] describes the different eIM Configuration scenarios.

### 2.10.1 eIM Configuration by eIM

If an eUICC has an Associated eIM, only Signed eIM Configuration SHALL be accepted by the eUICC after successful verification of the signature of the Associated eIM. Any Unsigned eCO SHALL be rejected by the eUICC.

When an eUICC has one or more Associated eIMs, a signed eUICC Package SHALL be prepared by one of the Associated eIM to make changes to the eIM Configuration Data of the eUICC. The signed eUICC Package SHALL be transferred from the eIM to the IPAd via ESipa and further to the eUICC where it will be processed. The details are described in section 3.5.1.

The signed eUICC Package contains one or more eIM Configuration Operations. The following eIM Configuration Operations MAY be supported by the eIM and SHALL be supported by the eUICC:

- **addEim:** Adds an Associated eIM to the eUICC by providing its eIM Configuration Data including the `eimID` to the eUICC.
- **deleteEim:** Deletes an Associated eIM identified by its `eimID` from the eUICC. If the successfully deleted Associated eIM was the last available Associated eIM, the eUICC SHALL allow ES10b.AddInitialEim again.
- **updateEim:** Updates eIM Configuration Data, i.e., the public key or Certificate and the related anti-replay counter value of an Associated eIM with a given `eimID` within the eUICC while keeping the same `eimID`.
- **listEim:** Requests the eUICC to provide a list of all currently configured Associated eIMs to the eIM.

NOTE1: Replacing an existing Associated eIM with a new Associated eIM could be realized by first adding a new Associated eIM and then deleting the existing Associated eIM.

NOTE2: How the eIM triggers the IPA is out of scope of this specification.

### 2.10.2 eIM configuration by IPA

If an eUICC does not have an Associated eIM, the eUICC SHALL accept Unsigned eCO.

To enable Unsigned eCO, the IPAd MAY and the eUICC SHALL support the following function:

- ES10b.AddInitialEim: Adds eIM Configuration Data of a particular eIM with a given eimID to an eUICC that does not yet contain any Associated eIM.

NOTE: Mechanisms and functions used for Unsigned eCO at eUICC production (see Annex B.1 of SGP.31 [1]) are out of scope of this specification.

In addition, the eUICC and the IPAd MAY support ES10b.eUICCMemoryReset to completely remove all eIM Configuration Data of all Associated eIMs from the eUICC.

NOTE: The security of this mechanism is left to the implementation as described in SGP.31 [1].

## 2.11 eIM Package Structures

This section describes the structure of eIM Package requests and responses that comprise:

- Signed eUICC Package and its execution result.
- IPA and/or eUICC data
  - Profile Download trigger
  - List of Notifications
- Acknowledgements of eUICC Package Results and Notifications

### 2.11.1 eIM Package Request

The eIM Package request contains either one of the following data objects:

- an EuiccPackageRequest,
- an IpaEuiccDataRequest,
- a ProfileDownloadTriggerRequest, or
- an EimAcknowledgements

#### 2.11.1.1 EuiccPackageRequest

The signed eUICC Package structure SHALL be encoded in the ASN.1 data object as shown below.

```
-- ASN1START
EuiccPackageRequest ::= [81] SEQUENCE { -- Tag ' BF51', #SupportedForPsmoV1.0.0#
    euiccPackageSigned EuiccPackageSigned,
    eimSignature [APPLICATION 55] OCTET STRING -- Tag '5F37'
}

EuiccPackageSigned ::= SEQUENCE {
    eimId [0] UTF8String (SIZE(1..128)),
    eidValue [APPLICATION 26] Octet16, -- Tag '5A'
    counterValue [1] INTEGER,
    eimTransactionId [2] TransactionId OPTIONAL,
    euiccPackage EuiccPackage
}

EuiccPackage ::= CHOICE {
    psmoList SEQUENCE OF Psmo, -- #SupportedForPsmoV1.0.0#
    ecoList SEQUENCE OF Eco -- #SupportedForEcoV1.0.0#
}
```

-- ASN1STOP

The `euiccPackageSigned` includes:

- eIM identifier (`eimId`): identifier of the eIM that issues the eUICC Package. See section 4.3.
- EID (`eidValue`) of the targeted eUICC.
- Counter value (`counterValue`) used by the eIM for replay protection. Each eIM manages its own counter. The eIM increments its `counterValue` by 1 for each eUICC Package being sent to the target eUICC. The eUICC implementation SHALL be able to handle counter values at least up to value of 8388607 (0x7FFFFFFF), included.
- Optional Transaction ID (`eimTransactionId`, see [4]).
- eUICC Package (`euiccPackage`) consisting of either PSMOs or eCOs.

The eIM SHALL sign the `euiccPackageSigned` data object concatenated with the `associationToken` data object (retrieved from the eIM Configuration Data) to create the `eimSignature`. If no association token is configured in the eIM Configuration Data, the `associationToken` data object SHALL be used with the value set to zero (i.e., '84 01 00') for creating the signature.

The eUICC returns the same `eimId`, `eimTransactionId` and `counterValue` in the eUICC Package Result, which allows the eIM to link a received eUICC Package Result to a sent eUICC Package.

An eUICC MAY have a limitation in `EuiccPackageRequest` maximum size (see section 5.9.1).

#### 2.11.1.1.1 eIM Configuration Data

The eIM Configuration Data stored by the eUICC for each Associated eIM consists of:

- `eimId`: uniquely identifies the Associated eIM within the eUICC. See section 4.3.
- `eimFqdn`: Identifier of the eIM that issues the eUICC Package encoded as a FQDN.

NOTE: In case an intermediate server, e.g. MQTT Broker, is used for the communication between the eIM and the IoT Device, the identifier of that intermediate server is provided in `eimFqdn`.

- `eimIdType`: For more information regarding eIM identifier types (`EimIdType`) see section 4.3.
- `counterValue`: used to protect against replay of eUICC Packages sent by the eIM. The highest received `counterValue` per Associated eIM is stored within the eUICC. The eUICC implementation SHALL be able to handle counter values at least up to value of 8388607 (0x7FFFFFFF), included. The eUICC SHALL return the error code `counterValueOutOfRange` in case the `counterValue` exceeds the maximum value supported by the eUICC.

- `associationToken`: optional, for replay protection. The association token is the value that a global counter on each eUICC had when the token was requested. This counter SHALL start at zero and SHALL be incremented to generate the next association token each time an association token is requested (i.e., the first association token will be 1). Each generated association token SHALL be uniquely assigned to the eIM for which it is calculated and stored with that eIM's configuration data. It SHALL NOT be possible to reset the counter by any mechanism (e.g., eUICC Memory Reset).

NOTE 1: The association token can protect against an attack scenario where a whole sequence of operations beginning with `AddInitialEim` is replayed after a reset of the eIM Configuration Data. It is optional to configure this mechanism for an eIM.

NOTE 2: If an eIM is removed from the eUICC, its configuration data - including its association token - is deleted; if the same eIM is added again later, a new association token will be generated based on the current counter value.

- `eimPublicKeyData`: eIM public key data either as a raw public key (`eimPublicKey`) or as a Certificate (`eimCertificate`), used for eUICC Package verification.
- `trustedPublicKeyDataTls`: either eIM public key (`trustedEimPkTls`) or a Certificate (`trustedCertificateTls`) used for TLS/DTLS.  
The `trustedCertificateTls` is a certificate that is trusted in the chain of certificates ending with the eIM certificate (leaf certificate). This certificate MAY be the certificate of the eIM itself, or the certificate of the root CA or the certificate of a trusted intermediate CA. This certificate SHALL contain the `SubjectKeyIdentifier` extension. A root CA or intermediate CA certificate SHALL include the `BasicConstraints` extension with the value of "cA" set to TRUE [25]. The IPAd MAY ignore this field.
- `eimSupportedProtocol`: For more information regarding eIM protocol support (`EimSupportedProtocol`) see section 4.2.
- `euiccCiPKId`: CI Public Key Identifier allowing to select the eUICC Certificate for signing eUICC Package Results returned to this eIM. If not provided, the first entry of `euiccCiPKIdListForSigning` in `eUICCInfo2` SHALL be used as `euiccCiPKId`.
- `indirectProfileDownload`: if present it indicates the eIM support of Indirect Profile download with indicated `EimSupportedProtocol`.

NOTE: `trustedCertificateTls` data object is needed if the IoT Device uses the IP Ae and the eIM Package transfer use `eimRetrieveHttps` or `eimRetrieveCoaps`. This will allow to avoid any potential failure of the TLS/DTLS session.

When adding, updating, or retrieving eIM Configuration Data for a particular eIM from the eUICC the following ASN.1 data object is used, where the optionality of certain fields is described below:

```
-- ASN1START
EimConfigurationData ::= SEQUENCE {
```

```

eimId [0] UTF8String (SIZE(1..128)), -- eimId of eIM
eimFqdn [1] UTF8String OPTIONAL, -- FQDN of eIM or intermediate server, if used
eimIdType [2] EimIdType OPTIONAL,
counterValue [3] INTEGER OPTIONAL, -- initial counterValue for the eIM
associationToken [4] INTEGER OPTIONAL,
eimPublicKeyData [5] CHOICE {
    eimPublicKey SubjectPublicKeyInfo, -- public key of eIM, used for eUICC
    Package signature verification, where the encoding follows X.509 standard
    eimCertificate Certificate -- certificate of eIM, used for eUICC Package
    signature verification, where the encoding follows X.509 standard
} OPTIONAL,
trustedPublicKeyDataTls [6] CHOICE {
    trustedEimPkTls SubjectPublicKeyInfo, -- public key of eIM, used for TLS or
    DTLS, where the encoding follows X.509 standard
    trustedCertificateTls Certificate -- either the certificate of eIM, used for
    (D)TLS, or the certificate of the CA, where the encoding follows X.509 standard
} OPTIONAL,
eimSupportedProtocol [7] EimSupportedProtocol OPTIONAL,
euiccCiPKId [8] SubjectKeyIdentifier OPTIONAL, -- CI Public Key Identifier
supported on the eUICC for signature creation
indirectProfileDownload [9] NULL OPTIONAL, -- support of Indirect Profile
download with indicated EimSupportedProtocol
eSipaProprietaryProtocolInformation [10] VendorSpecificExtension OPTIONAL --
Additional information for proprietary protocol(s)
}
EimIdType ::= INTEGER {
    eimIdTypeOid(1),
    eimIdTypeFqdn(2),
    eimIdTypeProprietary(3)
}
EimSupportedProtocol ::= BIT STRING {
    eimRetrieveHttps(0),
    eimRetrieveCoaps(1),
    eimInjectHttps(2),
    eimInjectCoaps(3),
    eimProprietary(4)
}
-- ASN1STOP
    
```

### 2.11.1.1.2 eUICC Package Request Containing eCOs

The eCOs within the eUICC Package request SHALL be encoded in the ASN.1 data object as shown below.

```

-- ASN1START
Eco ::= CHOICE {
    addEim [8] EimConfigurationData, -- for eIM configuration data see 2.11.1
    deleteEim [9] SEQUENCE {eimId [0] UTF8String},
    updateEim [10] EimConfigurationData,
    listEim [11] SEQUENCE {}
}
-- ASN1STOP
    
```

If the eCO is the addition of an Associated eIM (addEim), then eimId, counterValue and eimPublicKeyData SHALL be provided as part of the EimConfigurationData data object.

The trustedPublicKeyDataTls is optional in EimConfigurationData data object. It SHOULD be provided in addEim if the IpaMode in eUICCInfo2 is ipae, and if the eimSupportedProtocol contains eimRetrieveHttps or eimRetrieveCoaps.

**NOTE :** IPAd of some IoT Devices may need to retrieve `trustedPublicKeyDataTls` from the eUICC by calling `GetEimConfigurationData`. In this case, `trustedPublicKeyDataTls` needs to be included in `EimConfigurationData` data of `addEim`.

The `eimFqdn` is optional for the eIM to provide in `EimConfigurationData` data object. It SHALL be provided if `eimIdType` is not `eimIdTypeFqdn` in the `EimConfigurationData` data object. It SHALL NOT be provided if `eimIdType` is `eimIdTypeFqdn` in that `EimConfigurationData` data object, to avoid redundancy with `eimId`.

The `eimIdType` is optional for the eIM to provide in `EimConfigurationData` data object. If not provided, the eUICC SHALL assign the value `eimIdTypeProprietary` before storing the eIM Configuration Data. If the eUICC is requested to calculate an association token for the new Associated eIM, the `associationToken` data object SHALL be provided with the value set to -1. Otherwise, the data object SHALL NOT be provided.

The `eimSupportedProtocol` is optional for the eIM to provide in `EimConfigurationData` data object. If not provided, the eUICC SHALL assign the value where only the `eimProprietary` bit is set before storing the eIM Configuration Data.

The `euiccCiPKId` is optional for the eIM to provide in `EimConfigurationData`. If not provided, the eUICC SHALL assign the value of first entry of `euiccCiPKIdListForSigning` in `eUICCInfo2` before storing the eIM Configuration Data.

If the eCO is the update of the configuration of an Associated eIM (`updateEim`), then `eimId` of the target Associated eIM and at least one of either `counterValue`, `eimPublicKeyData`, `eimIdType`, `eimFqdn`, `eimSupportedProtocol`, `trustedPublicKeyDataTls`, `euiccCiPKId` or `indirectProfileDownload` SHALL be provided as part of the `EimConfigurationData` data object. The `associationToken` data object SHALL NOT be provided.

`eimId` is used to retrieve the associated configuration. Updating `eimId` SHALL NOT be permitted. Updating `counterValue` to a lower value without also updating `eimPublicKeyData` SHALL NOT be permitted.

If the eCO is the deletion of an Associated eIM (`deleteEim`), then only the `eimId` of the target Associated eIM SHALL be provided.

### 2.11.1.1.3 eUICC Package Request Containing PSMOs

The PSMOs within the eUICC Package request SHALL be encoded in the ASN.1 data object as shown below.

```
-- ASN1START
Psmo ::= CHOICE {
  enable [3] SEQUENCE {
    iccid [APPLICATION 26] Iccid,
    rollbackFlag NULL OPTIONAL
  },
  disable [4] SEQUENCE {iccid [APPLICATION 26] Iccid},
  delete [5] SEQUENCE {iccid [APPLICATION 26] Iccid},
}
```

```
listProfileInfo [45] ProfileInfoListRequest, -- Tag 'BF2D'  
getRAT [6] SEQUENCE {},  
configureImmediateEnable [7] SEQUENCE {  
    immediateEnableFlag [0] NULL OPTIONAL,  
    defaultSmdpOid [1] OBJECT IDENTIFIER OPTIONAL,  
    defaultSmdpAddress [2] UTF8String OPTIONAL  
},  
setFallbackAttribute [8] SEQUENCE {iccid [APPLICATION 26] Iccid},  
unsetFallbackAttribute [9] SEQUENCE {},  
setDefaultDpAddress [101] SetDefaultDpAddressRequest -- Tag 'BF65'  
}  
-- ASN1STOP
```

The `iccid` indicates the target Profile at the eUICC as defined in SGP.22 [4].

The `rollbackFlag` is presented by the eIM within the `enable PSMO`. The IPA MAY trigger the Rollback Mechanism using the `ES10b.ProfileRollback` function if the newly enabled Profile fails to provide connectivity.

The `listProfileInfo` SHALL be coded and processed as defined in section 5.9.14.

If no tag list is present for `listProfileInfo`, the eUICC SHALL return by default the following list, if the tags are present in the Profile:

- ICCID, tag '5A'
- ISD-P AID, tag '4F'
- Profile state, tag '9F70'
- Service provider name, tag '91'
- Profile name, tag '92'
- Profile Class, tag '95'
- ecallIndication, tag '9F7B'
- fallbackAttribute, tag '9F26'
- fallbackAllowed, tag '9F67'

Immediate Profile enabling is activated when `immediateEnableFlag` is present in the `configureImmediateEnable` command and deactivated when `immediateEnableFlag` is not present. The configuration of default SM-DP+ data (`defaultSmdpOid` and `defaultSmdpAddress`) for use in the immediate Profile enabling is possible independently of whether the `immediateEnableFlag` is present or not.

The `setDefaultDpAddress` SHALL be coded and processed as defined in section 5.9.25.

At most one `enable` command SHALL occur in an eUICC Package.

At most one `disable` command SHALL occur in an eUICC Package.

A `listProfileInfo` command SHALL NOT occur after an `enable`, `disable` or `delete` command in an eUICC Package. If `listProfileInfo` command is encountered while a Profile stage change is ongoing, the response SHALL indicate the error `profileChangeOngoing`.

The `setFallbackAttribute` command is used to set the Fallback Attribute for the target Profile. The IPA MAY enable the Fallback Profile with the Fallback Mechanism using the

ES10b.ExecuteFallbackMechanism function (see section 5.9.20) if the IoT Device detects a permanent loss of connectivity using the currently enabled Profile. The IPA MAY disable the Fallback Profile using the ES10b.ReturnFromFallback function (see section 5.9.21).

**NOTE:** How the IoT Device detects a permanent loss of connectivity is implementation specific. How and upon which events the IPA is instructed by the IoT Device to execute an enable or disable of the Fallback Profile is IoT Device implementation specific.

The `unsetFallbackAttribute` command is used to unset the Fallback Attribute for the target Profile.

### 2.11.1.2 IpaEuiccDataRequest

The `IpaEuiccDataRequest` is used by the eIM to retrieve data from eUICC and/or IPA. It SHALL be encoded as follows:

```
-- ASN1START
IpaEuiccDataRequest ::= [82] SEQUENCE { -- Tag BF52
  tagList [APPLICATION 28] OCTET STRING, -- Tag '5C'
  euiccCiPKIdentifierToBeUsed OCTET STRING OPTIONAL, -- CI Public Key Identifier
  (possibly truncated) supported on the eUICC for signature creation
  searchCriteriaNotification [1] CHOICE {
    seqNumber [0] INTEGER,
    profileManagementOperation [1] NotificationEvent
  } OPTIONAL,
  searchCriteriaEuiccPackageResult [2] CHOICE {
    seqNumber [0] INTEGER
  } OPTIONAL,
  eimTransactionId [3] TransactionId OPTIONAL -- Tag '83'
}
-- ASN1STOP
```

The eIM MAY request one or more IPA and eUICC data items by listing each tag of a requested item in the `tagList` of `IpaEuiccDataRequest`. The following IPA and eUICC data and tags are defined:

- List of Notifications tag 'A0'.
- Default SM-DP+ address (see SGP.22 [4]), tag '81'.
- eUICC Package Results, tag 'A2'.
- eUICCInfo1 (see section 5.9.2 of this specification), tag 'BF20'.
- eUICCInfo2 (see section 5.9.2 of this specification), tag 'BF22'.
- Root SM-DS address (see SGP.22 [4]), tag '83'.
- Association token (see sections 2.11.1 and 5.9.18 of this specification), tag '84'.
- EUM certificate (see SGP.22 [4]), tag 'A5'.
- eUICC certificate (see SGP.22 [4]), tag 'A6'.
- IPA Capabilities (see section 4.1), tag 'A8'.
- Device Information (see SGP.22 [4] section 4.2), tag 'A9'.

Any tag that is present in the `tagList` of `IpaEuiccDataRequest`, but is not in the list of tags above, is ignored without error.

In order to select between multiple of EUM certificates and eUICC certificates within the eUICC the eIM MAY provide the CI Public Key Identifier (`euiccCiPKIdentifierToBeUsed`). If

present, the full CI Public Key Identifier (20 bytes) SHALL be sent to an IPA without IPA Capability `minimizeEsipaBytes` and the CI Public Key Identifier in truncated form SHALL be sent to an IPA with IPA Capability `minimizeEsipaBytes` (to minimize the number of transmitted bytes). Truncated form means only enough leading bytes of the CI Public Key Identifier are provided for IPA to be able to uniquely locate and extract the chosen CI Public Key Identifier from the `euiccCiPKIdListForSigning` of `eUICCInfo1` / `eUICCInfo2`. If `euiccCiPKIdentifierToBeUsed` is not provided, the first entry of `euiccCiPKIdListForSigning` in `eUICCInfo1` / `eUICCInfo2` SHALL be used as `euiccCiPKIdentifierToBeUsed` when selecting the EUM certificate and/or eUICC certificate to return.

The `searchCriteriaNotification` and `searchCriteriaEuiccPackageResult` data object can be used to filter the list of Notifications or eUICC Package Results that SHALL be returned by the IPA according to the following:

- If tag 'A0' is present in the `tagList`, then
  - if there is no `searchCriteriaNotification` the IPA SHALL return the `notificationsList` as part of the `IpaEuiccDataResponse` containing all Notifications stored in the eUICC.
  - if the `searchCriteriaNotification` is present and specifies `seqNumber`, and there is a Notification with the given sequence number stored in the eUICC, then the IPA SHALL return the `notificationsList` as part of the `IpaEuiccDataResponse` containing the Notification.
  - if the `searchCriteriaNotification` is present and specifies `profileManagementOperation` the IPA SHALL return the `notificationsList` as part of the `IpaEuiccDataResponse` containing all the Notifications of the types specified by `profileManagementOperation` (a bit set to 1 in the `profileManagementOperation` indicates that the eUICC SHALL return all the Notifications corresponding to this type).
- If tag 'A2' is present in the `tagList`, then
  - if there is no `searchCriteriaEuiccPackageResult` the IPA SHALL return the `euiccPackageResults` as part of the `IpaEuiccDataResponse` containing all eUICC Package Results stored in the eUICC.
  - if the `searchCriteriaEuiccPackageResult` is present and specifies `seqNumber`, and there is an eUICC Package Result with the given sequence number stored in the eUICC, then the IPA SHALL return the `euiccPackageResults` as part of the `IpaEuiccDataResponse` containing the eUICC Package Result.
- If tag 'A0' or tag 'A2' are not present, the IPA SHALL ignore `searchCriteriaNotification` and `searchCriteriaEuiccPackageResult` respectively.

Optionally, eIM MAY provide eIM transaction identifier (`eimTransactionId`) in order to correlate `IpaEuiccDataRequest` and `IpaEuiccDataResponse`.

IPA uses the following functions to obtain eUICC data: ES10b.GetEUICCInfo (see section 5.9.2), ES10b.GetCerts (see section 5.9.10), ES10b.GetEID (see section 5.9.19), ES10b.RetrieveNotificationsList (see section 5.9.11), ES10a.GetEuiccConfiguredAddresses (see section 5.8.1) and ES10b.GetEimConfigurationData (see section 5.9.18). IPA SHALL support providing at least the association token that belongs to the eIM making the request and the IPA Capabilities to the eIM. The rest of the IPA and eUICC data are not mandatory to support by IPA but depends on the IPA Capabilities. If a particular IPA or eUICC data is not supported by IPA it is simply ignored by IPA when building the `IpaEuiccDataResponse` structure (see section 2.11.2.2). Depending on IPA Capabilities the IPA SHALL support the following:

- An IPA having IPA Capability `eimCtxParams1Generation` SHALL support providing `deviceInfo` to the eIM.
- An IPA having IPA Capability `minimizeEsipaBytes` SHALL support providing `euiccInfo1`, `euiccInfo2`, `eumCertificate`, and `euiccCertificate` to the eIM.

### 2.11.1.3 ProfileDownloadTriggerRequest

The `ProfileDownloadTriggerRequest` SHALL be encoded as follows:

```
-- ASN1START
ProfileDownloadTriggerRequest ::= [84] SEQUENCE { -- Tag 'BF54'
  profileDownloadData [0] ProfileDownloadData OPTIONAL,
  eimTransactionId [2] TransactionId OPTIONAL
}
ProfileDownloadData ::= CHOICE {
  activationCode [0] UTF8String (SIZE(0..255)),
  contactDefaultSmdp [1] NULL,
  contactSmds [2] SEQUENCE {
    smdsAddress UTF8String OPTIONAL
  }
}
-- ASN1STOP
```

The eIM SHALL provide `ProfileDownloadTriggerRequest` data object as follows.

- If and only if the IPA indicates `eimDownloadDataHandling` in IPA Capabilities and the eIM requests the IPA to call `ESipa.InitiateAuthentication` function, an empty data object is provided (i.e., without `profileDownloadData`). The IPA continues with the indirect Profile download or Event Retrieval procedure.
- Otherwise `profileDownloadData` is provided, containing one of the following:
  - `activationCode`, if the eIM provides an Activation Code.
  - `contactDefaultSmdp`, if the eIM requests the IPA to contact the default SMDP+ address that is available in the IoT Device.
  - `contactSmds`, if the eIM requests the IPA to contact the SMDS address that is available in the IoT Device (`smdsAddress` is absent) or specified by the eIM (`smdsAddress` is present). This triggers the retrieval of the SM-DP+ Address and EventID from the SM-DS, either over ES11 as defined in section 3.6.2 of SGP.22 [4] or over ESipa and ES11' as defined in section 3.9.2.2 of this document.

For each use case the IPA continues with the corresponding direct/indirect Profile download or Event Retrieval procedure depending on its capability.

The eIM transaction identifier (`eimTransactionId`) is OPTIONAL for the eIM to include in the `ProfileDownloadTriggerRequest`. It MAY be used for linking a `ProfileDownloadTriggerRequest` to a `ProfileInstallationResult` in a Direct Profile Download procedure, or determine the IoT Device in Indirect Profile Download procedure.

#### 2.11.1.4 EimAcknowledgements

The `EimAcknowledgements` structure is sent from the eIM to the IPA to inform the IPA that the eIM successfully received Notifications and/or eUICC Package Results sent from the IPA to the eIM. The structure contains the sequence number(s) of received Notifications and/or eUICC Package Results and is defined as follows:

```
-- ASN1START
EimAcknowledgements ::= [83] SEQUENCE OF SequenceNumber -- Tag BF53
SequenceNumber ::= [0] INTEGER
-- ASN1STOP
```

#### 2.11.2 eIM Package Result

The eIM Package Result contains either one of the following data objects:

- an `EuiccPackageResult`,
- an `EuiccPackageResult` concatenated with a `PendingNotificationList` (see section 2.11.2.2),
- an `IpaEuiccDataResponse`,
- a `ProfileDownloadTriggerResult`, or
- an `EimPackageResultResponseError`

##### 2.11.2.1 EuiccPackageResult

Before processing the PSMOs or the eCOs in the eUICC Package, the eUICC verifies the signed eUICC Package as defined in sections 3.3 and 5.9.1.

Upon successful verification of the signed eUICC Package, the PSMOs or eCOs in the eUICC Package are executed sequentially by the eUICC until the end is reached or an error is encountered for a PSMO or an eCO. Upon finishing the execution of the signed eUICC Package, the eUICC produces a signed eUICC Package Result (`euiccPackageResultSigned`) containing the PSMO or eCO execution result(s).

A signed eUICC Package Result with PSMO or eCO processing result(s) (`euiccPackageResultSigned`) SHALL be kept by the eUICC (which can hold one or several such signed eUICC Package Result) until explicitly deleted by the IPA, after successful delivery to the eIM. Before being deleted, the signed eUICC Package Result(s) MAY be retrieved at any time by the IPA. The unsigned and signed eUICC Package error

results generated from eUICC Package verification errors (`euiccPackageErrorUnsigned` and `euiccPackageErrorSigned`) are not kept by the eUICC.

When the eUICC needs to store a new signed eUICC Package Result with PSMO or eCO processing result(s) and if there is not enough space, the eUICC SHALL delete one or more of the previously stored signed eUICC Package Results and Pending Notifications in order of their sequence number, beginning with the lowest. If deleting all signed eUICC Package Results and Pending Notifications is still not enough to store a new signed eUICC Package Result, the eUICC SHALL return the signed eUICC Package Result to the IPA without storing it.

The eUICC Package Result SHALL be encoded in the ASN.1 data object as shown below.

```
-- ASN1START
EuiccPackageResult ::= [81] CHOICE { -- Tag 'BF51' #SupportedForPsmoV1.0.0#
    euiccPackageResultSigned EuiccPackageResultSigned,
    euiccPackageErrorSigned EuiccPackageErrorSigned,
    euiccPackageErrorUnsigned EuiccPackageErrorUnsigned
}
EuiccPackageResultSigned ::= SEQUENCE {
    euiccPackageResultDataSigned EuiccPackageResultDataSigned,
    euiccSignEPR [APPLICATION 55] OCTET STRING -- Tag '5F37'
}
EuiccPackageResultDataSigned ::= SEQUENCE { -- #SupportedForPsmoV1.0.0#
    eimId [0] UTF8String (SIZE(1..128)),
    counterValue [1] INTEGER,
    eimTransactionId[2] TransactionId OPTIONAL,
    seqNumber [3] INTEGER,
    euiccResult SEQUENCE OF EuiccResultData
}
EuiccResultData ::= CHOICE {
    enableResult [3] EnableProfileResult,
    disableResult [4] DisableProfileResult,
    deleteResult [5] DeleteProfileResult,
    listProfileInfoResult [45] ProfileInfoListResponse,
    getRATResult [6] RulesAuthorisationTable, -- see SGP.22
    configureImmediateEnableResult [7] ConfigureImmediateEnableResult,
    addEimResult [8] AddEimResult,
    deleteEimResult [9] DeleteEimResult,
    updateEimResult [10] UpdateEimResult,
    listEimResult [11] ListEimResult,
    rollbackResult [12] RollbackProfileResult,
    setFallbackAttributeResult [13] SetFallbackAttributeResult,
    unsetFallbackAttributeResult [14] UnsetFallbackAttributeResult,
    processingTerminated INTEGER {
        resultSizeOverflow(1),
        unknownOrDamagedCommand(2),
        interruption(3),
        undefinedError(127)
    },
    setDefaultDpAddressResult [101] SetDefaultDpAddressResponse
}
EuiccPackageErrorSigned ::= SEQUENCE {
    euiccPackageErrorDataSigned EuiccPackageErrorDataSigned,
    euiccSignEPE [APPLICATION 55] OCTET STRING -- Tag '5F37'
}
EuiccPackageErrorDataSigned ::= SEQUENCE {
    eimId [0] UTF8String (SIZE(1..128)),
    counterValue [1] INTEGER,
    eimTransactionId [2] TransactionId OPTIONAL,
    euiccPackageErrorCode EuiccPackageErrorCode
}
EuiccPackageErrorCode ::= INTEGER {
    invalidEid(3),
```

```
    replayError(4),
    counterValueOutOfRange(6),
    sizeOverflow(15),
    ecallActive(104),
    undefinedError(127) }

EuiccPackageUnsignedErrorCode ::= INTEGER {
    sizeOverflow(15),
    undefinedError(127)
}

EuiccPackageErrorUnsigned ::= SEQUENCE {
    eimId [0] UTF8String (SIZE(1..128)),
    eimTransactionId [2] TransactionId OPTIONAL,
    associationToken [4] INTEGER OPTIONAL,
    euiccPackageUnsignedErrorCode [15] EuiccPackageUnsignedErrorCode OPTIONAL
}

ConfigureImmediateEnableResult ::= INTEGER {
    ok(0),
    insufficientMemory(1),
    commandError(7),
    undefinedError(127)
}

EnableProfileResult ::= INTEGER {
    ok(0),
    iccidOrAidNotFound(1),
    profileNotInDisabledState(2),
    disallowedByPolicy(3),
    catBusy(5),
    rollbackNotAvailable(20),
    undefinedError(127)
}

DisableProfileResult ::= INTEGER {
    ok(0),
    iccidOrAidNotFound(1),
    profileNotInEnabledState(2),
    disallowedByPolicy(3),
    catBusy(5),
    undefinedError(127)
}

DeleteProfileResult ::= INTEGER {
    ok(0),
    iccidOrAidNotFound(1),
    profileNotInDisabledState(2),
    disallowedByPolicy(3),
    rollbackNotAvailable(20),
    returnFallbackProfile(21),
    undefinedError(127)
}

ProfileInfoListResponse ::= [45] CHOICE {
    profileInfoListOk SEQUENCE OF ProfileInfo, -- see SGP.22
    profileInfoListError ProfileInfoListError
}

ProfileInfoListError ::= INTEGER {
    incorrectInputValues(1),
    profileChangeOngoing(11),
    undefinedError(127)
}

RollbackProfileResult ::= INTEGER {
    ok(0),
    undefinedError(127)
}

SetFallbackAttributeResult ::= INTEGER {
    ok(0),
    iccidOrAidNotFound(1),
    fallbackNotAllowed(2),
    fallbackProfileEnabled(3),
```

```

        undefinedError (127)
    }
}
UnsetFallbackAttributeResult ::= INTEGER {
    ok(0),
    noFallbackAttribute(2),
    fallbackProfileEnabled (3),
    commandError(7),
    undefinedError (127)
}
}
AddEimResult ::= CHOICE {
    associationToken [4] INTEGER,
    addEimResultCode INTEGER {
        ok(0),
        insufficientMemory(1),
        associatedEimAlreadyExists(2),
        ciPKUnknown(3),
        invalidAssociationToken(5),
        counterValueOutOfRange(6),
        commandError(7),
        undefinedError(127)
    }
}
}
DeleteEimResult ::= INTEGER {
    ok(0),
    eimNotFound(1),
    lastEimDeleted(2), -- no eIM Configuration Data available in eUICC,
    commandError(7),
    undefinedError(127)
}
}
UpdateEimResult ::= INTEGER {
    ok(0),
    eimNotFound (1),
    ciPKUnknown(3),
    counterValueOutOfRange(6),
    commandError(7),
    undefinedError(127)
}
}
ListEimResult ::= CHOICE {
    eimIdList SEQUENCE OF EimIdInfo,
    listEimError INTEGER {
        undefinedError(127)
    }
}
}
EimIdInfo ::= SEQUENCE {
    eimId [0] UTF8String (SIZE(1..128)),
    eimIdType [2] EimIdType OPTIONAL -- present in case of eimIdTypeOid and
    eimIdTypeFqdn
}
}
-- ASN1STOP

```

The eIM transaction identifier (`eimTransactionId`) is OPTIONAL to use by the eIM in the signed eUICC Package. If it was included in the signed eUICC Package, the eUICC SHALL include the same `eimTransactionId` in the signed eUICC Package Result and also `euiccPackageErrorUnsigned` and `euiccPackageErrorSigned` in case of errors.

`euiccSignEPR` and `euiccSignEPE` SHALL be created using the SK.EUICC.ECDSA and verified using the related PK.EUICC.ECDSA. In case there are more than one SK.EUICC.ECDSA available for signing, the SK.EUICC.ECDSA chaining back to the CI public key identified by the `euiccCiPKId` of the eIM Configuration Data (of the Associated eIM for which the eUICC Package Result is intended) SHALL be used. `euiccSignEPR` SHALL apply on the concatenated data objects `euiccPackageResultDataSigned` and

V1.3 Page 42 of 231

associationToken. euiccSignEPE SHALL apply on the concatenated data objects euiccPackageErrorDataSigned and associationToken.

If no associationToken is configured for the eIM, the associationToken data object SHALL be used with the value set to zero for creating the signature.

The eIM SHALL use the PK.EUICC.ECDSA from CERT.EUICC.ECDSA to verify the signature euiccSignEPR and euiccSignEPE. The eIM SHALL verify CERT.EUICC.ECDSA.

When the Emergency Profile is enabled, the eUICC SHALL reject any EuiccPackageRequest, EuiccMemoryResetRequest, or ExecuteFallbackMechanism with the error codes ecallActive.

### 2.11.2.2 IpaEuiccDataResponse

The IpaEuiccDataResponse SHALL be encoded as follows:

```
-- ASN1START
IpaEuiccDataErrorCode ::= INTEGER {
    incorrectTagList (1),
    euiccCiPKIdNotFound(5),
    ecallActive(104),
    undefinedError(127)
}

IpaEuiccDataResponseError ::= SEQUENCE {
    eimTransactionId [0] TransactionId OPTIONAL,
    ipaEuiccDataErrorCode IpaEuiccDataErrorCode
}

IpaEuiccDataResponse ::= [82] CHOICE { -- Tag 'BF52'
    ipaEuiccData IpaEuiccData,
    ipaEuiccDataResponseError IpaEuiccDataResponseError
}

PendingNotificationList ::= SEQUENCE OF PendingNotification
EuiccPackageResultList ::= SEQUENCE OF EuiccPackageResult

IpaEuiccData ::= SEQUENCE {
    notificationsList [0] PendingNotificationList OPTIONAL, -- Tag 'A0'
    defaultSmdpAddress [1] UTF8String OPTIONAL, -- Tag '81'
    euiccPackageResultList [2] EuiccPackageResultList OPTIONAL, -- Tag 'A2'
    euiccInfo1 [32] EUICCInfo1 OPTIONAL, -- Tag 'BF20'
    euiccInfo2 [34] EUICCInfo2 OPTIONAL, -- Tag 'BF22'
    rootSmdsAddress [3] UTF8String OPTIONAL, -- Tag '83'
    associationToken [4] INTEGER OPTIONAL, -- Tag '84'
    eumCertificate [5] Certificate OPTIONAL, -- Tag 'A5'
    euiccCertificate [6] Certificate OPTIONAL, -- Tag 'A6'
    eimTransactionId [7] TransactionId OPTIONAL, -- Tag '87'
    ipaCapabilities [8] IpaCapabilities OPTIONAL, -- Tag 'A8'
    deviceInfo [9] DeviceInfo OPTIONAL -- Tag 'A9'
}
-- ASN1STOP
```

If the associationToken is requested, but no association token is configured for the eIM, the associationToken data object SHALL be absent in the response.

An IPA with IPA Capability minimizeEsipaBytes SHOULD include each pending Notification in notificationsList in compact format as described in section 5.14.7.

If `eimTransactionId` was present in the `IpaEuiccDataRequest`, the IPA SHALL return the same `eimTransactionId` in the `IpaEuiccData` or `IpaEuiccDataResponseError`.

If the Emergency Profile is enabled, then the IPA SHALL stop the procedure and return an `IpaEuiccDataResponseError` containing `ipaEuiccDataErrorCode` set to `ecallActive`.

### 2.11.2.3 ProfileDownloadTriggerResult

The `ProfileDownloadTriggerResult` SHALL be encoded as follows:

```
-- ASN1START
ProfileDownloadTriggerResult ::= [84] SEQUENCE { -- tag 'BF54'
  eimTransactionId [2] TransactionId OPTIONAL,
  profileDownloadTriggerResultData CHOICE {
    profileInstallationResult [55] ProfileInstallationResult, -- see SGP.22 [4]
    profileDownloadError SEQUENCE {
      profileDownloadErrorReason [0] INTEGER {
        ecallActive (104),
        undefinedError(127)
      },
      errorResponse OCTET STRING OPTIONAL
    }
  }
}
-- ASN1STOP
```

The IPA SHALL send `ProfileDownloadTriggerResult` to inform the eIM about the result of a direct Profile download that was requested by the eIM.

- If the eIM provided a transaction identifier (`eimTransactionId`) in `ProfileDownloadTriggerRequest`, the IPA SHALL return the same transaction identifier in `ProfileDownloadTriggerResult`. Otherwise, `eimTransactionId` SHALL NOT be present.
- If the Emergency Profile is enabled, then the IPA SHALL stop the procedure and return a `profileDownloadError` containing `profileDownloadErrorReason` set to `ecallActive`.
- If the IPA successfully downloaded a Profile and obtained a Profile Installation Result as a response of `ES10b.LoadBoundProfilePackage` function calls, the IPA SHALL return the Profile Installation Result in `profileInstallationResult`. Otherwise, the IPA SHALL return a `profileDownloadError` that MAY include an `errorResponse`. The `errorResponse`, if present, SHALL include either:
  - error response of an ES10b function or an ES9+ function with ASN.1 binding that the IPA encountered during the Profile download procedure; or
  - contain the `CancelSessionRequest` information element for cancellation as defined in section 5.7.14 of SGP.22 [4] for `ES10b.CancelSession` (excepted `endUserRejection` for `CancelSessionReason` which is not applicable in this specification), if the download is not successful because of an explicit IPA cancellation (e.g. using `ES10b.CancelSession`).

NOTE: the `errorResponse` for an error response of an ES9+ function with JSON binding is FFS.

## 3 Procedures

This section specifies the procedures associated with eUICC for IoT Devices.

Some call flows illustrate the case where the IPA is in the Device (IPAd). Such call flows with an IPAd would be identical except that all ES10 calls become internal to the eUICC and are out of scope of this specification.

### 3.1 eIM and IPA Communication

This section defines the procedures between the eIM and the IPA.

#### 3.1.1 Transferring eIM Package

These procedures are used to transfer an eIM Package from the eIM to the IPA. It could be the IPA which retrieves any pending command from the eIM (section 3.1.1.1 eIM Package Retrieval), or the eIM which directly sends the command to the IPA (section 3.1.1.2 eIM Package Injection). The eIM SHALL support at least, one of these procedures. The IPA implementing ESipa SHALL support, at least, one of these procedures.

If ESipa is established between the eIM and the IPA, the eIM and IPA SHALL support the same mode (“eIM Package retrieval” or “eIM Package injection”) and use the same secure connection method (see section 3.1.2).

NOTE: An eIM intended to serve many different types of IoT Devices is recommended to support both procedures (“eIM Package Retrieval” and “eIM Package Injection”).

##### 3.1.1.1 eIM Package Retrieval

This procedure is used by the IPA to retrieve any pending eIM Package from the eIM.

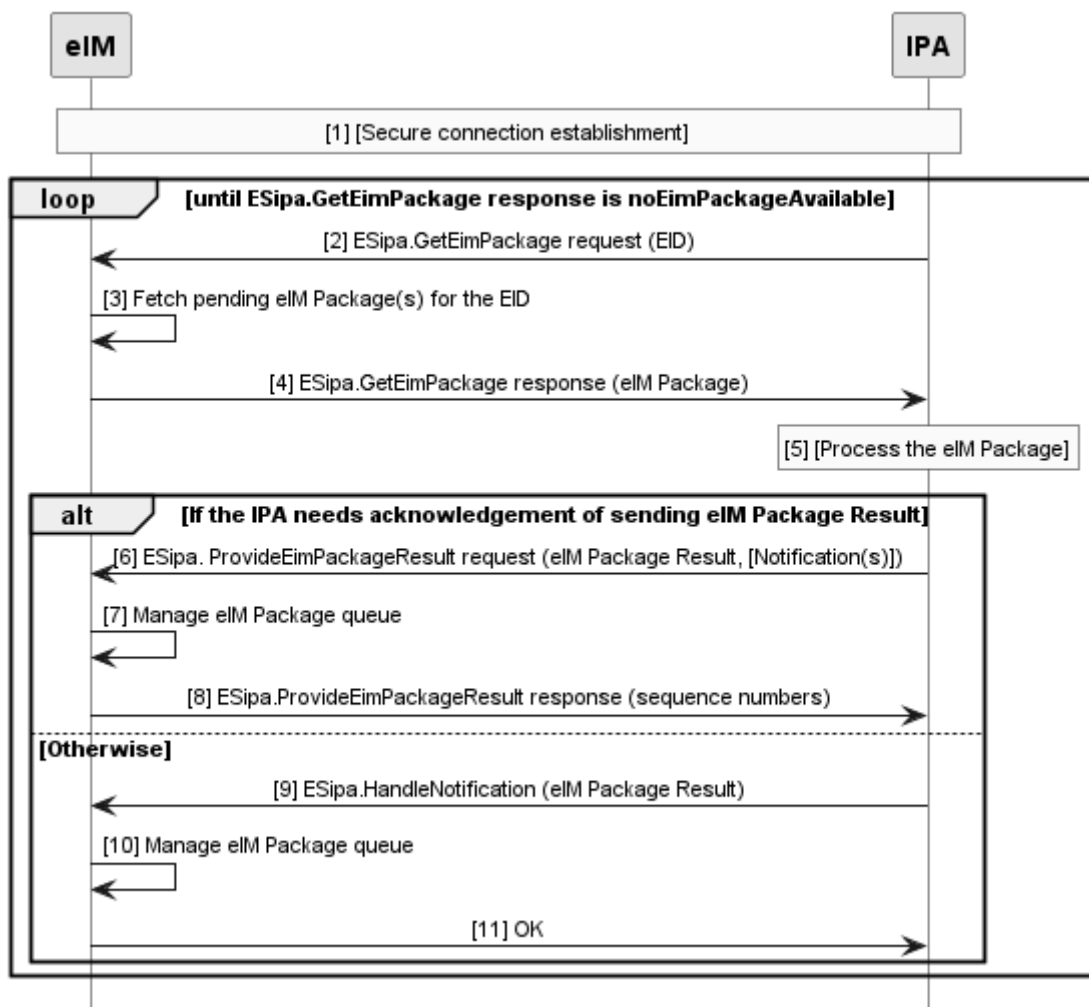
```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>eIM" as EIM
participant "<b>IPA" as IPA

rnote over EIM, IPA : [1] [Secure connection establishment]

loop until ESipa.GetEimPackage response is noEimPackageAvailable
IPA -> EIM : [2] ESipa.GetEimPackage request (EID)
EIM -> EIM : [3] Fetch pending eIM Package(s) for the EID
EIM -> IPA : [4] ESipa.GetEimPackage response (eIM Package)
rnote over IPA : [5] [Process the eIM Package]
alt If the IPA needs acknowledgement of sending eIM Package Result
IPA -> EIM : [6] ESipa.ProvideEimPackageResult request (eIM Package Result,
[Notification(s)])
EIM -> EIM : [7] Manage eIM Package queue
EIM -> IPA : [8] ESipa.ProvideEimPackageResult response (sequence numbers)
else Otherwise
IPA -> EIM : [9] ESipa.HandleNotification (eIM Package Result)
EIM -> EIM : [10] Manage eIM Package queue
EIM -> IPA : [11] OK
end
end
```

@endum1



**Figure 5 eIM Package Retrieval**

**Start Conditions:**

The IoT Device can trigger the IPA through an internal event (e.g., a timer expiration) without any knowledge about pending eIM Package. The IPA could also be informed, via a push Notification, about pending eIM Package in the eIM. For instance, the eIM or a backend system of the IoT Device maker can notify the IPA via out-of-band channel about pending eIM Package. The push Notification mechanism is out of scope of this document.

The IPA has the address (configured or retrieved using GetEimConfigurationData) of the Associated eIM. The IPA MAY have retrieved from the eUICC the Connectivity Parameters of the currently enabled Profile. The IPAd MAY use ES10b.GetConnectivityParameters. How the IPAd retrieves Connectivity Parameters is implementation specific.

**Procedure:**

1. The eIM and IPA establish a secure connection as defined in section 3.1.2.
2. The IPA calls ESipa.GetEimPackage function including EID.

3. The eIM fetches pending eIM Package(s) for the EID.
4. The eIM returns the eIM Package to the IPA.
5. The IPA processes the eIM Package.

If the IPA sends eIM Package Result together with Notifications, or if the IPA needs acknowledgements of sending eIM Package Result and optionally Notifications, the procedure continues in step (6). Otherwise, the procedure continues in step (9).

6. The IPA calls ESipa.ProvideEimPackageResult function including the eIM Package Result and optionally one or more Notifications.
7. The eIM processes the eIM Package Result and manages the eIM Packages pending in the queue for the eUICC.
8. The eIM acknowledges the reception of the eIM Package Result and optionally Notifications by returning the corresponding sequence numbers.
9. The IPA calls ESipa.HandleNotification function comprising either the eIM Package Result or a Notification.
10. The eIM processes the eIM Package Result and manages the eIM Packages pending in the queue for the eUICC.
11. The eIM returns OK.

The IPA MAY repeat the above procedure until the eIM response is noEimPackageAvailable.

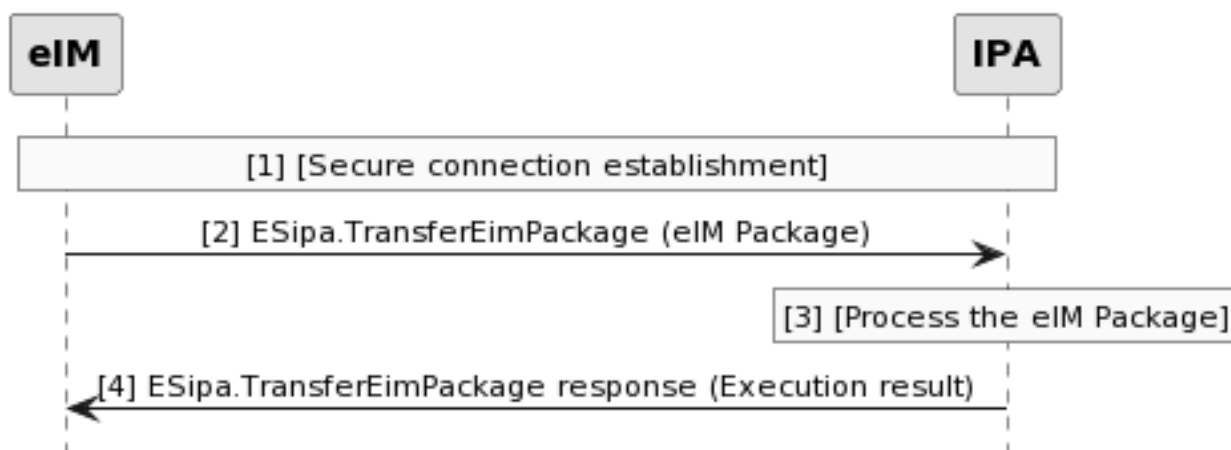
### 3.1.1.2 eIM Package Injection

This procedure is used by the eIM to inject eIM Package to the IPA.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>eIM" as EIM
participant "<b>IPA" as IPA

rnote over EIM, IPA : [1] [Secure connection establishment]
EIM -> IPA : [2] ESipa.TransferEimPackage (eIM Package)
rnote over IPA : [3] [Process the eIM Package]
IPA -> EIM : [4] ESipa.TransferEimPackage response (Execution result)
@enduml
```



**Figure 6 eIM Package Injection**

**Start Conditions:**

The eIM and the IPA could be part of a whole system responsible of maintaining the communication between the eIM and the IPA.

**Procedure:**

1. The eIM and IPA establish a secure connection as defined in section 3.1.2.

NOTE: the hosting system (e.g.: device management system) allows the eIM to retrieve the EID of the IoT Device.

2. The eIM calls ESipa.TransferEimPackage function to the IPA comprising the eIM Package.

NOTE: It is implementation specific how this package is transferred from the eIM to the IPA using the underlying transport protocol. See examples in Annex B.

3. The IPA processes the eIM Package.
4. The IPA returns the eIM Package Result to the eIM.

**3.1.2 Secure Connection Establishment**

The present document does not mandate any protocol between the eIM and the IPA for conveying an eIM Package. However, the protocol SHALL provide at least the integrity and the confidentiality of messages.

This permits IoT Device deployments using any suitable protocol for the exchange of eIM Packages and the delivery of Bound Profile Packages.

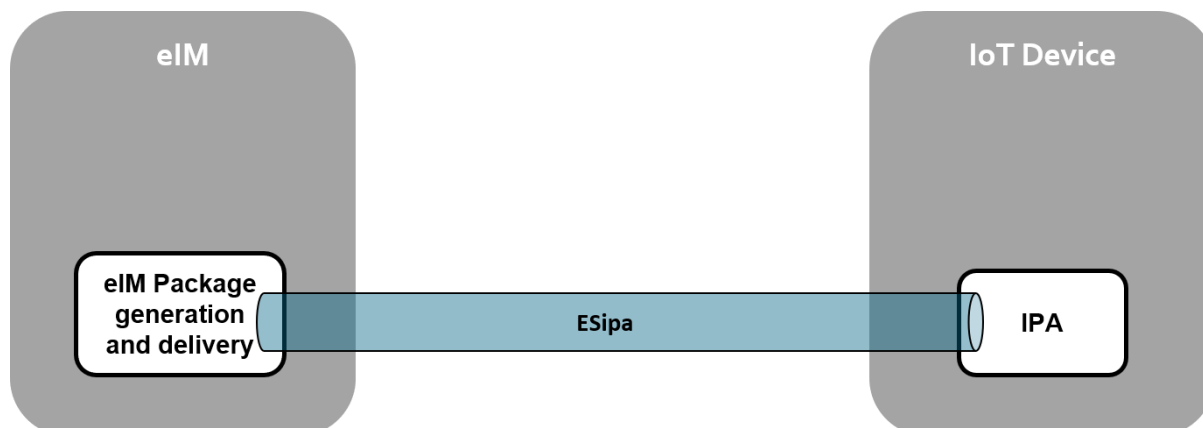
If ESipa is providing the protocol to transport ESipa messages as shown in Figure 7, the following ones are suggested:

- HTTP over TCP with TLS security.

NOTE: This could be used by a UICD which is not an NCD.

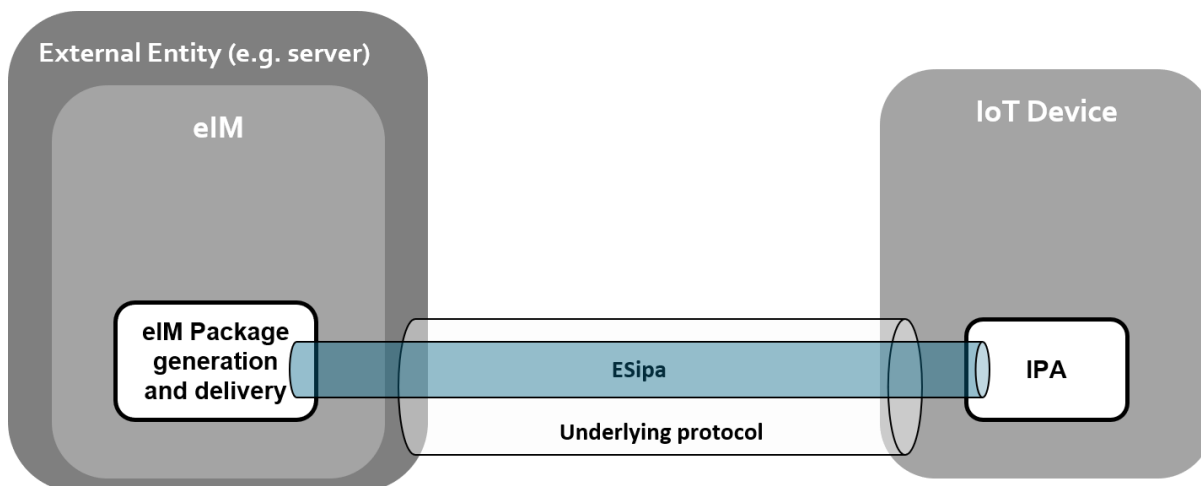
- CoAP over UDP with DTLS security.

NOTE: This mode could be used by an NCD.



**Figure 7 ESipa (eIM -- IPA)**

If the IoT Device has already implemented an interface and protocol with an external entity (e.g., a server) that is used for other functions of the IoT Device as shown in Figure 8, this underlying interface and its protocol can be used to transport ESipa messages.



**Figure 8 ESipa (eIM -- IPA), using underlying protocol to transport ESipa messages**

### 3.1.2.1 Secure Connection using HTTP over TCP

If HTTP/TLS (HTTPS) is used between the IPA and the eIM, the specification details in this section SHALL apply.

The HTTP/TLS (HTTPS) secure connection mode is used with server authentication. This means that only the eIM is authenticated by the IPA. If HTTPS is used, the TLS requirements defined in section 2.6.3.2 SHALL apply to the TLS session between IPA and eIM.

The ESipa functions requests and functions responses are sent over HTTPS. The interface binding over HTTP SHALL follow section 6.

NOTE: It is RECOMMENDED to use HTTP/TLS wherever TCP is available and working.

### **3.1.2.2 Secure Connection over CoAP**

If CoAP with DTLS security is used between the IPA and the eIM, the specification details in this section SHALL apply.

The DTLS secure connection mode is used with server authentication. This means that only the eIM is authenticated by the IPA.

If CoAP is used, the DTLS requirements defined in section 2.6.3.2 SHALL apply to the DTLS session between IPA and eIM.

The ESipa functions requests and functions responses SHALL be sent over CoAP. The interface binding over CoAP SHALL follow section 6.

If DTLS 1.2 is used, IPA and eIM SHOULD use Connection ID (CID) as per RFC 9146 [14] to maintain the current DTLS association for better reliability.

If DTLS 1.2 is used, IPA SHOULD implement both and eIM SHOULD implement at least one of the following session resumption methods:

- The Session Identifiers as defined in RFC 5246 [8], or
- The Session Tickets as defined in RFC 5077 [20].

### **3.1.2.3 Secure Connection by underlying transport layer**

This mode is used when the underlying transport layer supported by the IoT Device can transport ESipa messages.

ESipa messages SHALL be protected in terms of confidentiality, integrity, and authenticity.

This mode is not specified by this document. It's out of scope how the ESipa messages are conveyed. Hence, the compatibility of the exchanges between the IPA and eIM is ultimately the responsibility of the integrator.

Annex B gives some deployments scenario using the underlying transport layer.

## **3.2 Profile Download and Installation**

### **3.2.1 Profile Download Initiation**

This procedure is identical to the Profile Download Initiation Procedure defined in section 3.1.1 of SGP.22 [4].

### **3.2.2 Common Mutual Authentication**

This procedure is identical to Common Mutual Authentication Procedure defined in section 3.1.2 of SGP.22 [4]. It defines the mutual authentication procedure between the eUICC and the RSP Server.

### 3.2.3 Profile Download

#### 3.2.3.1 Direct Profile Download

This section describes the Profile download and installation procedure where the IoT Device directly connects to the SM-DP+.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid
participant "<b>Operator" as OP
participant "<b>SM-DP+" as DP
participant "<b>eIM" as EIM
participant "<b>IPA" as IPA
participant "<b>eUICC" as E

group If option (a) and AC is available to eIM
  rnote over EIM : [1] eIM parses the AC to retrieve SM-DP+ Address, Activation Code Token, [SM-DP+ OID]
end

group If IPA obtains a profile download trigger from eIM Package
  alt IPA-initiated
    rnote over EIM, IPA : [2a] eIM Package Retrieval Procedure including secure connection establishment (section 3.1.1.1)
  else eIM-initiated
    rnote over EIM, IPA : [2b] eIM Package Injection Procedure including secure connection establishment (section 3.1.1.2)
  end
end

rnote over IPA : [3] IPA parses the eIM Package and obtains profile \ndownload trigger that contain AC, or SM-DP+ Address and EventID \nor instruction to use the default SM-DP+ Address
end

group If option (a) and AC is available to IPA
  rnote over IPA : [4] IPA parses AC to retrieve SM-DP+ Address, Activation Code Token, [SM-DP+ OID]
end

rnote over DP, E #FFFFFF
[5]
- Execution of common mutual authentication procedure as defined in section 3.1.2 of SGP.22 [4].
- If immediate Profile enabling is activated, the eUICC SHALL verify that the download is coming from the default SM-DP+ configured in the eUICC.
  If the verification is successful the eUICC SHALL grant immediate enabling for this Profile.
endnote

rnote over DP #FFFFFF
[6]
- Look for Profile download pending order
- Eligibility Check using Device Info, euiccInfo2
endnote

Group Opt.
DP -> OP : [7] ES2+.HandleDownloadProgressInfo(...)
OP --> DP : OK
end
DP --> IPA : [error]
```

```
rnote over DP #FFFFFF
[8]
- Build Profile Metadata
- Check if download retry
- Build smdpSigned2 = {TransactionID,
  Confirmation Code Required Flag, [bppEuiccOtpk]}
- Compute smdpSignature2 over smdpSigned2 and euiccSignature1
endrnote

DP -> IPA : [9] TransactionID, Profile Metadata, smdpSigned2, smdpSignature2,
CERT.DPpb.ECDSA

group [10] Check if ProfileMetadata \ncontains PPR(s)
rnote over IPA #FFFFFF
endrnote
IPA -> E : [ES10b.GetRAT]
E --> IPA : [RAT]
IPA -> E : [ES10b.GetProfilesInfo]
E --> IPA : [ProfileInfoListOk]
end

rnote over DP, E #FFFFFF
[11] sub-procedure Profile Download and Installation - Download Confirmation as
defined in section 3.1.3.2 of SGP.22 [4]
endrnote

rnote over IPA, E #FFFFFF
[12] IPA installs the Profile to the eUICC as defined in sub-procedure
Profile Installation in section 3.1.3.3 of SGP.22 [4] steps 1 to 6
endrnote

IPA -> EIM : [13] ["ESipa.HandleNotification" function]

group [14].
IPA -> DP : "ES9+.HandleNotification" function

rnote over DP
SM-DP+ SHALL continue the procedure as defined
  in section 3.1.3.3 of SGP.22 [4] step 8 to 10
endrnote
end

group opt
rnote over IPA #FFFFFF
[15]

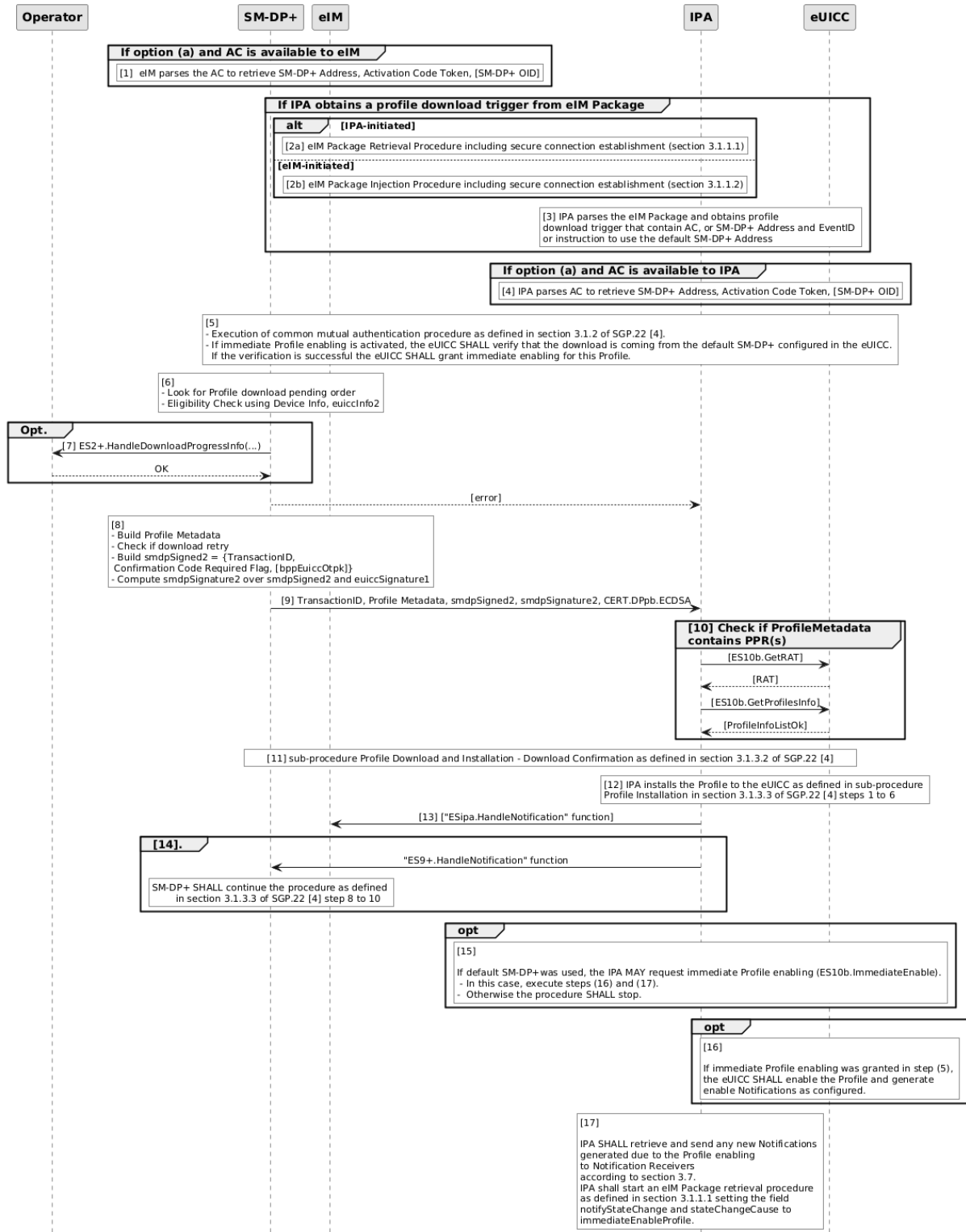
If default SM-DP+was used, the IPA MAY request immediate Profile enabling
(ES10b.ImmediateEnable).
- In this case, execute steps (16) and (17).
- Otherwise the procedure SHALL stop.
endrnote
end

group opt
rnote over E #FFFFFF
[16]

If immediate Profile enabling was granted in step (5),
the eUICC SHALL enable the Profile and generate
enable Notifications as configured.
endrnote
end

rnote over IPA #FFFFFF
[17]
```

IPA SHALL retrieve and send any new Notifications generated due to the Profile enabling to Notification Receivers according to section 3.7.  
 IPA shall start an eIM Package retrieval procedure as defined in section 3.1.1.1 setting the field notifyStateChange and stateChangeCause to immediateEnableProfile.  
 Endrnote  
 @enduml



## Figure 9 Direct Profile Download

### Start Conditions:

In addition to the start conditions required by the Common Mutual Authentication procedure defined in section 3.1.2 of SGP.22 [4], this procedure requires the following start conditions depending on the following exclusive options in step 1:

Option (a) use of an Activation Code: The eIM or IPA has an Activation Code that is coded as defined in section 4.1 of SGP.22 [4].

Option (b) use of SM-DS: The eIM or the IPA has previously been triggered to perform Event Retrieval and has the SM-DP+ Address and EventID from an SM-DS, as an outcome of the procedure defined in sections 3.9.2.1 or 3.9.2.2.

Option (c) use of default SM-DP+: The eIM or the IPA has been triggered to perform the Profile download from the default SM-DP+. The triggered entity (IPA/eIM) knows the default SM-DP+ Address (e.g. by IPA having called the function `ES10a.GetEuiccConfiguredAddresses` or by eIM having sent an eIM Package containing an `IpaEuiccDataRequest`).

NOTE: How the eIM is triggered is out of scope of this version of the specification. How the IPA is triggered could be through received `ProfileDownloadTriggerRequest` containing `contactDefaultSmdp` or by IoT Device specific means.

The Emergency Profile is not enabled.

### Procedure:

1. If option (a) is used and the Activation Code is available to the eIM, the eIM parses the Activation Code and finds the SM-DP+ address, Activation Code Token, and optional SM-DP+ OID. If the format of the Activation Code is invalid, the procedure SHALL stop.
2. In case of a Profile download trigger from an eIM Package: a secure connection is established between the IPA and the eIM for the IPA to obtain the eIM Package. How this is triggered is out of scope of this specification. There are two options for the delivery of the eIM Package to the IPA:
  - The eIM Package Retrieval Procedure defined in section 3.1.1.1 is executed between the IPA and the eIM, or
  - The eIM Package Injection Procedure defined in section 3.1.1.2 is executed between the IPA and the eIM.
3. The IPA parses the eIM Package received in step 2. The IPA identifies if the eIM Package contains a `ProfileDownloadTriggerRequest`. If option (a) is used this trigger SHALL contain the Activation Code. If option (b) is used this trigger contains the SM-DP+ Address and EventID (formatted as an Activation Code). If option (c) is used this trigger contains instruction to use the default SM-DP+ Address. If the format of the trigger profile download eIM Package is invalid, or data needed by IPA to perform the profile download is missing the IPA SHALL return `invalidPackageFormat` error and the procedure SHALL stop.

4. If the IPA retrieved an Activation Code in step 3, or if the IPA retrieved an Activation Code by some other means outside of this specification before the start of this procedure, the IPA parses the Activation Code and finds the SM-DP+ Address, Activation Code Token, and optional SM-DP+ OID. If the format of the Activation Code is invalid the IPA SHALL return `invalidPackageFormat` error and the procedure SHALL stop.
5. The common mutual authentication procedure defined in section 3.1.2 of SGP.22 [4] SHALL be executed. When this procedure is used for Profile download and installation, SM-XX is SM-DP+. CERT.XXauth.ECDSA, PK.XXauth.ECDSA and SK.XXauth.ECDSA are CERT.DPauth.ECDSA, PK.DPauth.ECDSA and SK.DPauth.ECDSA respectively. ESXX is ES9+.

During the common mutual authentication procedure at step (10) in section 3.1.2 of SGP.22 [4], the IPA SHALL verify that the SM-DP+ OID contained in the CERT.DPauth.ECDSA returned by the SM-DP+ is identical to the SM-DP+ OID if the IPA has acquired it from the Activation Code at step (4). If the comparison fails, the procedure SHALL stop.

During the common mutual authentication procedure at step (10) in section 3.1.2 of SGP.22 [4], the IPA SHALL build the `ctxParams1` data object to provide the MatchingID, Device Info to the eUICC for signature. The value of the MatchingID SHALL be set as follows:

- If an Activation Code is used (option (a)), the MatchingID value SHALL be set to Activation Code Token.
- If an SM-DS is used (option (b)), the MatchingID value SHALL be set to EventID.
- If a default SM-DP+ is used (option (c)), the MatchingID SHALL be missing.

If immediate Profile enabling is activated, the eUICC SHALL verify that the download is coming from the default SM-DP+ configured in the eUICC. If the verification is successful, the eUICC SHALL grant immediate enabling for this Profile.

6. After having successfully authenticated the eUICC at the end of the step (5) above, the SM-DP+ SHALL:
  - Verify that there is a related pending Profile download order for the provided MatchingID.
  - If this Profile download order is already linked to an EID, verify that it matches the EID of the authenticated eUICC.
  - Verify that the Profile corresponding to the pending Profile download order is in 'Released' state, or, in case of a retry due to a previous installation failure, in 'Downloaded' state (section 3.1.6 of SGP.22 [4]).

If any of these verifications fail, the SM-DP+ SHALL return the relevant error status and the procedure SHALL stop.

The SM-DP+ SHALL increment the count of download attempts for the identified Profile. If the maximum number of attempts has been exceeded, the SM-DP+ SHALL terminate the corresponding Profile download order and notify the Operator by calling the "ES2+.HandleDownloadProgressInfo" function with an operation status indicating

'Failed' with the relevant error status, and the procedure SHALL be stopped. Otherwise, the SM-DP+ SHALL perform appropriate eligibility checks, based on the Device Info and/or eUICCInfo2. These checks SHALL include the check if the eUICC can install one more Profile. See Annex F of SGP.22 [4] for more information on Eligibility checks.

7. (Optional step) Depending on the agreed behaviour with the Operator (out of scope of this specification), the SM-DP+ SHALL notify the Operator with the outcome of the eligibility check using the function "ES2+.HandleDownloadProgressInfo". The SM-DP+ SHALL provide the EID, the ICCID, the identification of the point reached (in that case it SHALL be 'Eligibility check'), the timestamp when this point was reached, and the execution result of this step.

NOTE: This notification step MAY be done asynchronously.

8. If the eligibility check fails, the SM-DP+ SHALL:
  - set the Profile corresponding with the pending Profile download order in 'Error' state (section 3.1.6 of SGP.22 [4]).
  - return an error status to the IPA and the procedure SHALL stop.

Otherwise, the SM-DP+ SHALL:

- Determine whether the Profile is already bound to the EID from a previous unsuccessful download attempt. If so, the SM-DP+ MAY include the otPK.Euicc.ECKA obtained in the previous session in the smdpSigned2 data structure.
- Determine if a Confirmation Code is required for this pending order.

NOTE: How the Confirmation Code is sent to the IoT Device is out of the scope of this specification.

- Generate a smdpSigned2 data structure containing the TransactionID and the Confirmation Code Required Flag.
- Compute the smdpSignature2 over smdpSigned2 and euiccSignature1 using the SK.DPpb.ECDSA.

9. The SM-DP+ returns the TransactionID, ProfileMetadata, smdpSigned2, smdpSignature2 and CERT.DPpb.ECDSA to the IPA.
10. On reception of the SM-DP+ response, the IPA SHALL verify the Profile Metadata according to steps 7a, b, c and 8 of section 3.1.3 (Profile Download and Installation) of SGP.22 [4]. For these verifications, the IPA MAY use previously fetched Rules Authorisation Table and/or list of installed Profiles.

NOTE: IoT Devices without user interface are not supposed to handle PPRs requiring user interaction in the RAT.

11. The IPA and eUICC process ES10b.PrepareDownload function call as defined in section 5.7.5 of SGP.22 [4] and the ongoing RSP session SHALL continue with the sub-procedure Profile Download and Installation – Download Confirmation as defined in section 3.1.3.2 of SGP.22 [4] with the following changes:

- Upon any error returned in the verifications of ES10b.PrepareDownload, the eUICC SHALL terminate the RSP Session.
12. The IPA installs the Profile to the eUICC as defined in section 3.1.3.3 (Sub-procedure Profile Installation) of SGP.22 [4], steps (1) to (6).  
  
NOTE: IPA implementations according to versions prior to V1.3 of this specification may perform the Sub-procedure "Profile Download and installation – Download rejection" as defined in SGP.22 [4] with reason code "metadataMismatch", if the Profile to be downloaded is an Emergency Profile, i.e. contains the `ecallIndication` data element set to TRUE within its Profile Metadata.
  13. In case of a Profile download trigger from an eIM Package: the IPA SHALL send the `ProfileDownloadTriggerResult` to the eIM as defined in section 2.11.2.3 by using the function "ESipa.HandleNotification".  
  
NOTE: This step is executed whenever an error is reported between steps 5 and 12.
  14. On reception of the "ES9+.HandleNotification" function, the SM-DP+ SHALL proceed as defined in section 3.1.3.3 of SGP.22 [4] step 8 to 10.
  15. If default SM-DP+ was used, the IPA MAY request immediate Profile enabling by calling ES10b.ImmediateEnable. In this case steps (16) and (17) SHALL be executed. Otherwise, the procedure SHALL stop.
  16. If immediate Profile enabling was granted in step (5), the eUICC SHALL enable the Profile and generate enable Notifications as configured.
  17. The IPA SHALL retrieve and send any new Notifications generated due to the Profile enabling to Notification Receivers according to section 3.7, and SHALL start an eIM Package Retrieval procedure, if applicable, as defined in section 3.1.1.1, setting the field `notifyStateChange`, together with setting `stateChangeCause` to `immediateEnableProfile`.

### 3.2.3.2 Indirect Profile Download

This section describes the indirect Profile download and installation procedure where the eIM is used as a proxy between the IoT Device and the SM-DP+.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>SM-DP+</b>" as DP
participant "<b>eIM</b>" as EIM
participant "<b>IPA</b>" as IPA
participant "<b>eUICC</b>" as E
```

```
group If option (a) and AC is available to eIM
rnote over EIM : [1] eIM parses the AC to retrieve SM-DP+ Address, Activation Code
Token, [SM-DP+ OID]
end

group If IPA obtains a Profile download trigger from eIM Package
alt IPA-initiated
rnote over EIM, IPA : [2a] eIM Package Retrieval Procedure including secure
connection establishment (section 3.1.1.1)
else eIM-initiated
rnote over EIM, IPA : [2b] eIM Package Injection Procedure including secure
connection establishment (section 3.1.1.2)
end

rnote over IPA : [3] IPA parses the eIM Package that contains \n
ProfileDownloadTriggerRequest
end

group If option (a) and AC is available to IPA
rnote over IPA : [4] IPA parses AC to retrieve SM-DP+ Address, Activation Code
Token, [SM-DP+ OID]
end

opt Get euiccInfo1 if not yet retrieved by IPA or eIM
IPA -> E : [5] ES10b.GetEuiccInfo
E --> IPA : euiccInfo1
end
IPA -> E : [6] ES10b.GetEuiccChallenge
E -> IPA : euiccChallenge

group If a secure connection is not yet established
rnote over EIM, IPA : [7] Establish secure connection
end
IPA -> EIM : [8] ESipa.InitiateAuthentication request
rnote over EIM, DP : [9] Establish HTTPS connection
rnote over EIM, DP : [10] ES9+'.InitiateAuthentication function
EIM -> IPA : [11] ESipa.InitiateAuthentication response
rnote over IPA, E: [12] ES10b.AuthenticateServer function
IPA -> EIM : [13] ESipa.AuthenticateClient request
rnote over EIM, DP : [14] ES9+'.AuthenticateClient function
EIM -> IPA : [15] ESipa.AuthenticateClient response
rnote over IPA, E: [16] ES10b.PrepareDownload function
IPA -> EIM : [17] ESipa.GetBoundProfilePackage request
rnote over EIM, DP : [18] ES9+'.GetBoundProfilePackage function
EIM -> IPA : [19] ESipa.GetBoundProfilePackage response
group Profile Installation Procedure
rnote over IPA, E : [20] Profile Installation
group opt
rnote over IPA #FFFFFF
[21]

If default SM-DP+ used, the IPA MAY request immediate Profile enabling
(ES10b.ImmediateEnable). Otherwise, continue in step (23).
endrnote
end

group opt
rnote over E #FFFFFF
[22]

If immediate Profile enabling was granted in step (12),
the eUICC SHALL enable the Profile and generate
enable Notifications as configured.
endrnote
end

rnote over DP, E : [23] Notification Handling
```

```
IPA -> EIM : [a] ESipa.HandleNotification
EIM -> IPA : [b] ACK
IPA -> E : [c] ES10b.RemoveNotificationFromList
EIM -> DP : [d] ES9+'.HandleNotification
rnote over DP : [e] Step 8-10 section 3.1.3.3 of SGP.22
end

rnote over DP, E #FFFFFF
[24]

IPA SHALL retrieve and send any new Notifications
generated due to the Profile enabling
to Notification Receivers
according to section 3.7.
endrnote

@enduml
```

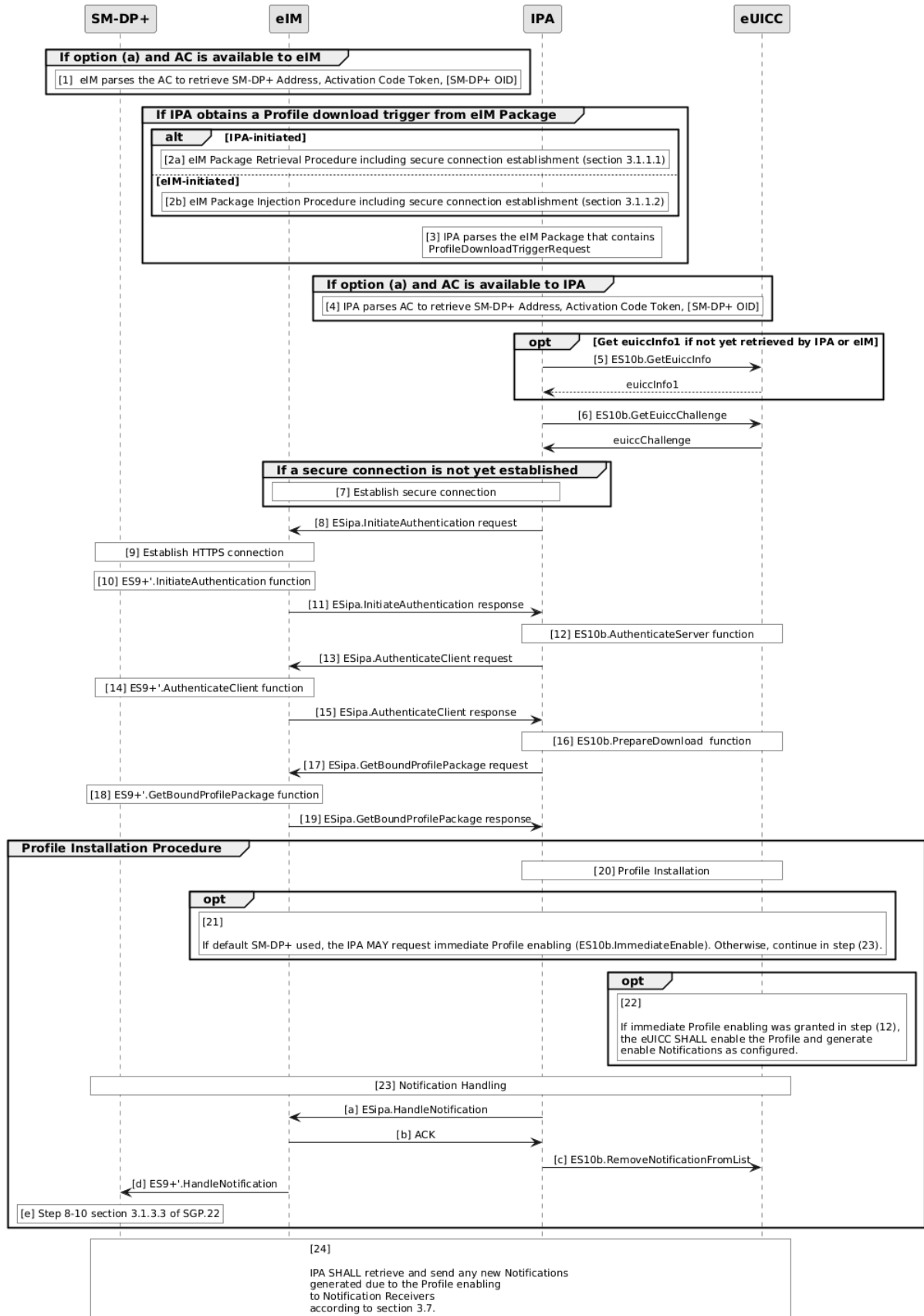


Figure 10 Indirect Profile Download and Installation

### Start Conditions:

In addition to the start conditions required by the Common Mutual Authentication procedure defined in section 3.1.2 of SGP.22 [4], this procedure requires the following start conditions depending on the following exclusive options in step 1:

Option (a) use of an Activation Code: The eIM or IPA has an Activation Code that is coded as defined in section 4.1 of SGP.22 [4].

Option (b) use of SM-DS: The eIM has retrieved the SM-DP+ Address and EventID from the SM-DS, as defined in section 3.9.2.2 of this document.

Option (c) use of default SM-DP+: The eIM or the IPA has been triggered to perform the Profile download from the default SM-DP+. The triggered entity (IPA/eIM) knows the default SM-DP+ Address (e.g.: by IPA having called the function ES10a.GetEuiccConfiguredAddresses or by eIM having sent an eIM Package containing an IpaEuiccDataRequest).

The Emergency Profile is not enabled.

IPA indicated in IPA Capability the support of `indirectRspServerCommunication`.

The eIM indicated the support of `indirectProfileDownload`.

### Procedure:

1. If option (a) is used and the Activation Code is available to the eIM, the eIM parses the Activation Code and finds the SM-DP+ address, Activation Code Token, and optional SM-DP+ OID. If the format of the Activation Code is invalid, the procedure SHALL stop.
2. In case a Profile download trigger from an eIM Package: a secure connection is established between the IPA and the eIM for IPA to obtain the eIM Package. How this is triggered is out of scope of this specification. There are two options for the delivery of the eIM Package to the IPA:
  - The eIM Package Retrieval Procedure steps 1 to 4 as defined in section 3.1.1.1 are executed between the IPA and the eIM.
  - The eIM Package Injection Procedure steps 1 and 2 as defined in section 3.1.1.2 are executed between the IPA and the eIM.
3. The IPA parses the eIM Package received in step 2. The IPA identifies that the eIM Package contains a `ProfileDownloadTriggerRequest`. If `ProfileDownloadTriggerRequest` data object is empty (i.e., without `profileDownloadData`), then the Procedure continues in step 5. If option (a) is used this trigger MAY contain the Activation Code. Alternatively, the eIM handles the AC (see IPA Capability `eimDownloadDataHandling`). If option (b) is used this trigger MAY contain the SM-DP+ Address and EventID (formatted as an Activation Code), if handled by IPA. If option (c) is used this trigger MAY contain information to use the default SM-DP+ Address. If the format of the trigger profile download eIM Package is invalid, or data needed by IPA to perform the profile download is missing

the IPA SHALL return `invalidPackageFormat` error and the procedure SHALL stop.

4. If the IPA retrieved an Activation Code in step 3, or if IPA retrieved an Activation Code by some other means outside of this specification before the start of this procedure, the IPA parses the Activation Code and finds the SM-DP+ address, Activation Code Token, and optional SM-DP+ OID. If the format of the Activation Code is invalid the IPA SHALL return `invalidPackageFormat` error and the procedure SHALL stop.
5. If `euiccInfo1` is not yet retrieved by the eIM or IPA, the IPA MAY request `euiccInfo1` from the eUICC by calling `ES10b.GetEuiccInfo` (see section 5.9.2).
6. The IPA requests an eUICC Challenge from the eUICC by calling the `ES10.GetEUICCChallenge` function (see section 5.9.3).
7. A secure connection is established between the IPA and the eIM if it is not yet established.
8. The IPA calls `ESipa.InitiateAuthentication` function comprising eUICC Challenge, optionally SM-DP+ FQDN, and optionally `euiccInfo1` (see IPA Capability `minimizeEsipaBytes`) and `eimTransactionId` (if included by eIM in `ProfileDownloadTriggerRequest`). If the IPA does not provide SM-DP+ FQDN or `euiccInfo1`, the eIM SHALL identify the relevant information.
9. The eIM establishes an HTTPS connection with the SM-DP+ in server authentication mode according to SGP.22 [4].
10. The eIM and the SM-DP+ process `ES9+'.InitiateAuthentication` function comprising eUICC Challenge, SM-DP+ FQDN, and `euiccInfo1` as defined in section 5.6.1 of SGP.22 [4]. If the SM-DP+ FQDN provided in `ES9+'.InitiateAuthentication` in step 9 did not come from IPA (was not obtained from IPA in step 7 e.g., due to IPA has capability `eimDownloadDataHandling`), the eIM SHALL verify that this SM-DP+ FQDN matches the SM-DP+ FQDN returned by the SM-DP+ (in `serverSigned1`). If not, the procedure SHALL be stopped. If the Activation Code contains the SM-DP+ OID and is available to the eIM, the eIM SHALL check that the SM-DP+ OID from the AC matches the SM-DP+ OID of the SM-DP+ Certificate (`serverCertificate`). If not, the procedure SHALL stop.
11. The eIM sends the `ESipa.InitiateAuthentication` response to the IPA, based on the `ES9+'.InitiateAuthentication` response from the SM-DP+. The eIM SHALL additionally provide `ctxParams1` as defined in SGP.22 [4] if the IPA does not generate it (see IPA capability `eimCtxParams1Generation`).
12. The IPA and eUICC process `ES10b.AuthenticateServer` function call as defined in section 5.7.13 of SGP.22 [4]. If immediate Profile enabling is activated, the eUICC SHALL verify that the download is coming from the default SM-DP+ configured in the eUICC. If the verification is successful, the eUICC SHALL grant immediate enabling for this Profile.

13. The IPA calls ESipa.AuthenticateClient function comprising euiccSigned1 or compactEuiccSigned1, euiccSignature1, optionally CERT.EUM.ECDSA, and optionally CERT.EUICC.ECDSA. If the IPA does not provide CERT.EUM.ECDSA and CERT.EUICC.ECDSA and/or provide the compactEuiccSigned1 instead of euiccSigned1 (see IPA Capability `minimizeEsipaBytes`), the eIM SHALL identify the relevant information and build euiccSigned1.
14. The eIM and SM-DP+ process the ES9+'.AuthenticateClient function as defined in 5.6.3 of SGP.22 [4].
15. The eIM sends the ESipa.AuthenticateClient response to the IPA, based on the ES9+'.AuthenticateClient response from the SM-DP+. If the response returns an error, IPA SHALL trigger the cancellation of the on-going RSP session (see section 3.2.3.3) with reason `undefinedReason` and the procedure SHALL stop.
  - If the IPA is capable of verifying the Profile Metadata, the eIM SHALL provide the Profile Metadata to IPA that SHALL verify the Profile Metadata according to section 3.2.3.1 (Direct Profile Download) step 10. For this verification, the IPA MAY use previously fetched Rules Authorisation Table and/or list of installed Profiles. If the IPA has not already fetched the required information, the IPA SHALL request those from the eUICC by calling the ES10b.GetRAT and/or ES10b.GetProfilesInfo functions. If the verification fails, the IPA SHALL trigger the cancellation of the on-going RSP session (see section 3.2.3.3) and the procedure SHALL stop.
  - If the IPA is not capable of verifying the Profile Metadata (see IPA Capability `eimProfileMetadataVerification`) according to section 3.1.3 (Profile Download and Installation) of SGP.22 [4], the eIM SHALL verify the Profile Metadata according to steps 7a, b, c and 8 of section 3.1.3 (Profile Download and Installation) of SGP.22 [4] and SHALL NOT provide the Profile Metadata to IPA. For this verification, the eIM SHALL use Rules Authorisation Table and information on installed Profiles available to the eIM, e.g., obtained prior to the start of this procedure using the PSMOs `getRAT` and `listProfileInfo`. If the verification fails, the eIM SHALL return an error code to IPA and the procedure SHALL stop. The error code triggers IPA to cancel the on-going RSP session with SM-DP+ according to section 3.2.3.3. The cancel session reason (`pprNotAllowed`) to be used by IPA when calling the ESipa.CancelSession is given by the error code.
16. The IPA and eUICC process ES10b.PrepareDownload function call as defined in section 5.7.5 of SGP.22 [4] with the following change:
  - Upon any error returned in the verifications of ES10b.PrepareDownload, the procedure SHALL stop and the eUICC SHALL terminate the RSP Session.
17. The IPA calls ESipa.GetBoundProfilePackage function comprising euiccSignature2 and either euiccSigned2 or compactEuiccSigned2. If compactEuiccSigned2 is provided (see IPA Capability `minimizeEsipaBytes`), the eIM SHALL identify the relevant information and build euiccSigned2.
18. The eIM and SM-DP+ process ES9+'.GetBoundProfilePackage function as defined in section 5.6.2 of SGP.22 [4].

NOTE: In the ES9+'.GetBoundProfilePackage function request, the eIM forwards the ESipa.GetBoundProfilePackage function request.

19. The eIM sends the ESipa.GetBoundProfilePackage response to the IPA, based on the ES9+'.GetBoundProfilePackage response from the SM-DP+. If the eIM or IPA previously verified the Profile Metadata in step (15), the eIM or IPA SHOULD check if the Profile Metadata has been changed. If so, IPA SHALL be triggered to cancel the on-going RSP session with the SM-DP+ (see section 3.2.3.3) with cancel session reason metadataMismatch and the procedure SHALL be terminated. In case of eIM performing the check (see IPA Capability `eimProfileMetadataVerification`), the eIM SHALL return an error code that triggers IPA to cancel the session and also indicates the cancel session reason (`metadataMismatch`) to be used.
20. The IPA installs the Profile to the eUICC as defined in section 3.1.3.3 (Sub-procedure Profile Installation) of SGP.22 [4], steps (1) to (6).
  - If there are errors during the installation of the Profile:
    - if a Profile Installation Result is available, the procedure SHALL continue in step (23).
    - otherwise, the cancel session procedure in section 3.2.3.3 SHALL be triggered with cancel session reason `loadBppExecutionError` and the procedure SHALL stop.

NOTE: IPA implementations according to versions prior to V1.3 of this specification may perform the Sub-procedure "Profile Download and installation – Download rejection" as defined in SGP.22 [4] with reason code "metadataMismatch", if the Profile to be downloaded is an Emergency Profile, i.e. contains the `ecallIndication` data element set to TRUE within its Profile Metadata.

21. If default SM-DP+ was used, the IPA MAY request immediate Profile enabling by calling ES10b.ImmediateEnable and the procedure SHALL continue with step (22). Otherwise, the procedure SHALL continue in step (23).
22. If immediate Profile enabling was granted in step (12), the eUICC SHALL enable the Profile and generate enable Notifications as configured.
23. The following changes are made to the Notification procedure compared to section 3.1.3.3 of SGP.22 [4]:
  - The IPA calls ESipa.HandleNotification function comprising PendingNotification data object that contains ProfileInstallationResult that contains the TransactionID generated by the SM-DP+. If a compact version of the ProfileInstallationResult is provided (see IPA Capability `minimizeEsipaBytes`), the eIM SHALL identify the relevant information and build the full ProfileInstallationResult.
  - The eIM acknowledges the reception of the ProfileInstallationResult.
  - Upon acknowledgement of successful reception of the ProfileInstallationResult, IPA calls the ES10b.RemoveNotificationFromList function with corresponding seqNumber as input parameter and the eUICC deletes the ProfileInstallationResult from its memory.

- The eIM and SM-DP+ process ES9+'.HandleNotification function as defined in section 5.7.4 of this document.
- On reception of the ES9+'.HandleNotification function, the SM-DP+ SHALL proceed as defined in section 3.1.3.3 of SGP.22 [4] step 8 to 10.

NOTE: the IPA may instead perform the handling of the PIR Notification as defined in step (23) before step (21).

24. The IPA SHALL retrieve and send any new Notifications generated due to the Profile enabling to Notification Receivers according to section 3.7. If a compact version of the OtherSignedNotification is provided (see IPA Capability minimizeEsipaBytes), the eIM SHALL identify the relevant information and build the full OtherSignedNotification. The IPA SHALL start an eIM Package Retrieval procedure, if applicable, as defined in section 3.1.1.1, setting the field `notifyStateChange`, together with setting `stateChangeCause` to `immediateEnableProfile`.

### 3.2.3.3 Cancel Session Procedure for Indirect Profile Download

This section describes the cancel session call flow used to cancel an indirect Profile download session. This procedure can occur due to an error at the following steps of the protocol:

- after the response to "ESipa.AuthenticateClient"
- after the response to "ESipa.GetBoundProfilePackage"

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>SM-DP+</b>" as DP
participant "<b>eIM</b>" as eIM
participant "<b>IPA</b>" as IPA
participant "<b>eUICC</b>" as E

IPA -> E : [1] ES10b.CancelSession(TransactionID, reason)
rnote over E
[2]
- Generate euiccCancelSessionSigned = {
  TransactionID, reason}
- Compute euiccCancelSessionSignature
  over euiccCancelSessionSigned
endrnote

E --> IPA : [3] cancelSessionResponseOk

rnote over IPA
[Stop procedure if the reason is undefinedReason]
endrnote

IPA -> eIM : [4] ESipa.CancelSession(TransactionID, cancelSessionResponseOk)
eIM -> DP : [4] ES9+'.CancelSession(TransactionID, cancelSessionResponseOk)

rnote over DP
[5]
- Retrieve the on-going RSP session
- Verify euiccCancelSessionSignature
```

```

- Verify the SM-DP+ OID
endnote
DP --> eIM : [ERROR]
eIM --> IPA : [ERROR]

rnote over DP
[6] Process the reason according to SGP.22 [4]
endnote

DP --> eIM : [10] OK
eIM --> IPA : [10] OK
@enduml
    
```

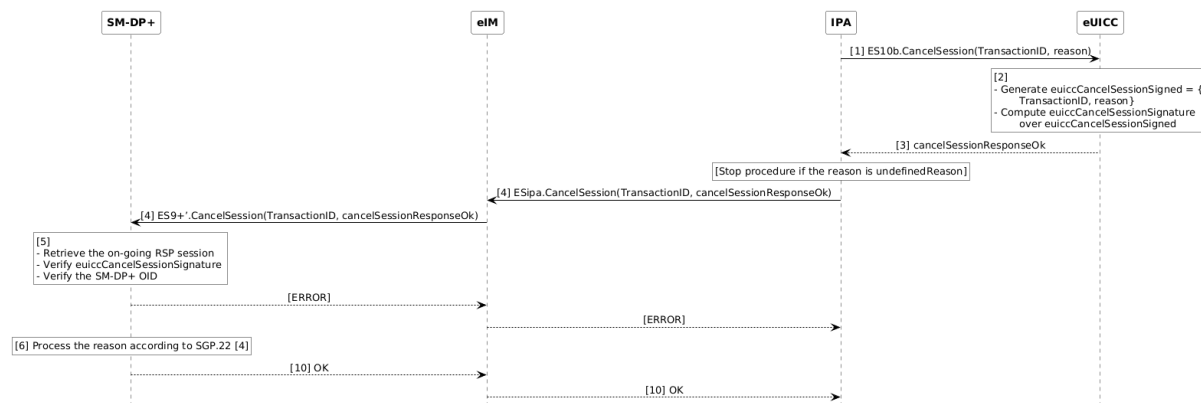


Figure 11 Cancel Session Procedure

**Start Conditions:**

This procedure can be used in the following cases.

Reasons in the response to "ESipa.AuthenticateClient":

- The IPA receives "pprNotAllowed" error in the response to "ESipa.AuthenticateClient" due to that eIM, when verifying the Profile Metadata, discovered that the PPR(s) in the Profile Metadata are not allowed according to the Rules Authorisation Table, or PPR1 is present in the Profile Metadata and an Operational Profile is already installed on the eUICC. In this case the reason code for step (1) SHALL be pprNotAllowed.
- The IPA receives any other error in the response to "ESipa.AuthenticateClient". In this case the reason for step (1) SHALL be undefinedReason.

Cancel reasons after "ESipa.AuthenticateClient" related to Profile download:

- The IPA verifies Profile Metadata and discovers that the PPR(s) in the Profile Metadata are not allowed according to the Rules Authorisation Table, or PPR1 is present in the Profile Metadata and an Operational Profile is already installed on the eUICC. In these cases, the reason code for step (1) SHALL be pprNotAllowed.

Reasons in the response to "ESipa.GetBoundProfilePackage":

- The IPA receives metadataMismatch error in the response to "ESipa.GetBoundProfilePackage" due to that eIM discovered that the Profile Metadata in the Bound Profile Package does not match the Profile Metadata received previously in the response to "ES9+'.AuthenticateClient". In this case the reason code for step (1) SHALL be metadataMismatch.

Cancel reasons after "ESipa.GetBoundProfilePackage":

- The IPA discovers that the Profile Metadata in the Bound Profile Package does not match the Profile Metadata received previously in the response to "ESipa.AuthenticateClient". In this case the reason for step (1) SHALL be metadataMismatch.
- The IPA has encountered an error while installing a Bound Profile Package. In this case the reason for step (1) SHALL be loadBppExecutionError.

**Procedure:**

1. The IPAd SHALL call the "ES10b.CancelSession" function with input data comprising the TransactionID and the reason.
2. The eUICC SHALL:
  - Generate the euiccCancelSessionSigned data object containing the TransactionID and the reason provided by the IPAd.
  - Compute the euiccCancelSessionSignature over euiccCancelSessionSigned using the SK.EUICC.ECDSA corresponding to the euiccCiPKIdentifierToBeUsed as received during the common mutual authentication procedure.
3. The eUICC SHALL return the euiccCancelSessionSigned and euiccCancelSessionSignature. If the reason is undefinedReason, the IPAd SHALL ignore the response from the eUICC and stop the procedure.
4. The IPA SHALL call the "ESipa.CancelSession" function with input data comprising TransactionID, euiccCancelSessionSigned and euiccCancelSessionSignature. Upon receiving the "ESipa.CancelSession" function call, the eIM SHALL call the "ES9+'.CancelSession" function with the input data from "ESipa.CancelSession".
5. On reception of the "ES9+'.CancelSession" function, the SM-DP+ SHALL:
  - Retrieve the on-going RSP session identified by the TransactionID. If the TransactionID is unknown, the SM-DP+ SHALL return a function execution status 'Failed' with relevant status code and the procedure SHALL stop.
  - Verify the euiccCancelSessionSignature performed over euiccCancelSessionSigned using the PK.EUICC.ECDSA associated with the ongoing RSP session. If the signature is invalid, the SM-DP+ SHALL return a function execution status 'Failed' with relevant status code and the procedure SHALL stop.
  - Verify that the received OID is the same value as the one contained in the CERT.DPauth.ECDSA used during the common mutual authentication procedure. If the value does not match, the SM-DP+ SHALL return a function execution status 'Failed' with relevant status code and the procedure SHALL stop.
6. The SM-DP+ SHALL process the error reason according to steps 5 to 7 in Section 3.1.3.1 of SGP.22 [4].

7. The SM-DP+ SHALL return a function execution status 'Executed-Success' and the procedure SHALL stop. The eIM returns the function execution status from "ES9+'.CancelSession" function as the function execution status of "ESipa.CancelSession".

#### **3.2.3.4 Cancel Session Procedure for Direct Profile Download**

This section describes the procedure to cancel a direct Profile download session.

The procedure is identical to Sub-procedure Profile Download and Installation – Download Rejection, described in Section 3.1.3.1 of SGP.22 [4], where the IPA plays the role of LPA and where, due to the lack of End User, the start conditions are updated according to below with relevant values of CancelSessionReason.

##### **Start Conditions:**

This procedure can be used in the following cases.

- Common Mutual Authentication Procedure, described in Section 3.1.2 of SGP.22 [4], has been performed and an error occurred during its step (16): The IPA receives any error in the response to "ES9+.AuthenticateClient". In this case the reason for step (1) SHALL be undefinedReason.

After Common Mutual Authentication Procedure, described in Section 3.1.2 of SGP.22 [4], has been performed:

- The IPA verifies Profile Metadata and discovers that the PPR(s) in the Profile Metadata are not allowed according to the Rules Authorisation Table, or PPR1 is present in the Profile Metadata and an Operational Profile is already installed on the eUICC. In these cases, the reason code for step (1) SHALL be pprNotAllowed.

After Sub-procedure Profile Download and Installation – Download Confirmation, described in Section 3.1.3.2 of SGP.22 [4]:

- The IPA discovers that the Profile Metadata in the Bound Profile Package does not match the Profile Metadata received previously in the response to "ES9+.AuthenticateClient". In this case the reason for step (1) SHALL be metadataMismatch.
- The IPA has encountered an error while installing a Bound Profile Package. In this case the reason for step (1) SHALL be loadBppExecutionError.

### **3.3 eUICC Package Handling**

#### **3.3.1 Generic eUICC Package Download and Execution**

This procedure describes the download of an eUICC Package contained within an eIM Package. It also describes how to process the eUICC Package and how to return a signed result of the eUICC Package execution to the eIM.

NOTE: The Sub-procedures for Profile State Management, described in section 3.4, and eIM Configuration, as described in section 3.5.1, are executed in step 5 of the procedure described in this section.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
skinparam ParticipantPadding 70
hide footbox

participant "<b>SM-DP+" as DP
participant "<b>eIM" as eIM
participant "<b>IPAd" as IPAd
participant "<b>eUICC" as E
participant "<b>Device Baseband" as DevBB

rnote left IPAd
[1]
- Build euiccPackageSigned = {eimId, eidValue, counterValue, [eimTransactionId],
euiccPackage}
  and compute eimSignature over euiccPackageSigned
- Build eIMPackage
endrnote

rnote over eIM, IPAd: [2] Transfer eIM Package containing signed \n eUICC Package
from eIM to IPAd (see section 3.1.1)

group Process the eIM Package
IPAd -> E : [3] ES10b.LoadEuiccPackage (eUICC Package)

rnote over E: [4] Verify the eIM signature \nand check EID and replay counter

E --> IPAd : [ERROR]

loop Up to the number of\n PSMOs/eCOs in the eUICC Package
rnote over E
[5] Execute PSMO/eCO
endrnote
end

rnote over E: [6] Generate and sign the result of the eUICC Package \nincluding
eUICC sequence number

end
E -> IPAd : [7] Signed eUICC Package Result
opt Any marked Profile
opt "to be disabled", "to be disabled and deleted", or "to be enabled"
E -> DevBB: [8a] REFRESH
DevBB -> E: Terminal Response or RESET
End opt
rnote over E
[8b] The marked Profile(s) are enabled, disabled and/or deleted
endrnote

rnote over DevBB
[Network attach procedure
with the newly enabled Profile]
end rnote
end opt

group If the IPAd sends eUICC Package Result and Notifications to the eIM in a single
eIM Package Result
IPAd -> E : [9] ES10b.RetrieveNotificationsList
E -> IPAd : List of pending Notifications

end
```

```
rnote over eIM, IPA: [10] Transfer eIM Package Result containing signed eUICC
Package Result \nand optionally Notifications list from IPAd to eIM (see section
3.1.1)

rnote over eIM: [11] Extract eUICC Package Result, \nverify the eUICC signature and
check \nsequence number to prevent replay

rnote over eIM: [12] Process the result of the eUICC Package

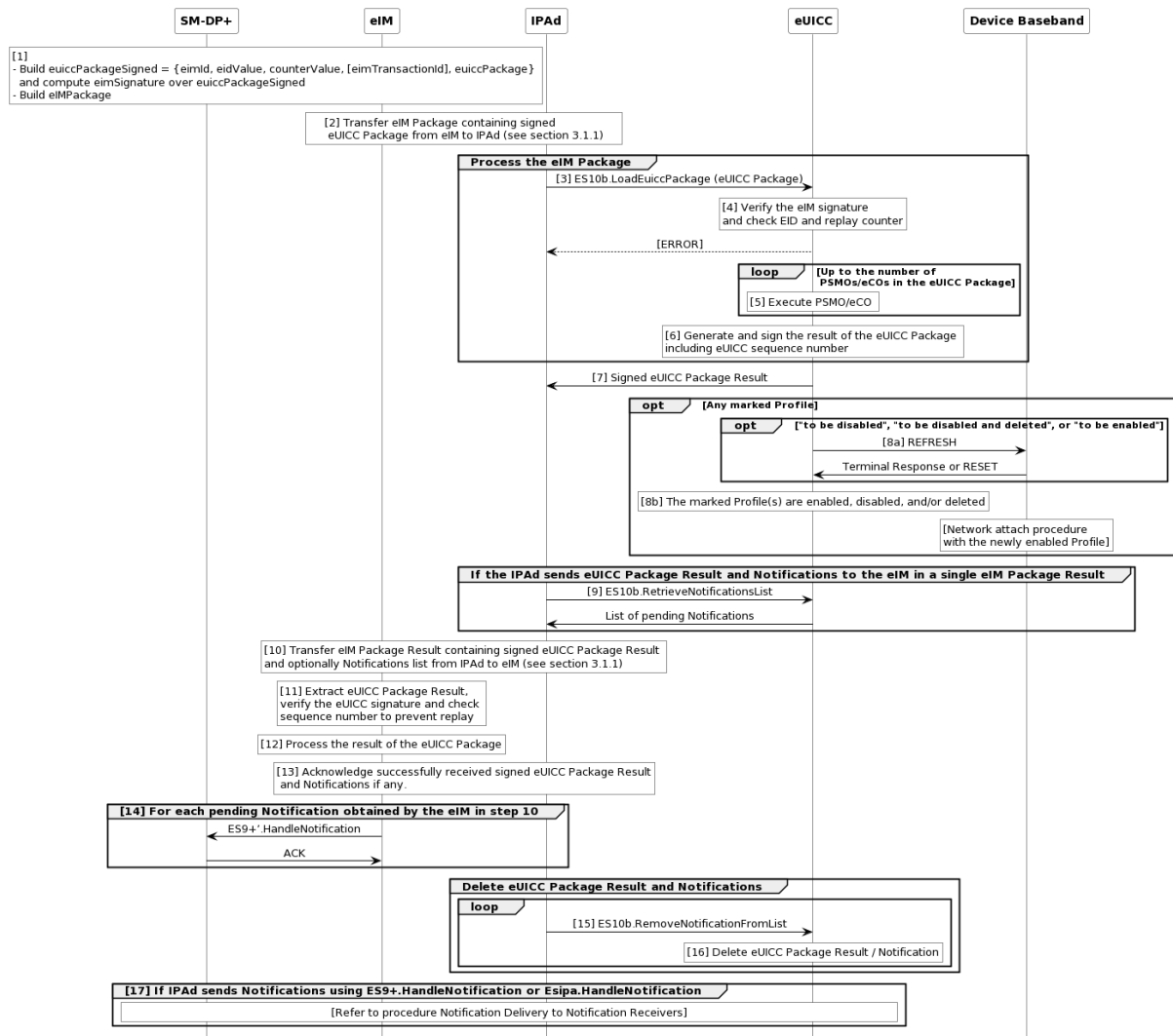
rnote over eIM, IPA: [13] Acknowledge successfully received signed eUICC Package
Result\n and Notifications if any.
group [14] For each pending Notification obtained by the eIM in step 10
eIM -> DP : ES9+'.HandleNotification
DP -> eIM : ACK
end

group Delete eUICC Package Result and Notifications
loop
IPA -> E : [15] ES10b.RemoveNotificationFromList

rnote over E: [16] Delete eUICC Package Result / Notification
end
end

group [17] If IPAd sends Notifications using ES9+.HandleNotification or
ESipa.HandleNotification
rnote over DP, E #FFFFFF : [Refer to procedure Notification Delivery to Notification
Receivers]
end

@enduml
```



**Figure 12 Generic eUICC Package Download and Execution**

**Start Condition:**

This procedure requires the following start conditions:

- The eUICC is associated with the eIM and has the eIM public key and eIM ID.
- The eIM has received a request to perform a PSMO/eCO for the particular eUICC and has the EID of the eUICC in its storage along with the current value of the counter for replay protection as well as the relevant information of the target Profile in the eUICC.
- A Secure Connection between the eIM and the IPA is established.

**Procedure:**

1. The eIM SHALL prepare the eUICC Package structure including signing the eUICC Package using the eIM private key. The signed data includes, for the particular eUICC, the eIMID, the EID and the current value incremented by 1 of the counters for replay protection of eUICC Packages obtained from eIM storage. The current value of the counter for replay protection in eIM storage is updated accordingly. The eIM then builds

the eIM Package containing the signed eUICC Package. `eimTransactionId`, if included, MAY be used for linking an eUICC Package with an eUICC Package Result.

2. The eIM SHALL transfer the eIM Package (containing the signed eUICC Package) to the IPAd as described in section 3.1.1.
3. IPAd then processes the eIM Package. The IPAd SHALL call `ES10b.LoadEuiccPackage` to provide the signed eUICC Package to the eUICC.
4. The eUICC SHALL verify the eIM signature and check the EID and replay counter. See details in section 5.9.1. In case of error, the eUICC SHALL return an error message and the procedure SHALL stop.
5. If all verifications are successful, the eUICC SHALL process the eUICC Package. The PSMOs or eCOs in the eUICC Package SHALL be executed sequentially by the eUICC until the end is reached or an error is encountered for a PSMO/eCO. If an interruption (e.g. power loss) occurs during the processing of the `ES10b.LoadEuiccPackage` function, the eUICC SHALL restore its original state prior to execution, see details in section 5.9.1.
6. The eUICC SHALL generate the result of the eUICC Package execution and sign it using the eUICC private key `SK.EUICC.ECDSA`. The signed data includes the next (not used) value of the eUICC sequence number (defined in SGP.22 [4] for use with Notifications) and the eUICC sequence number is incremented by 1 within the eUICC. The eUICC SHALL update the stored counter value to the counter value of the signed eUICC Package received in step 4.
7. The IPAd obtains the signed eUICC Package Result from the eUICC.

If any Profile is marked as "to be disabled", "to be disabled and deleted", "to be enabled", or "to be deleted":

- 8a. If any Profile is marked as "to be disabled", "to be disabled and deleted", or "to be enabled", the eUICC SHALL send the REFRESH command in "eUICC Profile State Change" mode (if supported by the Device) or "UICC Reset" mode to the Device, according to ETSI TS 102 223 [5], to prepare the IoT Device for a change in Profile state.

NOTE: Enabling or disabling via PSMO always uses REFRESH.

- 8b. Upon reception of the Terminal Response or after the RESET, or in case the REFRESH command is not sent, the ISD-R SHALL:
  - If a Profile is marked "to be disabled": disable the marked Profile.
  - If a Profile is marked "to be disabled and deleted": disable the marked Profile and then delete it.
  - If a Profile is marked "to be enabled": enable the marked Profile.
  - If a Profile is marked "to be deleted": delete the marked Profile.
  - If any marked Profile is successfully enabled, disabled, or deleted in the previous steps, generate as many Notifications as configured in each Profile

Metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.

- Unmark all marked Profiles.

If a Profile is now in Enabled state, the IoT Device baseband executes the network attach procedure.

9. If the IPAd sends eUICC Package Result and Notifications to the eIM in a single eIM Package Result and if the eUICC Package contains PSMO(s), the IPAd SHALL retrieve pending Notifications by calling `ES10b.RetrieveNotificationsList` function.
10. The IPAd SHALL transfer the eIM Package Result containing the signed eUICC Package Result to the eIM. The details on the transfer of the eIM Package Result depends on the underlying ESipa transport and is further described in section 3.1.1. If the IPAd sends eUICC Package Result and Notifications to the eIM in a single eIM Package Result (using `ESipa.ProvideEimPackageResult`), the IPAd SHALL, in case of a non-empty list of pending Notifications, include the list of pending Notifications (`PendingNotificationList` as defined in section 2.11.2.2) together with the signed eUICC Package Result in the eIM Package Result. In case of an IPAd with IPA Capability `minimizeEsipaBytes`, the IPAd SHOULD include each pending Notification in the list in compact format as described in section 5.14.7.

If the IPAd does not send Notifications together with the eUICC Package Result, the IPAd MAY use `ESipa.HandleNotification` instead to send the eUICC Package Result to the eIM.

If the IPAd fails sending the eIM Package Result to the eIM due to a lack of connectivity, it SHOULD apply the Profile Rollback procedure described below.

11. The eIM SHALL extract the eUICC Package Result and verify the eUICC signature of the signed eUICC Package Result using the public key `PK.EUICC.ECDSA` of the eUICC obtained from the eUICC Certificate in the eIM storage. Upon successful verification, the eIM SHALL retrieve the counter value included in the eUICC Package Result and map it to the related eUICC Package. It SHALL then check that the sequence number contained in the eUICC Package Result is greater than the value of the sequence number currently stored in the eIM for the particular eUICC. If all checks are successful, the eIM SHALL update the stored value of the sequence number for the particular eUICC to the value of the signed eUICC Package Result. If any of the checks fail, the eIM SHALL stop the processing of the eUICC Package Result. In case the eIM received pending Notifications in step 10, the eIM continues the execution in step 13 to acknowledge the received Notifications (but not the eUICC Package Result) and sends the received Notifications to Notification Receivers (see step 14).
12. The eIM SHALL process the result of the eUICC Package execution.
13. If the eIM Package Result was transmitted using the `ESipa.ProvideEimPackageResult` function (resp. the response to `ESipa.TransferEimPackage`), then the eIM SHALL acknowledge the signed eUICC Package Result and Notifications (if any) that were successfully processed by sending their sequence numbers in the response to `ESipa.ProvideEimPackageResult` (resp. the next call to `ESipa.TransferEimPackage`).

14. If the If the eIM Package Result was transmitted using ESipa.HandleNotification, then the eIM does not acknowledge the signed eUICC Package Result and received Notifications (if any). In case the eIM received pending Notifications in step 10, the eIM SHALL forward them to the Notification Receivers. If a pending Notification is in compact format (see IPA Capability `minimizeEsipaBytes`), the eIM SHALL identify the relevant information and build the full pending Notification before forwarding the Notification to the Notification Receiver.
15. IPAd SHALL call ES10b.RemoveNotificationFromList (see SGP.22 [4]) one or more times to delete the eUICC Package Result and pending Notifications for which acknowledgement has been received. The eUICC Package Result and each pending Notification to be deleted are identified by the sequence number(s) received in step 13.
16. The eUICC SHALL delete the eUICC Package Result or pending Notification from its memory.
17. If IPAd sends Notifications using ES9+.HandleNotification or ESipa.HandleNotification, the IPAd SHALL continue execution according to procedure "Notification Delivery to Notification Receivers."

NOTE: Notifications (if any) needs to be sent during this procedure by one of described means (either together with eUICC Package Results in step 10, or using ES9+.HandleNotification or ESipa.HandleNotification in step 17).

### 3.3.2 Profile Rollback Procedure

If the IPAd fails sending the eIM Package Result to the eIM due to a lack of connectivity, it SHOULD call the ES10b.ProfileRollback function (see section 5.9.16) to request the eUICC to roll back to the previously Enabled Profile (if any).

If the result of ES10b.ProfileRollback indicates successful processing (the eUICC returns 'ok'), then a new eUICC Package Result is available, and:

- The IPAd SHALL discard the previous eIM Package Result, i.e., the IPAd SHALL NOT send the previous eIM Package Result to the eIM.
- The IPAd SHALL build a new eIM Package Result including the new eUICC Package Result returned by the ES10b.ProfileRollback function and try to send this new eIM Package Result to the eIM.

NOTE1: When the ES10b.ProfileRollback is successful, the eUICC automatically discards the previous eUICC Package Result (see section 5.9.16) and there is no need for the IPAd to remove it from the eUICC. In particular, the ES10b.RemoveNotificationFromList function would return `nothingToDelete(1)` if the IPAd tried to do so.

NOTE2: As Profiles have returned to their states prior to the processing of the eUICC Package Request, as an optimization, the eUICC may discard Notifications (if any) generated by the processing of the eUICC Package Request and ES10b.ProfileRollback.

## 3.4 Profile State Management

### 3.4.1 Enable Profile

This procedure defines the execution of an Enable command contained within a eUICC Package as defined in 3.3.1 Generic eUICC Package Download and Execution, used to enable a Profile already downloaded and installed on an eUICC.

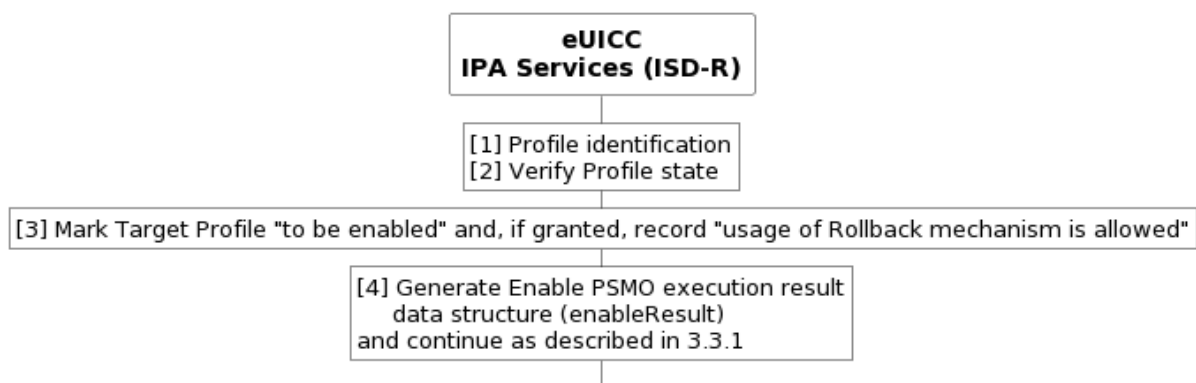
```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC\n<b>IPA Services (ISD-R)" as IPAServices

rnote over IPAServices
[1] Profile identification
[2] Verify Profile state
End rnote

rnote over IPAServices
[3] Mark target Profile "to be enabled" and, if granted, record "usage of Rollback Mechanism is allowed"
endrnote

rnote over IPAServices
[4] Generate Enable PSMO execution result
    data structure (enableResult)
and continue as described in 3.3.1
endrnote
@enduml
```



**Figure 13 Enable Profile**

#### Start Conditions:

A Profile is already downloaded and installed in the eUICC. A eUICC Package containing an 'Enable Profile' is received within an "ES10b.LoadEuiccPackage" by the eUICC as described in 3.3.1 Generic eUICC Package Download and Execution.

The eIM signature the EID, and replay counter are verified as defined in 3.3.1 Generic eUICC Package Download and Execution.

**Procedure:**

If an Enable command has already been processed in this eUICC Package, then the ISD-R SHALL indicate a failure in the PSMO execution result and procedure SHALL continue in step 4.

1. The ISD-R SHALL find the target Profile with the ICCID. If the target Profile is not found, the ISD-R SHALL indicate a failure in the PSMO execution result with `iccidOrAidNotFound(1)` error code and procedure SHALL continue in step 4. If the target Profile is the Emergency Profile with `ecallIndication` set to TRUE, the ISD-R SHALL indicate a failure in the PSMO execution result with `undefinedError(127)` error code and procedure SHALL continue in step 4.
2. The ISD-R SHALL verify the state of the target Profile and, if usage of the Rollback Mechanism is granted, whether a Profile is currently enabled (or marked as "to be disabled"). If the target Profile is not in Disabled state, or if usage of the Rollback Mechanism is granted and there is no currently Enabled Profile (or no Profile marked as "to be disabled"), then the ISD-R SHALL indicate a failure in the PSMO execution result respectively with `profileNotInDisabledState(2)` or `rollbackNotAvailable(20)` error codes and procedure SHALL continue in step 4.
3. The ISD-R SHALL mark the target Profile "to be enabled" and the currently Enabled Profile (if any) "to be disabled" and SHALL record whether usage of the Rollback Mechanism is granted (together with a reference to the Profile (if any) which is to be disabled), and clear any information of the previously enabled Profile referenced by the Fallback Mechanism, if any.

NOTE: If the currently enabled Profile is a Fallback Profile it is also marked as "to be disabled".

4. The eUICC SHALL generate the PSMO execution result data structure indicating the result of the PSMO 'Enable Profile', and the procedure continues as described in section 3.3.1.

**End Conditions:**

The target Profile is marked "to be enabled" and the currently Enabled Profile (if any) is marked "to be disabled". If granted, the authorisation to use the Rollback Mechanism has been recorded (together with a reference to the Profile (if any) which is to be disabled).

The signed eUICC Package execution result data structure containing the result of Enable command is stored in the eUICC.

### 3.4.2 Disable Profile

This procedure defines the execution of a Disable command contained within a eUICC Package as defined in 3.3.1 Generic eUICC Package Download and Execution, used to disable a Profile already downloaded and installed on an eUICC.

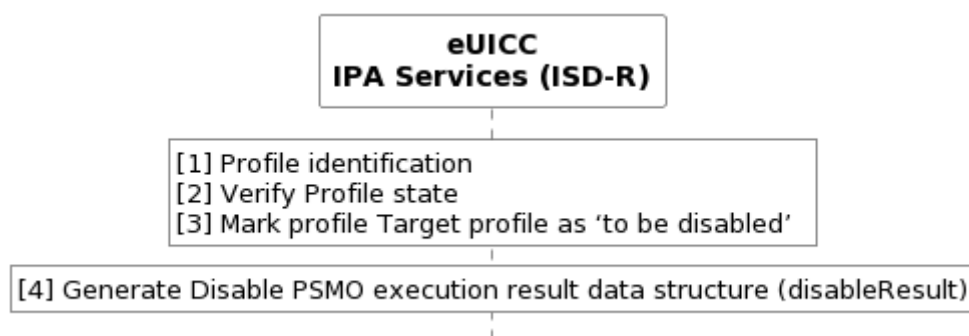
```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC\n<b>IPA Services (ISD-R)" as IPAServices

rnote over IPAServices
[1] Profile identification
[2] Verify Profile state
[3] Mark profile target Profile as 'to be disabled'
End rnote

rnote over IPAServices
[4] Generate Disable PSMO execution result data structure (disableResult)
endrnote

@enduml
```



**Figure 14 Disable Profile**

**Start Conditions:**

A Profile is already downloaded and installed in the eUICC.

A eUICC Package containing a 'Disable Profile' is received within an "ES10b.LoadEuiccPackage" by the eUICC as described in 3.3.1 Generic eUICC Package Download and Execution.

The eIM signature, the EID, and replay counter are verified as defined in 3.3.1 Generic eUICC Package Download and Execution.

**Procedure:**

If a Disable or an Enable command has already been processed in this eUICC Package, then the ISD-R SHALL indicate a failure in the PSMO execution result and procedure SHALL continue in step 4.

1. The ISD-R SHALL find the target Profile with the ICCID. If the target Profile is not found, the ISD-R SHALL indicate a failure in the PSMO execution result and procedure SHALL continue in step 4.
2. The ISD-R SHALL verify the state of the target Profile. If the target Profile is not in Enabled state, the ISD-R SHALL indicate a failure in the PSMO execution result and procedure SHALL continue in step 4.
3. The eUICC SHALL mark the target Profile "to be disabled".
4. The eUICC SHALL generate the PSMO execution result data structure indicating the result of the PSMO 'Disable Profile', and the procedure continues as described in section 3.3.1.

#### End Conditions:

The target Profile is marked "to be disabled".

The signed eUICC Package execution result data structure containing the result of disabled Profile is stored in the eUICC.

NOTE: To maintain connectivity, a Disable command needs to be followed by an Enable command in the same eUICC Package.

### 3.4.3 Delete Profile

This procedure defines the execution of a Delete command contained within a eUICC Package as defined in 3.3.1 Generic eUICC Package Download and Execution, used to delete a Profile already downloaded and installed on an eUICC.

The conditions under which a Provisioning Profile or a Test Profile could be deleted are implementation-dependent and out of the scope of this specification. The eUICC implementation MAY not support deletion of a Provisioning Profile or a preloaded Test Profile.

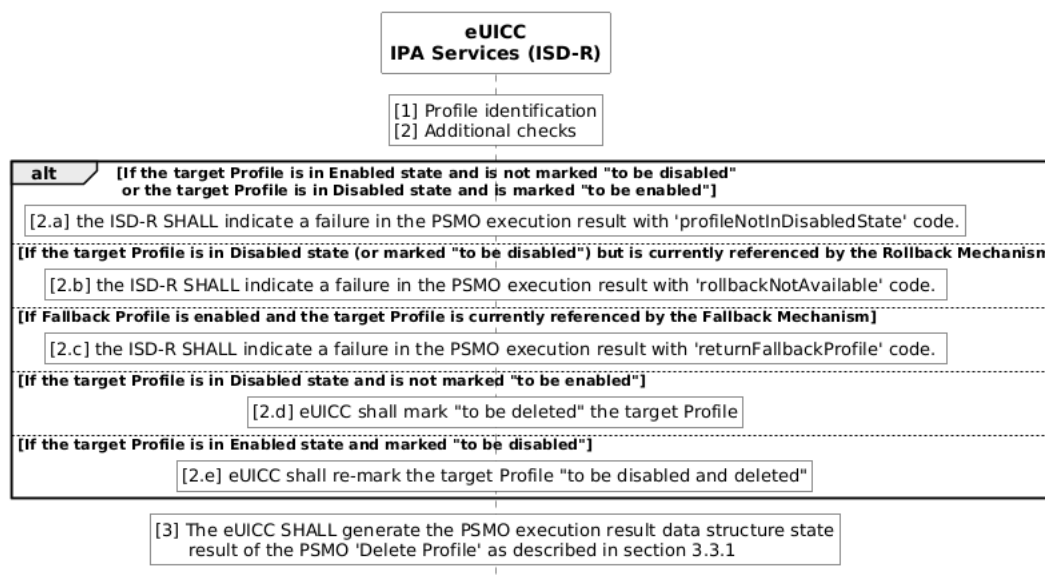
An eUICC Package SHALL NOT include an Enable command with granted usage of the Rollback Mechanism and a Delete command, where the Delete command would try to delete the Profile to which the IPA might want to roll back.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC\n<b>IPA Services (ISD-R)" as IPAServices
rnote over IPAServices
[1] Profile identification
[2] Additional checks
endrnote
    alt If the target Profile is in Enabled state and is not marked "to be disabled"\n or the target Profile is in Disabled state and is marked "to be enabled"
        rnote over IPAServices
```

```

    [2.a] the ISD-R SHALL indicate a failure in the PSMO execution result with
    'profileNotInDisabledState' code.
    endnote
    else If the target Profile is in Disabled state (or marked "to be disabled") but
    is currently referenced by the Rollback Mechanism
    rnote over IPAServices
        [2.b] the ISD-R SHALL indicate a failure in the PSMO execution result with
        'rollbackNotAvailable' code.
    endnote
    else If Fallback Profile is enabled and the target Profile is currently
    referenced by the Fallback Mechanism
    rnote over IPAServices
        [2.c] the ISD-R SHALL indicate a failure in the PSMO execution result with
        'returnFallbackProfile' code.
    endnote
    else If the target Profile is in Disabled state and is not marked "to be
    enabled"
    rnote over IPAServices
        [2.d] eUICC shall mark "to be deleted" the target Profile
    endnote
    else If the target Profile is in Enabled state and marked "to be disabled"
    rnote over IPAServices
        [2.e] eUICC shall re-mark the target Profile "to be disabled and deleted"
    endnote
end
rnote over IPAServices
[3] The eUICC SHALL generate the PSMO execution result data structure state
    result of the PSMO 'Delete Profile' as described in section 3.3.1
endnote
@enduml
    
```



**Figure 15 Delete Profile**

**Start Conditions:**

A Profile is already downloaded and installed in the eUICC. An eUICC Package containing a 'Delete Profile' is received within an "ES10b.LoadEuiccPackage" by the eUICC as described in 3.3.1 Generic eUICC Package Download and Execution.

The eIM signature the EID, and replay counter are verified as defined in 3.3.1 Generic eUICC Package Download and Execution.

### Procedure:

1. The ISD-R SHALL find the target Profile with the ICCID. If the target Profile is not found, then the ISD-R SHALL indicate a failure with error code: `iccidOrAidNotFound(1)` and procedure SHALL continue in step (3).
2. The ISD-R SHALL:
  - a. If the target Profile is in Enabled state and is not marked "to be disabled", or the target Profile is in Disabled state and is marked "to be enabled", then the ISD-R SHALL indicate a failure in the PSMO execution result with `profileNotInDisabledState(2)` error code and procedure SHALL continue in step (3).
  - b. If the target Profile is in Disabled state (or marked "to be disabled") but is currently referenced by the Rollback Mechanism (see Enable command), then the ISD-R SHALL indicate a failure in the PSMO execution result with `rollbackNotAvailable(20)` error code.

NOTE: The ISD-R does not check if the target Profile is a Fallback Profile, i.e., it is allowed to delete a Fallback Profile.

- c. If the Fallback Profile is enabled and if the target Profile is the Profile that is currently referenced by the Fallback Mechanism (the Profile that was disabled using the `ES10b.ExecuteFallbackMechanism` function, the ISD-R SHALL indicate a failure in the PSMO execution result with `returnFallbackProfile(21)` error code and procedure SHALL continue in step (3).
  - d. If the target Profile is in Disabled state and is not marked "to be enabled", then the eUICC SHALL mark the target Profile "to be deleted".
  - e. If the target Profile is in Enabled state and marked "to be disabled" and is not currently referenced by the Rollback Mechanism (see Enable command), then the eUICC SHALL mark the target Profile "to be disabled and deleted".
3. The eUICC SHALL generate the PSMO execution result data structure indicating the result of the PSMO 'Delete Profile' as described in section 3.3.1.

### End Conditions:

The target Profile is either marked as "to be deleted", marked "to be disabled and deleted" or remains in "enabled" state and a failure is indicated in the signed eUICC Package execution result.

NOTE: If the target Profile is deleted, all the associated data of this Profile is deleted.

The signed eUICC Package execution result data structure containing the result of Delete Profile is stored in the eUICC.

### 3.4.4 Configure by eIM of Immediate Profile Enabling

This procedure defines the execution of a `configureImmediateEnable` command contained within an eUICC Package as defined in 3.3.1 Generic eUICC Package Download

and Execution, used to activate or deactivate immediate Profile enabling in the eUICC and to add or update default SM-DP+ data used in the immediate Profile enabling.

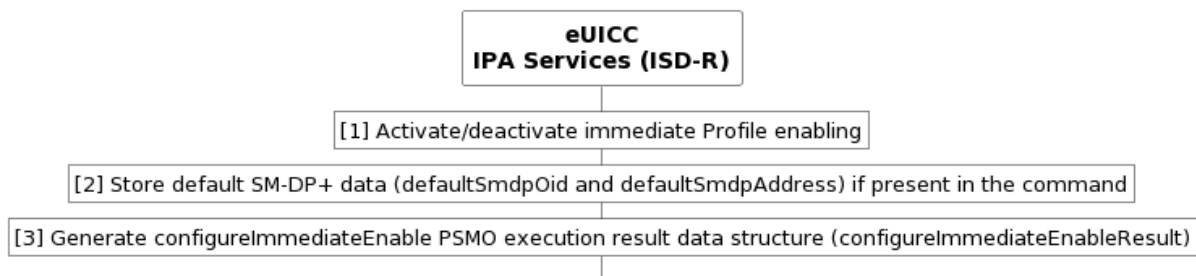
```

@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC\n<b>IPA Services (ISD-R)" as IPAServices

rnote over IPAServices
[1] Activate/deactivate immediate Profile enabling
endrnote
rnote over IPAServices
[2] Store default SM-DP+ data (defaultSmdpOid and defaultSmdpAddress) if present in the command
endrnote
rnote over IPAServices
[3] Generate configureImmediateEnable PSMO execution result data structure (configureImmediateEnableResult)
endrnote

@enduml
    
```



**Figure 16 Configuration by eIM of Immediate Profile Enabling**

**Start Conditions:**

An eUICC Package containing a 'configureImmediateEnable' command was received within an "ES10b.LoadEuiccPackage" by the eUICC.

The eIM signature, the EID, and replay counter has been verified (see 3.3.1 Generic eUICC Package Download and Execution).

**Procedure:**

1. If `immediateEnableFlag` is present and immediate Profile enabling is not activated, the ISD-R SHALL activate immediate Profile enabling. If `immediateEnableFlag` is not present and immediate Profile enabling is activated, the ISD-R SHALL deactivate immediate Profile enabling. In all other cases, the immediate Profile enabling state is left unchanged.
2. If present, the ISD-R SHALL store default SM-DP+ data (`defaultSmdpOid` and/or `defaultSmdpAddress`) for use in the immediate Profile enabling.

**NOTE:** if `defaultSmdpAddress` is stored here, then it is also used as the default SM-DP+ address in the response that is provided by the `GetEuiccConfiguredAddresses` function (described in 5.8.1 of this specification).

- The eUICC SHALL generate the PSMO execution result data structure indicating the result of the PSMO 'Configure Immediate Profile Enabling' and the procedure continues as described in section 3.3.1.

### 3.4.5 Configure by IPA of Immediate Profile Enabling

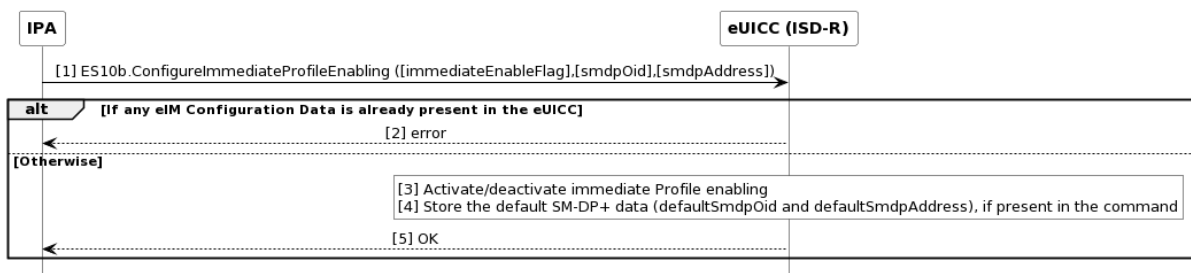
This procedure describes how to configure immediate Profile enabling when requested by IPA.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>IPA" as IPA
participant "<b>eUICC (ISD-R)" as eUICC

IPA -> eUICC : [1] ES10b.ConfigureImmediateProfileEnabling
([immediateEnableFlag],[smdpOid],[smdpAddress])
alt If any eIM Configuration Data is already present in the eUICC
eUICC -> IPA : [2] error
else Otherwise
rnote over eUICC
[3] Activate/deactivate immediate Profile enabling
[4] Store the default SM-DP+ data (defaultSmdpOid and defaultSmdpAddress), if
present in the command
end rnote
eUICC -> IPA : [5] OK
end

@enduml
```



**Figure 17 Configure by IPA of Immediate Profile Enabling**

**Start Conditions:**

The eUICC does not contain any eIM Configuration Data.

**Procedure:**

1. The IPA calls "ES10b. ConfigureImmediateProfileEnabling" function optionally comprising `immediateEnableFlag`, `defaultSmdpOid`, and `defaultSmdpAddress`.
2. If any eIM Configuration Data is already present in the eUICC, the eUICC SHALL return an error and the procedure SHALL stop. Otherwise, the procedure continues in step (3).
3. If `immediateEnableFlag` is present and immediate Profile enabling is not activated, the eUICC SHALL activate immediate Profile enabling. If `immediateEnableFlag` is not present and immediate Profile enabling is activated, the eUICC SHALL deactivate immediate Profile enabling. In all other cases, the immediate Profile enabling state is left unchanged.
4. If provided by IPA in step (1), the eUICC SHALL store default SM-DP+ data (`defaultSmdpOid` and/or `defaultSmdpAddress`) for use in the immediate Profile enabling.

NOTE: if `defaultSmdpAddress` is stored here, then it is also used as the default SM-DP+ address in the response that is provided by the `GetEuiccConfiguredAddresses` function (described in 5.8.1 of this specification).

5. The eUICC returns ok to the IPA.

NOTE: How the configuration data (`immediateEnableFlag`, `defaultSmdpOid`, and `defaultSmdpAddress`) is sent to the IP Ae is implementation specific.

### 3.4.6 Setting the Fallback Attribute for the target Profile

This procedure defines the execution of a `setFallbackAttribute` command contained within an eUICC Package as defined in 3.3.1 Generic eUICC Package Download and Execution. The command is used to set the Fallback Attribute for the target Profile.

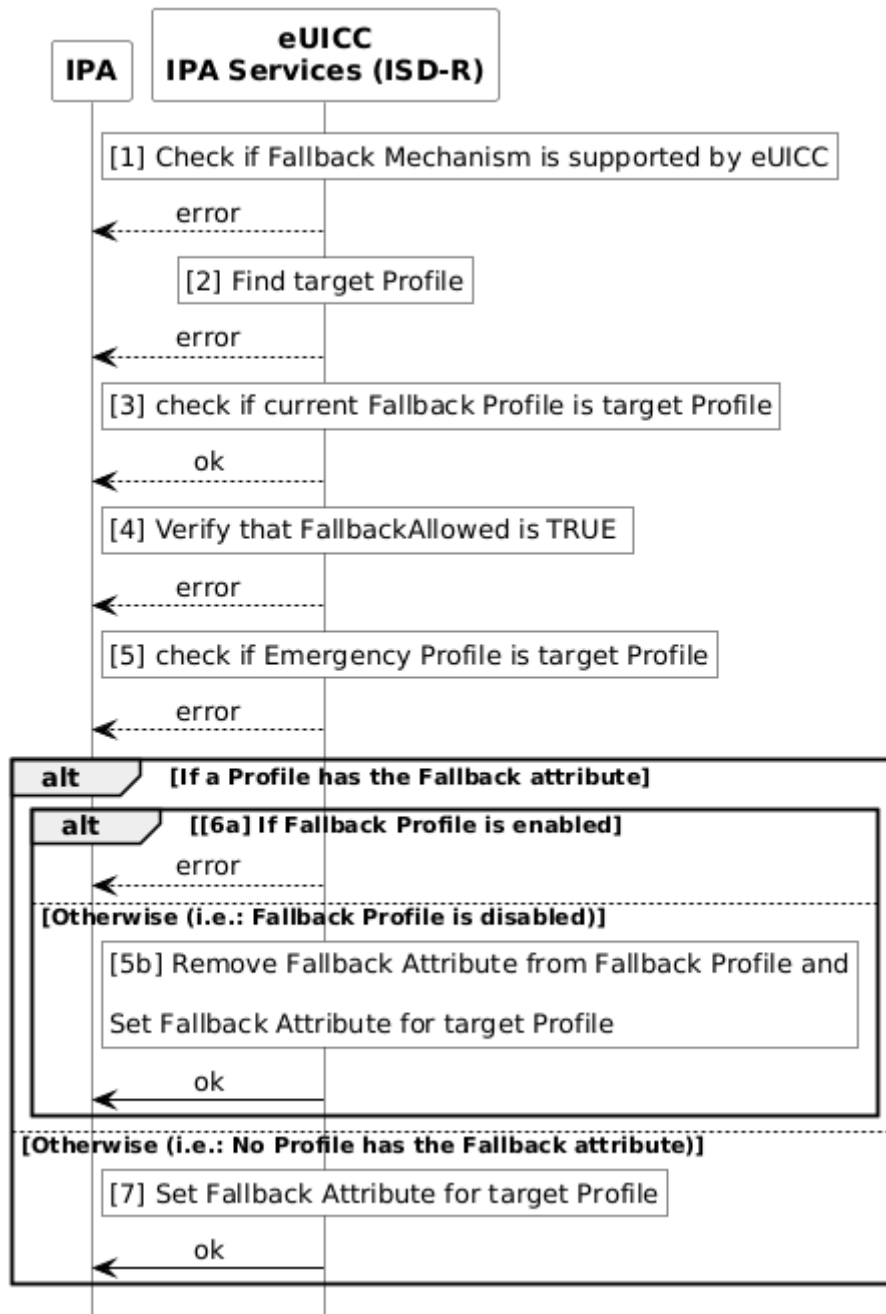
```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>IPA" as IPA
participant "<b>eUICC\n<b>IPA Services (ISD-R)" as E

rnote right IPA
    [1] Check if Fallback Mechanism is supported by eUICC
End rnote
E --> IPA: error
rnote over E
    [2] Find target Profile
End rnote
E --> IPA: error
rnote right IPA
```

```
[3] check if current Fallback Profile is target Profile
end rnote
E --> IPA: ok
rnote right IPA
  [4] Verify that FallbackAllowed is TRUE
End rnote
E --> IPA: error
rnote right IPA
  [5] check if Emergency Profile is target Profile
End rnote
E --> IPA: error
alt If a Profile has the Fallback attribute
alt [6a] If Fallback Profile is enabled
  E --> IPA: error
else Otherwise (i.e.: Fallback Profile is disabled)
  rnote right IPA
    [5b] Remove Fallback Attribute from Fallback Profile and

    Set Fallback Attribute for target Profile
  end rnote
  E -> IPA: ok
end
else Otherwise (i.e.: No Profile has the Fallback attribute)
  rnote right IPA
    [7] Set Fallback Attribute for target Profile
  end rnote
  E -> IPA: ok
end
@enduml
```



**Figure 18 Set Fallback Attribute**

**Start Conditions:**

An eUICC Package containing a 'setFallbackAttribute' command was received within an "ES10b.LoadEuiccPackage" by the eUICC.

The eIM signature and replay counter have successfully been verified (see 3.3.1 Generic eUICC Package Download and Execution).

**Procedure:**

1. If the eUICC does not support the Fallback Mechanism the procedure SHALL stop and the result of this command SHALL indicate error ('processingTerminated' with 'unknownOrDamagedCommand').
2. The ISD-R SHALL find the target Profile with the given ICCID. If the target Profile is not available, the ISD-R SHALL stop the procedure with a result indicating a failure ('iccidOrAidNotFound').
3. The ISD-R SHALL check if the target Profile already has the Fallback Attribute set, if yes, the procedure SHALL stop and the result of this command SHALL indicate success ('ok').
4. The ISD-R SHALL check if Fallback is allowed for the target Profile by verifying that the `fallbackAllowed` flag is set to TRUE. If the flag is absent or set to FALSE, the procedure SHALL stop and the result of this command SHALL indicate error ('fallbackNotAllowed(2)').
5. The ISD-R SHALL check if the target Profile is the Emergency Profile with `ecallIndication` set to TRUE. If yes, the procedure SHALL stop and the result of this command SHALL indicate error ('undefinedError(127)').
6. The ISD-R SHALL check if there is a Profile that already has the Fallback Attribute set (i.e., a Fallback Profile is present). If so, the ISD-R SHALL check the state of this Fallback Profile:
  - a. if this Fallback Profile is enabled, the procedure SHALL stop and the result of this command SHALL indicate error ('fallbackProfileEnabled(3)').
  - b. if this Fallback Profile is disabled, the ISD-R SHALL unset the Fallback Attribute from this Profile, and set the Fallback Attribute of the target Profile. After that the procedure SHALL stop and the result of this command SHALL indicate success ('ok').
7. The ISD-R SHALL set the Fallback Attribute of the Target Profile. After that the procedure SHALL stop and the result of this command SHALL indicate success ('ok').

### 3.4.7 Unsetting the Fallback Attribute for the target Profile

This procedure defines the execution of a `unsetFallbackAttribute` command contained within an eUICC Package as defined in 3.3.1 Generic eUICC Package Download and Execution. The command is used to unset the Fallback Attribute of the current Fallback Profile.

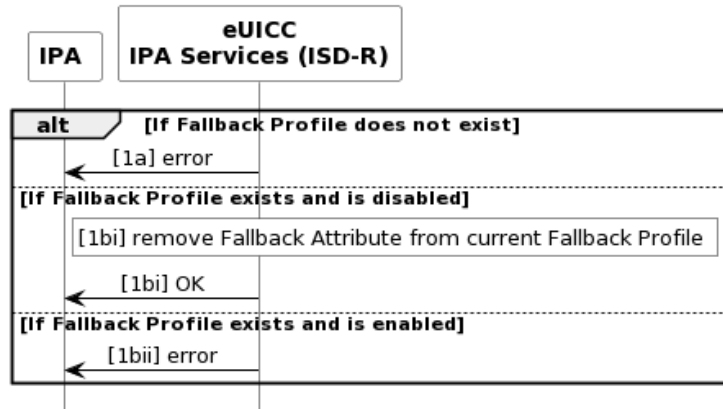
```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>IPA " as IPA
participant "<b>eUICC\n<b>IPA Services (ISD-R)" as E

alt If Fallback Profile does not exist
```

```

E -> IPA: [1a] error
else If Fallback Profile exists and is disabled
  rnote right IPA
  [1bi] remove Fallback Attribute from current Fallback Profile
end rnote
E -> IPA: [1bi] OK
else If Fallback Profile exists and is enabled
  E-> IPA: [1bii] error
end
@enduml
    
```



**Figure 19 Unset Fallback Attribute**

**Start Conditions:**

An eUICC Package containing a 'unsetFallbackAttribute' command was received within an "ES10b.LoadEuiccPackage" by the eUICC.

The eIM signature and replay counter have successfully been verified (see 3.3.1 Generic eUICC Package Download and Execution).

The Fallback Mechanism is supported by the eUICC.

**Procedure:**

1. The ISD-R SHALL check if the Fallback Attribute is set for any Profile on the eUICC and:
  - a. if not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('noFallbackAttribute')
  - b. if yes, and:
    - i. if the Fallback Profile is disabled, unset the Fallback Attribute of that Profile. After that, the procedure is completed, and the result of this command SHALL indicate success ('ok').
    - ii. if the Fallback Profile is enabled, the procedure SHALL stop and the result of this command SHALL indicate error ('fallbackProfileEnabled(3)').

**NOTE:** Upon successful completion of this command, no Fallback Profile exists on the eUICC.

### 3.5 eIM Configuration at eUICC

#### 3.5.1 eIM Configuration Data managed by eIM

This section describes the procedures related to managing eIM Configuration Data requested by the eIM. All the initiated requests by the eIM SHALL be signed by an Associated eIM and verified by the eUICC.

The eIM initiated procedures are to:

- Add a new eIM Configuration Data.
- Delete an existing eIM Configuration Data.
- Update an existing eIM Configuration Data.
- Request for a list of Associated eIMs.

##### 3.5.1.1 Addition of eIM Configuration Data

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC (ISD-R)" as eUICC

rnote over eUICC
[1] Add eIM Configuration Data to eUICC storage
end rnote

rnote over eUICC
[2] Generate addEIM eCO execution result data structure (addEimResult)
end rnote

@enduml
```

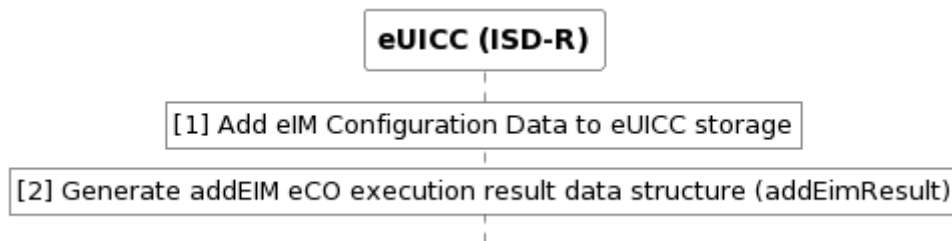


Figure 20 Addition of eIM Configuration Data

#### Start Conditions:

- A eUICC Package containing an 'addEim' was received within an "ES10b.LoadEuiccPackage" by the eUICC.
- The eUICC contains eIM Configuration Data of at least one Associated eIM and the eUICC Package was verified as defined in section 3.3.1.

#### Procedure:

1. Upon successful verifications of the eUICC Package (see section 3.3.1), the ISD-R SHALL add the eIM Configuration Data of the new eIM to its list of Associated eIMs. The eUICC SHALL stop the procedure indicating a failure if:
  - a. the eUICC does not have sufficient memory to store the new eIM ("insufficientMemory")
  - b. no eimId is provided in the eUICC Package or eimId exceeds its maximum length ("commandError")
  - c. the provided eimId belongs to an eIM already associated with the eUICC ("associatedEimAlreadyExists")
  - d. the counterValue exceeds the maximum value supported by the eUICC ("counterValueOutOfRange")
  - e. the eUICC is requested to generate an associationToken but the value provided is not set to -1 ("invalidAssociationToken")
  - f. the provided euiccCiPKId is not an entry within euiccCiPKIdListForSigning in eUICCInfo2 ("ciPKUnknown")
  - g. any other error during command execution occurs ("undefinedError").
2. The procedure continues as defined in section 3.3.1.

#### End Conditions:

- eIM Configuration Data of the new eIM is added to the list of Associated eIMs within the eUICC.
- The new eUICC is added to the list of eUICCs within the eIM.

#### 3.5.1.2 Deletion of eIM Configuration Data

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

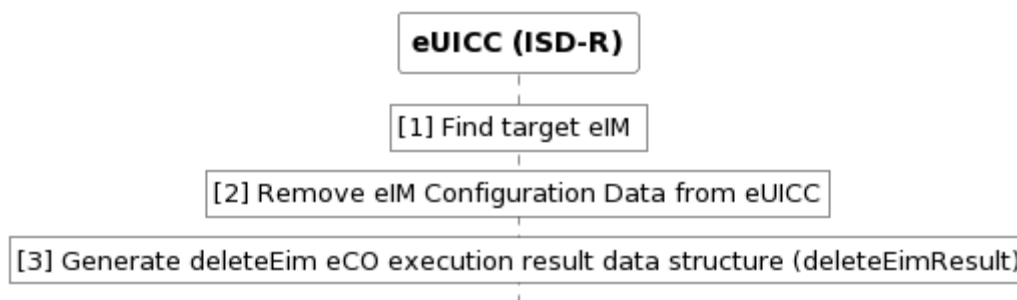
participant "<b>eUICC (ISD-R)" as eUICC

rnote over eUICC
[1] Find target eIM
end rnote

rnote over eUICC
[2] Remove eIM Configuration Data from eUICC
end rnote

rnote over eUICC
[3] Generate deleteEim eCO execution result data structure (deleteEimResult)
end rnote

@enduml
```



**Figure 21 Deletion of eIM Configuration Data**

**Start Conditions:**

- A eUICC Package containing a 'deleteEim' was received within an "ES10b.LoadEuiccPackage" by the eUICC.
- The eUICC contains eIM Configuration Data of at least one Associated eIM and the verifications as defined in section 3.3.1 were performed.

**Procedure:**

1. Upon successful verifications of the eUICC Package (see section 3.3.1), the ISD-R SHALL find the eIM Configuration Data of the target eIM in its list of Associated eIMs. The eUICC SHALL stop the procedure indicating a failure if:
  - a. the eUICC does not find the `eimId` within its list of Associated eIMs ("eimNotFound")
  - b. in case no `eimId` is provided in the eUICC Package ("commandError")
  - c. any other error during command execution occurs ("undefinedError").
2. The eUICC SHALL remove the eIM Configuration Data from its list of Associated eIMs. If the target Associated eIM to be deleted is the requesting eIM, the eUICC SHALL include the relevant information into the eCO execution result and SHALL only delete the eIM Configuration Data after the eCO execution result was generated and signed. If the target Associated eIM to be deleted is the last available Associated eIM within the eUICC, the eUICC SHALL set its state such, that it allows for addition of eIM Configuration Data using Unsigned eCO and SHALL return "lastEimDeleted" within its eCO execution result.
3. The procedure continues as defined in section 3.3.1.

**End Conditions:**

- eIM Configuration Data of the target eIM is deleted from the list of Associated eIMs within the eUICC.

**3.5.1.3 Update of eIM Configuration Data**

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
```

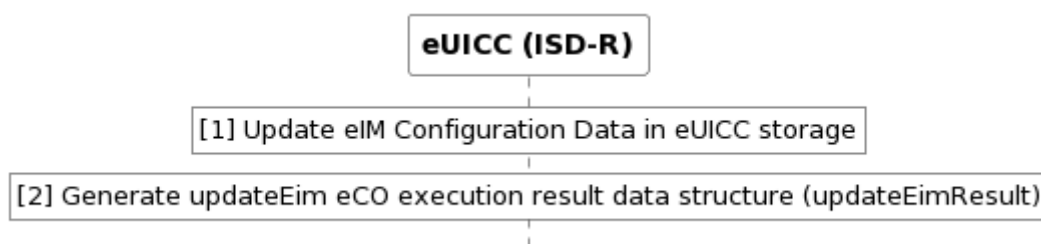
```
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC (ISD-R)" as eUICC

rnote over eUICC
[1] Update eIM Configuration Data in eUICC storage
end rnote

rnote over eUICC
[2] Generate updateEim eCO execution result data structure (updateEimResult)
end rnote

@enduml
```



**Figure 22 Update of eIM Configuration Data**

**Start Conditions:**

- A eUICC Package containing an 'updateEim' was received within an "ES10b.LoadEuiccPackage" by the eUICC.
- The eUICC contains eIM Configuration Data of at least one Associated eIM and the eUICC Package was verified as defined in section 3.3.1.

**Procedure:**

1. Upon successful verifications of the eUICC Package (see section 3.3.1), the ISD-R SHALL update the eIM Configuration Data of the eIM addressed by the provided eimId. The eUICC SHALL stop the procedure indicating a failure if:
  - a. the eIM cannot be found in the list of Associated eIMs or no eimId is provided in the eUICC Package ("eimNotFound")
  - b. the associationToken is included in the EimConfigurationData data object ("commandError")
  - c. none of counterValue , eimPublicKeyData, eimIdType, eimFqdn, eimSupportedProtocol, trustedPublicKeyDataTls or euiccCiPKId is included in the EimConfigurationData data object ("commandError")
  - d. the counterValue is updated to a value lower than the currently stored value while the eimPublicKeyData are not updated ("commandError").
  - e. the provided euiccCiPKId is not an entry within euiccCiPKIdListForSigning in eUICCInfo2 ("ciPKUnknown")
  - f. any other error during command execution occurs ("undefinedError")

2. The procedure continues as defined in section 3.3.1.

**End Conditions:**

- The eIM Configuration Data of the targeted eIM is updated within the list of Associated eIMs within the eUICC.

**3.5.1.4 Request for a list of Associated eIMs**

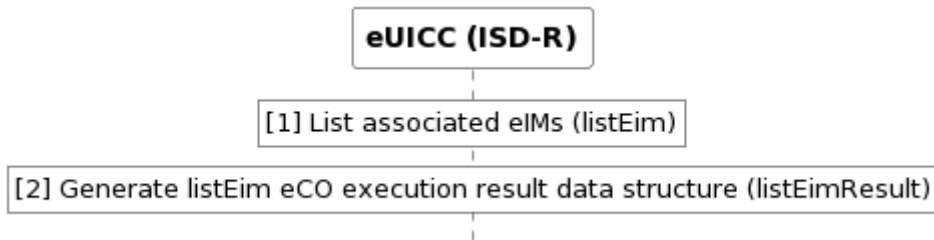
```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC (ISD-R)" as eUICC

rnote over eUICC
[1] List associated eIMs (listEim)
end rnote

rnote over eUICC
[2] Generate listEim eCO execution result data structure (listEimResult)
end rnote

@enduml
```



**Figure 23 Request for a list of Associated eIMs**

**Start Conditions:**

- A eUICC Package containing a 'listEim' was received within an "ES10b.LoadEuiccPackage" by the eUICC.
- The eUICC contains eIM Configuration Data of at least one Associated eIM and the eUICC Package was verified as defined in section 3.3.1.

**Procedure:**

1. Upon successful verifications of the eUICC Package (see section 3.3.1), the ISD-R SHALL provide as a result the eIM identifier(s) of Associated eIM(s).
2. The procedure continues as defined in section 3.3.1.

**End Conditions:**

- The eIM identifier of Associated eIM(s) can be provided through the signed eUICC Package Result to the IPA and via eIM Package Result to the eIM.

### 3.5.2 eIM Configuration Data managed by IPA

This section describes the procedures related to managing eIM Configuration Data requested by the IPA. All the initiated requests by the IPA SHALL be protected as per the ES10 security.

The IPA initiated procedures are to:

- Add of an initial eIM Configuration Data.
- Complete removal of eIM Configuration Data.

NOTE: The protection of IPA initiated request is implementation specific under the responsibility of IoT Device manufacturer.

#### 3.5.2.1 Addition of initial eIM Configuration Data

```

@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>IPA" as IPA
participant "<b>eUICC (ISD-R)" as eUICC

IPA -> eUICC : [1] ES10b.AddInitialEim(EimConfigurationData)
alt If any eIM Configuration Data is already present in the eUICC
eUICC --> IPA : [2] error (associatedEimAlreadyExist)
else Otherwise continue processing the command
alt Any of the subsequent checks fails
eUICC --> IPA : [3] error (one of: insufficientMemory, invalidAssociationToken,
counterValueOutOfRange, commandError, ciPKUnknown, undefinedError)
else Otherwise
note over eUICC
[4] Store eIM Configuration Data
end note
eUICC --> IPA : [5] AddInitialEimResponse
end

@enduml
    
```

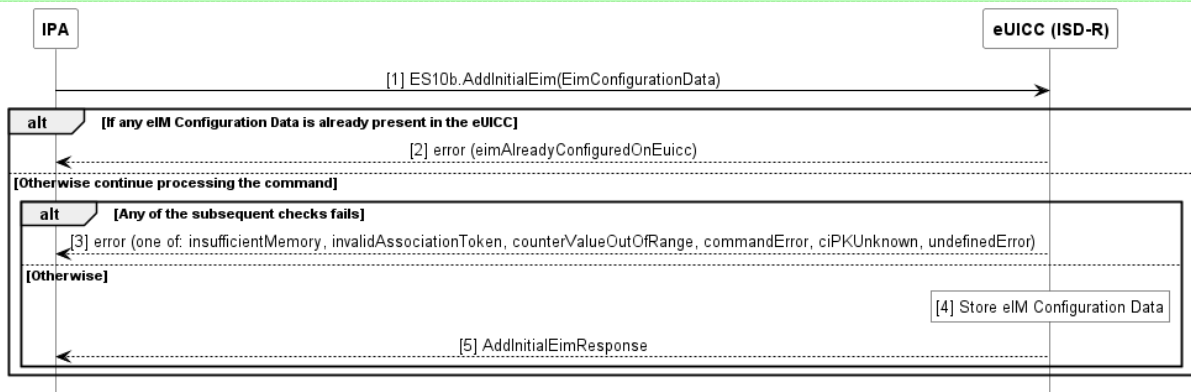


Figure 24 Addition of initial eIM Configuration Data

Start Conditions:

- The eUICC does not contain any eIM Configuration Data.

### Procedure:

1. The IPA calls "ES10b.AddInitialEim" function comprising eIM Configuration Data (EimConfigurationData).
2. If any eIM Configuration Data is already present in the eUICC, the eUICC SHALL return an error ("associatedEimAlreadyExists") and the procedure SHALL stop.
3. The eUICC SHALL stop the procedure indicating a failure if:
  - a. the counterValue exceeds the maximum value supported by the eUICC ("counterValueOutOfRange").
  - b. the eUICC is requested to generate an associationToken but the value provided is not set to -1 ("invalidAssociationToken")
  - c. the eUICC does not have sufficient memory to store the initial eIM ("insufficientMemory")
  - d. no eimId is provided or eimId exceeds its maximum length ("commandError")
  - e. the provided euiccCiPKId is not an entry within euiccCiPKIdListForSigning in eUICCInfo2 ("ciPKUnknown")
  - f. any other error during command execution occurs ("undefinedError").
4. If requested, the eUICC SHALL calculate the association token and store it together with the eIM Configuration Data.
5. The eUICC returns the AddInitialEimResponse to the IPA.

### End Conditions:

- The eUICC is associated with the eIM as per the eIM Configuration Data.

NOTE: In case IP Ae is used, the addition of initial eIM Configuration Data uses the EimConfigurationData structure. How the eIM Configuration Data is sent to the eUICC is implementation specific.

### 3.5.2.2 Deletion of all eIM Configuration Data

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>IPA" as IPA
participant "<b>eUICC (ISD-R)" as eUICC

IPA -> eUICC : [1] ES10b.EuiccMemoryReset(resetEimConfigData)
rnote over eUICC
[2] Removes all eIM Configuration Data
end rnote
eUICC --> IPA : [3] ok or nothingToDelete

@enduml
```



**Figure 25 Deletion of all eIM Configuration Data**

**Start Conditions:**

- None

**Procedure:**

1. The IPA calls "ES10b.EuiccMemoryReset" function with `resetEimConfigData` bit set to one.
2. The eUICC removes all eIM Configuration Data.
3. The eUICC returns `ok` or `nothingToDelete`.

**End Conditions:**

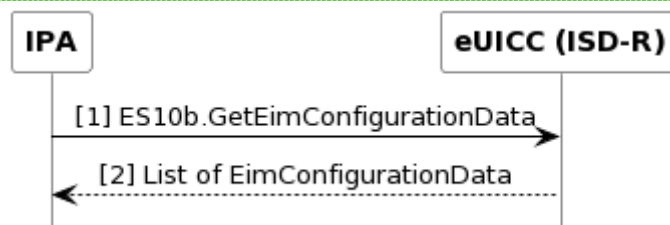
- The eUICC does not contain any eIM Configuration Data.

**3.5.2.3 Reading eIM Configuration Data**

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>IPA" as IPA
participant "<b>eUICC (ISD-R)" as eUICC

IPA ->> eUICC : [1] ES10b.GetEimConfigurationData
eUICC --> IPA : [2] List of EimConfigurationData
@enduml
```



**Figure 26 Reading eIM Configuration Data by IPA**

**Start Conditions:**

- None

**Procedure:**

1. The IPA calls "ES10b.GetEimConfigurationData" function.
2. The eUICC returns the list of eIM Configuration Data stored in the eUICC to the IPA.

**End Conditions:**

- None

### 3.6 Error handling within an RSP session

The RSP sessions comprise a sequence of operations between the SM-DP+, the IPA, the eUICC, and/or the eIM over a period of time. In addition to errors reported by ES9+, ES9+', ES10 and/or ESipa functions, other conditions MAY impact the successful execution of this procedure. The IPA SHOULD indicate such failures to external entities, e.g., a device management platform; however, the specific presentation of these errors is out of the scope of this document.

The IPA SHOULD NOT initiate a new RSP session while there is an active RSP session. However, in the event that this does occur, the eUICC SHALL discard its session state with the possible exception that unused one-time keys MAY be stored for future retries, when a new RSP session is started with "ES10b.GetEUICCChallenge".

If an eUICC Memory Reset or eUICC Test Memory Reset is successfully processed during an RSP session, the eUICC SHALL discard its session state.

An RSP Session MAY fail because of a communications failure between the IPA and the RSP Server, or the IPA and the eIM. The IPA MAY retry the failed RSP session for a period of time. The IPA SHALL reset its own session state when all retry attempts have failed.

An RSP Session could fail while the IPA is invoking an ES10 function for reasons other than an error status reported by the eUICC. Examples of such failures include:

- In the case of a removable eUICC card, the card is unexpectedly removed.
- The IoT Device is powered off, e.g., by running out of the battery or by the user.
- A software fault could cause a crash of the IPA, host IoT Device, and/or baseband processor.

When possible (e.g., when power is restored), the IPA SHOULD provide an appropriate error indication to external entities, e.g., the device management platform, and MAY restart the relevant procedure. The specific presentation of such an error Notification is out of scope of this document.

### 3.7 Notification Delivery to Notification Receivers

This procedure describes the delivery of Notifications to Notification Receivers. Notifications MAY also be delivered together with an eUICC Package Result as described in Section 3.3.1.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
```

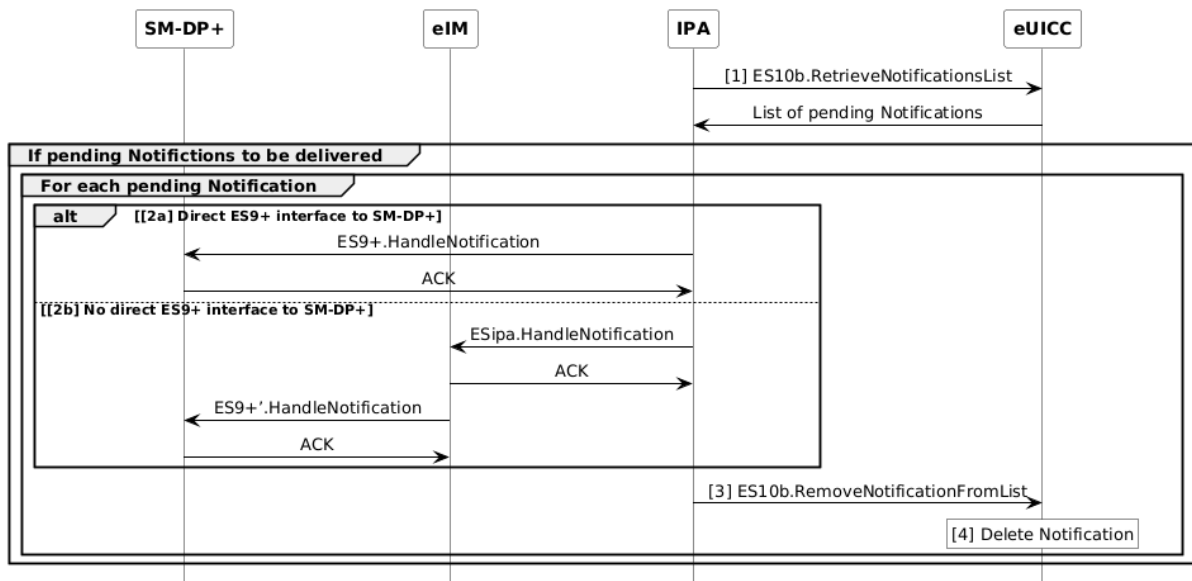
```

skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
skinparam ParticipantPadding 70
hide footbox

participant "<b>SM-DP+</b>" as DP
participant "<b>eIM</b>" as eIM
participant "<b>IPA</b>" as IPA
participant "<b>eUICC</b>" as E

IPA -> E : [1] ES10b.RetrieveNotificationsList
E -> IPA : List of pending Notifications

group If pending Notifications to be delivered
group For each pending Notification
alt [2a] Direct ES9+ interface to SM-DP+
IPA -> DP : ES9+.HandleNotification
DP -> IPA : ACK
else [2b] No direct ES9+ interface to SM-DP+
IPA -> eIM : ESipa.HandleNotification
eIM -> IPA : ACK
eIM -> DP : ES9+'.HandleNotification
DP -> eIM : ACK
end
IPA -> E : [3] ES10b.RemoveNotificationFromList
note over E: [4] Delete Notification
end
end
@enduml
    
```



**Figure 27 Notification Delivery to Notification Receivers**

**Start Condition:**

This procedure requires the following start conditions:

- A Secure Connection between the eIM and the IPA is established.
- Either a profile installation or a PSMO has been executed such that one or more Notifications MAY have been generated by the eUICC.

- IPA is configured to send Notifications using ES9+.HandleNotification or ESipa.HandleNotification.

**Procedure:**

1. The IPA SHALL retrieve pending Notifications by calling the ES10b.RetrieveNotificationsList function.
2. In case of a non-empty list of pending Notifications, IPA SHALL deliver the pending Notifications to the Notification Receivers:
  - a. If a direct ES9+ interface to the SM-DP+ is used to deliver Notifications, the IPA SHALL send the Notifications by using the ES9+.HandleNotification function as described in section 5.6.4 of SGP.22 [4].
  - b. If Notifications are delivered via the eIM to the SM-DP+, the IPA SHALL send the Notifications to the eIM by using the ESipa.HandleNotification function as described in section 5.14.7 of this document and the eIM SHALL forward them to the Notification Receivers by using the ES9+'.HandleNotification function as described in section 5.7.4 of this document. In case of an IPA with IPA Capability `minimizeEsipaBytes` the IPA SHOULD send Notifications in compact format as described in Section 5.14.7. An eIM receiving a pending Notification in compact format SHALL identify the relevant information and build the full pending Notification before forwarding the Notification to the Notification Receiver.
3. IPA SHALL call ES10b.RemoveNotificationFromList (see SGP.22 [4]) to delete the pending Notification for which acknowledgement has been received.
4. The eUICC SHALL delete the pending Notification.

### **3.8 IoT Device and eUICC Initialisation**

The IoT Device and eUICC SHALL be initialised according to sections 3.8.1, 3.8.2, 3.8.3 and 3.8.4 of this specification.

#### **3.8.1 eUICC Initialisation**

The eUICC SHALL be initialised according to sections 3.4.1 of SGP.22 [4].

#### **3.8.2 IoT Device Capabilities**

The eUICC SHALL request the IoT Device to send the Terminal Capability command by setting the related bit in the file control parameters of the MF.

The IoT Device SHALL report its support of IPA functions using the Terminal Capability command data defined in ETSI TS 102 221 [34]. This command SHALL be sent before the SELECT ISD-R command defined in section 3.8.4 of this specification.

Within the Terminal Capability template (tag 'A9'), the tag '84' is used for indicating the IoT Device's support for eUICC related functions.

The IPAd support is indicated in the first byte within the TLV object under tag '84':

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
-	-	-	-	-	-	-	1	IPAd supported
-	-	-	-	-	-	-	0	IPAd not supported
x	x	x	x	x	x	x	-	Not applicable

**Table 6 eUICC-related IoT Device Capabilities**

If an IoT Device supports IPAd, b1 SHALL be set to 1. Otherwise, b1 SHALL be set to 0.

The eUICC SHALL only enable the ES10 functions if the IoT Device indicates support for IPAd.

The conditions for activating the IP Ae are defined in section 3.8.4.

### 3.8.3 eUICC File Structure

The eUICC File structure SHALL be initialised according to sections 3.4.3 of SGP.22 [4].

### 3.8.4 ISD-R Selection and IP Ae Activation

Before sending any command to the eUICC, the IPAd SHALL establish a logical channel and select the ISD-R.

The opening of the logical channel and the selection of the ISD-R SHALL be done explicitly using, respectively, the MANAGE CHANNEL command and the SELECT command defined in GlobalPlatform Card Specification [32]. This MANAGE CHANNEL and SELECT commands can be intrinsically used via a dedicated IoT Device Operating System API (e.g., OMAPI defined by GlobalPlatform [33] if provided).

The IoT Device SHALL ensure that only the IPAd, but no other application on the IoT Device, is permitted to select the ISD-R, except that the ISD-R MAY also be selected during eUICC initialisation as defined in section 3.8.1.

In order to provide information about the capabilities supported by the eUICC at an early point in time, additional information is provided by the ISD-R.

On the reception of the SELECT ISD-R Command, the following data SHALL be returned within the FCI template after the objects defined in GlobalPlatform Card Specification [32]:

```
-- ASN1START
ISDRProprietaryApplicationTemplateIoT ::= [PRIVATE 1] SEQUENCE { -- Tag 'E1'
    euiccConfiguration BIT STRING {
        ipaeSupported(0), -- IPA in the eUICC (IP Ae) supported
        enabledProfile(1) -- eUICC contains an Enabled Profile
    }
}
-- ASN1STOP
```

The eUICC SHALL indicate its support for IP Ae by setting the `ipaeSupported` bit in the `euiccConfiguration` field.

If the IoT Device supports the requirements on IP Ae as defined in Annex A.2 and if the eUICC indicated support for IP Ae in the `ISDRProprietaryApplicationTemplateIoT`, the IoT Device MAY activate the IP Ae by sending an `IpaeActivationRequest` to the ISD-R. The

`IpaeActivationRequest` SHALL be sent to the ISD-R using the transport mechanism defined in section 5.7.2 of SGP.22 [4].

The command data SHALL be coded as follows:

```
-- ASN1START
IpaeActivationRequest ::= [66] SEQUENCE { -- Tag 'BF42'
  ipaeOption BIT STRING {
    activateIpae(0) -- IP Ae activation
  }
}
-- ASN1STOP
```

The response data SHALL be coded as follows:

```
-- ASN1START
IpaeActivationResponse ::= [66] SEQUENCE { -- Tag 'BF42'
  ipaeActivationResult INTEGER {ok(0), notSupported(1)}
}
-- ASN1STOP
```

If the IoT Device indicates in the Terminal Capability command the support for IPAd (see section 3.8.2) and the IPAd does not send an `IpaeActivationRequest`, the eUICC SHALL NOT activate the IP Ae. If the IoT Device indicates that IPAd is not supported, the eUICC SHALL activate the IP Ae, if supported by the eUICC.

NOTE: For IoT Devices and eUICC that support both IP Ae and IPAd, the IoT Device can reset the eUICC and send new Terminal Capability indicating that IPAd is supported to activate the IPAd in case the IP Ae is activated.

### 3.8.5 Triggering of the IP Ae for eIM Package retrieval

It SHALL be possible to trigger the IP Ae to establish a connection to the eIM using at least one of these CAT mechanisms supported by the IoT Device (see also Annex A.2):

- STATUS commands
- Location status event
- Timer expiration command

Additionally, it MAY be possible to trigger the IP Ae using other mechanisms out of the scope of this specification.

## 3.9 SM-DS Use

### 3.9.1 Event Registration

Event Registration SHALL follow the procedure defined in section 3.6.1 of SGP.22 [4].

### 3.9.2 Event Retrieval

#### 3.9.2.1 Event Retrieval by the IPA

The Event Retrieval by the IPA SHALL follow the procedure defined in section 3.6.2 of SGP.22 [4]. The IPA plays the role of the LPA.

#### 3.9.2.2 Event Retrieval by the eIM

This section describes the indirect Event Retrieval from SM-DS to eIM.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>SM-DS" as DS
participant "<b>eIM" as EIM
participant "<b>IPA" as IPA
participant "<b>eUICC" as E

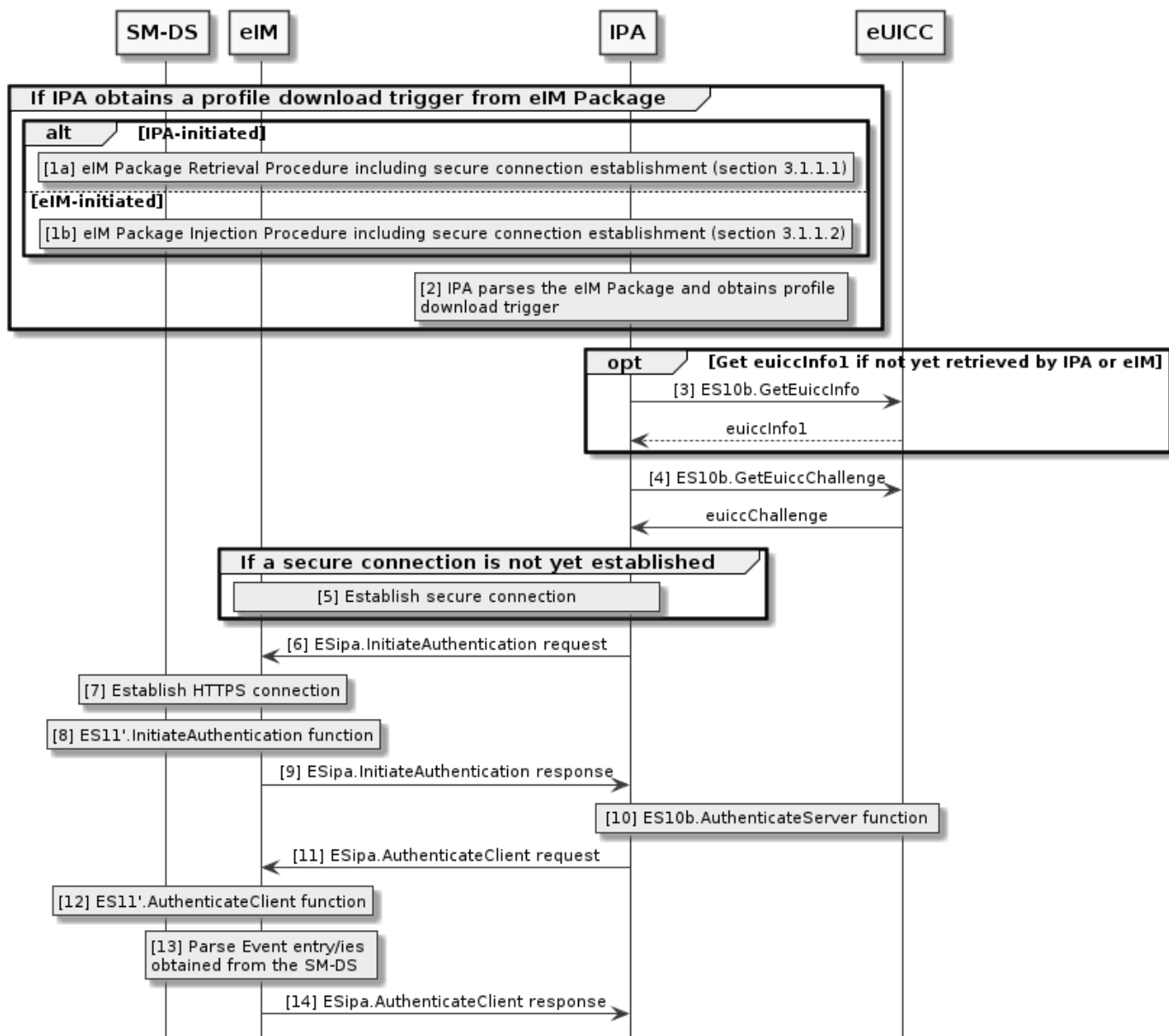
group If IPA obtains a profile download trigger from eIM Package
alt IPA-initiated
rnote over EIM, IPA : [1a] eIM Package Retrieval Procedure including secure
connection establishment (section 3.1.1.1)
else eIM-initiated
rnote over EIM, IPA : [1b] eIM Package Injection Procedure including secure
connection establishment (section 3.1.1.2)
end

rnote over IPA : [2] IPA parses the eIM Package and obtains profile \ndownload
trigger
end

opt Get euiccInfo1 if not yet retrieved by IPA or eIM
IPA -> E : [3] ES10b.GetEuiccInfo
E --> IPA : euiccInfo1
end
IPA -> E : [4] ES10b.GetEuiccChallenge
E -> IPA : euiccChallenge

group If a secure connection is not yet established
rnote over EIM, IPA : [5] Establish secure connection
end
IPA -> EIM : [6] ESipa.InitiateAuthentication request
rnote over EIM, DS : [7] Establish HTTPS connection
rnote over EIM, DS : [8] ES11'.InitiateAuthentication function
EIM -> IPA : [9] ESipa.InitiateAuthentication response
rnote over IPA, E: [10] ES10b.AuthenticateServer function
IPA -> EIM : [11] ESipa.AuthenticateClient request
rnote over EIM, DS : [12] ES11'.AuthenticateClient function
rnote over EIM : [13] Parse Event entry/ies \nobtained from the SM-DS
EIM -> IPA : [14] ESipa.AuthenticateClient response

@enduml
```



**Figure 28 Event Retrieval by the eIM**

**Start Conditions:**

In addition to the start conditions required by the Common Mutual Authentication procedure defined in section 3.1.2 of SGP.22 [4], this procedure requires the following start condition:

- The eIM and/or the IPA has the SM-DS Address.

**Procedure:**

1. In case a Profile download trigger from an eIM Package: a secure connection is established between the IPA and the eIM for IPA to obtain the eIM Package. How this is triggered is out of scope of this specification. There are two options for the delivery of the eIM Package to the IPA:

- a) The eIM Package Retrieval Procedure defined in section 3.1.1.1 is executed between the IPA and the eIM.
  - b) The eIM Package Injection Procedure defined in section 3.1.1.2 is executed between the IPA and the eIM.
2. The IPA parses the eIM Package received in step 1. The IPA identifies that the eIM Package contains a `ProfileDownloadTriggerRequest`. To trigger Event Retrieval from the SM-DS, the `ProfileDownloadTriggerRequest` is either an empty data object (i.e., without `profileDownloadData`) or contains a `profileDownloadData` with `contactSmids`. In the first case the IPA is not expected to provide SM-DS FQDN in step (6) whereas in the latter case the IPA is expected to provide SM-DS FQDN in step (6). If the format of `ProfileDownloadTriggerRequest` is invalid, or data needed by IPA for the Event Retrieval is missing, the procedure SHALL stop.
  3. If `euiccInfo1` is not yet retrieved by the eIM or IPA, the IPA MAY request `euiccInfo1` from the eUICC by calling `ES10b.GetEuiccInfo` (see section 5.9.2).
  4. The IPA requests an eUICC Challenge from the eUICC by calling the `ES10b.GetEUICCChallenge` function (see section 5.9.3).
  5. A secure connection is established between the IPA and the eIM if it is not yet established.
  6. The IPA calls `ESipa.InitiateAuthentication` function comprising eUICC Challenge, optionally SM-DS FQDN, and optionally `euiccInfo1` (see IPA Capability `minimizeEsipaBytes`). If the IPA does not provide SM-DS FQDN or `euiccInfo1`, the eIM SHALL identify the relevant information.
  7. The eIM establishes an HTTPS connection with the SM-DS in server authentication mode according to SGP.22 [4].
  8. The eIM and the SM-DS process `ES11'.InitiateAuthentication` function comprising eUICC Challenge, SM-DS FQDN, and `euiccInfo1` as defined in section 5.11.1. If the SM-DS FQDN provided in `ES11'.InitiateAuthentication` did not come from IPA (was not obtained from IPA in step 6 e.g., due to IPA has capability `eimDownloadDataHandling`), the eIM SHALL verify that this SM-DS FQDN matches the SM-DS FQDN returned by the SM-DS (in `serverSigned1`). If not, the procedure SHALL stop.
  9. The eIM sends the `ESipa.InitiateAuthentication` response to the IPA, based on the `ES11'.InitiateAuthentication` response from the SM-DS. The eIM SHALL additionally provide `ctxParams1` as defined in SGP.22 [4] if the IPA does not generate it. (see IPA capability `eimCtxParams1Generation`).
  10. The IPA and eUICC process `ES10b.AuthenticateServer` function call as defined in section 5.7.13 of SGP.22 [4].
  11. The IPA calls `ESipa.AuthenticateClient` function comprising `euiccSigned1` or `compactEuiccSigned1`, `euiccSignature1`, optionally `CERT.EUM.ECDSA`, and

optionally CERT.EUICC.ECDSA. If the IPA does not provide CERT.EUM.ECDSA and CERT.EUICC.ECDSA and/or provide the compactEuiccSigned1 instead of euiccSigned1 (see IPA Capability `minimizeEsipaBytes`), the eIM SHALL identify the relevant information and build euiccSigned1.

12. The eIM and SM-DS process the ES11'.AuthenticateClient function as defined in section 5.11.2.
13. The eIM parses the Event entries obtained from the ES11'.AuthenticateClient from the SM-DS.
14. The eIM sends the ESipa.AuthenticateClient response to the IPA, based on the ES11'.AuthenticateClient response from the SM-DS. In case of a successful ES11'.AuthenticateClient response the eIM sends only the transactionId (without including eventEntries). The eIM MAY include a ProfileDownloadTriggerRequest data object in order to trigger the IPA to start downloading a new Profile. In case of an error, IPA SHALL trigger the cancellation of the on-going RSP session (see section 3.2.3.3) with reason undefinedReason and the procedure SHALL stop.

### 3.9.3 Event Deletion

Event Deletion SHALL follow the procedure defined in section 3.6.3 of SGP.22 [4].

## 4 Data Elements

### 4.1 IPA Capabilities

The IPA Capabilities are intended for the eIM for the functional split related to profile download between the eIM and the IPA and the use of compact data object structures in ESipa functions. In addition, IPA Capabilities MAY indicate the support of eIM Package transfer including the type of connection over ESipa.

IPA Capabilities SHALL include:

- Support for direct profile download
- Support for indirect profile download
  - Support for eIM handled download data (e.g. Activation Code) during indirect profile download.
  - Support for eIM generated ctxParams1 during indirect profile download.
  - Support for eIM performed Profile Metadata verification during indirect profile download.
  - Use of compact data objects to minimize number of transferred ESipa bytes during indirect profile download.

IPA Capabilities MAY include support for eIM Package transfer over ESipa:

- eIM Package retrieval using HTTPs over TCP (see section 3.1.2.1)
- eIM Package retrieval using CoAP/DTLS over UDP (see section 3.1.2.2)
- eIM Package injection using HTTPs over TCP (see section 3.1.2.1)
- eIM Package injection using CoAP/DTLS over UDP (see section 3.1.2.2)
- proprietary

IPA Capabilities data object structure is defined as follows:

```
-- ASN1START
IpaCapabilities ::= SEQUENCE {
    ipaFeatures [0] BIT STRING {
        directRspServerCommunication (0),
        indirectRspServerCommunication (1),
        eimDownloadDataHandling (2),
        eimCtxParams1Generation (3),
        eimProfileMetadataVerification (4),
        minimizeEsipaBytes (5)
    },
    ipaSupportedProtocols [1] BIT STRING {
        ipaRetrieveHttps(0),
        ipaRetrieveCoaps(1),
        ipaInjectHttps(2),
        ipaInjectCoaps(3),
        ipaProprietary(4)
    } OPTIONAL,
    eSipaProprietaryProtocolInformation [2] VendorSpecificExtension OPTIONAL --
Additional information for proprietary protocol(s)
}
-- ASN1STOP
```

ipaFeatures indicates the features supported by the IPA:

- The `directRspServerCommunication` set to 1 indicates that IPA supports direct Profile download according to Section 3.2.3.1. This includes direct delivery of Notifications to the SM-DP+.
- The `indirectRspServerCommunication` set to 1 indicates that IPA supports indirect Profile download according to Section 3.2.3.2. This includes delivery of Notifications via the eIM to SM-DP+.
- An IPA supporting indirect profile download MAY further have the following capabilities:
  - The `eimDownloadDataHandling` set to 1 indicates that IPA does not handle download data such as Activation Codes, SM-DP+ data retrieved from SM-DS, and default SM-DP+ address during indirect Profile download and expects Activation Codes and Event Records to remain at the eIM where the eIM parses and extracts the relevant data such as SM-DP+ address, matchingId, and SM-DP+ OID for use in ESipa.InitiateAuthentication.

- The `eimCtxParams1Generation` set to 1 indicates that IPA does not support generation of `CtxParams1` and expects the eIM to generate and provide `CtxParams1` to IPA as part of `ESipa.InitiateAuthentication`.
- The `eimProfileMetadataVerification` set to 1 indicates that IPA does not support verification of Profile Metadata and expects the eIM to verify Profile Metadata as part of `ESipa.AuthenticateClient`. It is also expected that eIM checks as part of `ESipa.GetBoundProfilePackage` that the Profile Metadata provided in the BPP matches the Profile Metadata in `ESipa.AuthenticateClient`.
- The `minimizeEsipaBytes` set to 1 indicates that IPA uses compact data objects and avoids sending data over ESipa that the other party already knows in order to minimize the number of transmitted bytes between IPA and eIM. It is expected that the eIM is able to identify missing parts (bytes) and rebuild full data object structures. The eIM MAY leverage pre-configured static eUICC data, eUICC data obtained from another IoT Device within the same batch for which the data is common, eUICC data read from the IoT Device, and session data obtained from the SM-DP+. For more information refer to the profile download related ESipa function descriptions in Section 5.14.

`ipaSupportedProtocols` indicates which eIM Package transfer (see 3.1.1) is supported by the IPA:

- `ipaRetrieveHttps` set to 1 indicates support for eIM Package retrieval (see section 3.1.1.1) using HTTPs over TCP as described in section 3.1.2.1
- `ipaRetrieveCoaps` set to 1 indicates support for eIM Package retrieval (see section 3.1.1.1) using CoAP/DTLS over UDP as described in section 3.1.2.2
- `ipaInjectHttps` set to 1 indicates support for eIM Package injection (see section 3.1.1.2) using HTTPs over TCP as described in section 3.1.2.1
- `ipaInjectCoaps` set to 1 indicates support for eIM Package injection (see section 3.1.1.2) using CoAP/DTLS over UDP as described in section 3.1.2.2
- `ipaProprietary` set to 1 indicates support of a transport protocol not described in section 3.1.2.

The eIM SHALL obtain IPA Capabilities to determine whether it is able to support profile download and profile management for the particular IoT Device/IPA. IPA Capabilities MAY be obtained out-of-band from the IoT Device owner/user, or it MAY be requested from IPA in an eIM Package.

`eSipaProprietaryProtocolInformation` provides additional IPA Capabilities information related to the additional proprietary supported protocol(s). This field MAY be provided when `ipaProprietary` is set to 1. This field SHALL NOT be provided if `ipaProprietary` is not set.

## 4.2 eIM Supported Protocol

`EimSupportedProtocol` indicates which transport protocol, and eIM Package transfer method is supported by the eIM over ESipa. `eimRetrieveHttps` set to 1 indicates support for eIM Package retrieval (see section 3.1.1.1) using HTTPs over TCP as described in section 3.1.2.1

- `eimRetrieveCoaps` set to 1 indicates support for eIM Package retrieval (see section 3.1.1.1) using CoAP/DTLS over UDP as described in section 3.1.2.2
- `eimInjectHttps` set to 1 indicates support for eIM Package injection (see section 3.1.1.2) using HTTPs over TCP as described in section 3.1.2.1
- `eimInjectCoaps` set to 1 indicates support for eIM Package injection (see section 3.1.1.2) using CoAP/DTLS over UDP as described in section 3.1.2.2
- `eimProprietary` set to 1 indicates support of a transport protocol not described in section 3.1.2.
- `eSipaProprietaryProtocolInformation` provides additional eIM Capabilities information related to the additional proprietary supported protocol(s). This field MAY be provided when `eimProprietary` is set to 1. This field SHALL NOT be provided if `eimProprietary` is not set.

## 4.3 eIM Identifier

The `eimId` is an eIM identifier that SHALL exist uniquely within the eUICC's list of Associated eIMs. The `eimId` is a text string (ASN.1 type `UTF8String`) as shown in section 2.11.1.1.1 and section 2.11.1.1.2. The total length of the `eimId` SHALL not exceed 128 Bytes.

NOTE: The eIM ID is defined as common name in the subject field of both CERT.EIM.ECDSA and CERT.EIM.TLS (see sections 2.6.4.3 and 2.6.4.4). According to RFC 5280 [25] the maximum length of a common name is 64 characters. It is recommended to align the total length of the `eimId` with the common name used in the subject field of the eIM certificates.

The `eimId` text string MAY be of the following types as defined by `EimIdType` in section 2.11.1.1.1: OID, FQDN, and Proprietary.

Where an eIM operator holds or can request an OID assignment, an eIM OID SHOULD be allocated to the eIM and used to encode the `eimId`.

An `eimId` of type `eimIdTypeOid` SHALL be encoded using the OID, in a sub arc allocated by <https://www.iana.org/assignments/enterprise-numbers/enterprise-numbers>.

NOTE1: It is up to the eIM owner to define child nodes and values accordingly.

NOTE2: An example OID "iso.org.dod.internet.private.enterprise.exampleCompanyName.eim.eimowner.124" could be represented as follows: "1.3.6.1.4.1.32473.999.10.124".

An `eimId` of type `eimIdTypeFqdn` SHALL be encoded as a FQDN.

NOTE: An example could be: "eim.enterprise.com".

For an `eimId` of type `eimIdTypeProprietary` the `eimId` value can be freely chosen, following the requirements set out in this section.

#### 4.4 Profile MetaData

The profile Metadata as defined in SGP.22 [4] SHALL be replaced as follows:

```
-- ASN1START
-- Definition of ProfileInfo
ProfileInfo ::= [PRIVATE 3] SEQUENCE { -- Tag 'E3'
    iccid Iccid OPTIONAL,
    isdpAid [APPLICATION 15] OctetTo16 OPTIONAL, -- AID of the ISD-P containing the
Profile, tag '4F'
    profileState [112] ProfileState OPTIONAL, -- Tag '9F70'
    profileNickname [16] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '90'
    serviceProviderName [17] UTF8String (SIZE(0..32)) OPTIONAL, -- Tag '91'
    profileName [18] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '92'
    iconType [19] IconType OPTIONAL, -- Tag '93'
    icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94', see condition in
ES10b:GetProfilesInfo
    profileClass [21] ProfileClass OPTIONAL, -- Tag '95'
    notificationConfigurationInfo [22] SEQUENCE OF
NotificationConfigurationInformation OPTIONAL, -- Tag 'B6'
    profileOwner [23] OperatorId OPTIONAL, -- Tag 'B7'
    dpProprietaryData [24] DpProprietaryData OPTIONAL, -- Tag 'B8'
    profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
    serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, -- Tag
'BF22'
    ecallIndication [123] BOOLEAN OPTIONAL, -- Tag '9F7B'
    fallbackAttribute [38] BOOLEAN DEFAULT FALSE, -- Tag '9F26', indicates a
Fallback Profile if set to TRUE
    fallbackAllowed [103] BOOLEAN OPTIONAL, -- Tag '9F67', indicates if Fallback is
authorized by the Profile Owner
    iotSpecificProfileInfo [100] SEQUENCE {
        -- Data objects and their tags
    } OPTIONAL -- Tag 'BF64' IoT specific Data object
}
-- ASN1STOP
```

The data item `ecallIndication` MAY be present. If absent it SHALL be treated as if it was set to FALSE. An `ecallIndication` data item set to TRUE indicates, that the enabling or disabling of an Emergency Profile MAY be triggered using the `ES10b.EnableEmergencyProfile` and `ES10b.DisableEmergencyProfile` functions.

It SHALL NOT be possible to have multiple Profiles on an eUICC with the `ecallIndication` data item set to TRUE.

The data item `fallbackAttribute` MAY be present. If set to TRUE, the Profile can be enabled using the `ES10b.ExecuteFallbackMechanism` function. If the `fallbackAttribute` data item is absent or set to FALSE the Profile cannot be enabled using the `ES10b.ExecuteFallbackMechanism` function.

It SHALL NOT be possible to have multiple Profiles on an eUICC with the `fallbackAttribute` set to TRUE.

The data item `fallbackAllowed` MAY be present. Only if set to TRUE, it SHALL be possible to set the `fallbackAttribute`.

NOTE 1: The Profile Owner decides upon Profile generation to include or not the `fallbackAllowed` flag into the Profile.

NOTE 2: The structure `iotSpecificProfileInfo` is for forward compatibility and is empty in this version of the specification.

## 5 Functions

### 5.1 Overview of Functions per Interface

This section provides the description of the interfaces and functions within the Remote SIM Provisioning and Management system involving the eUICC, including the following:

#### ***eUICC Interfaces***

- ES6: The interface used by the Operator to manage the content of their Profile.
- ES8+: Provides a secure end-to-end channel between the SM-DP+ and the eUICC for the administration of the ISD-P and the associated Profile during download and installation.
- ES9+: Used to provide a secure transport between the SM-DP+ and the IPAe for the delivery of the Profile Package.
- ES10a: Used by the IPAd to get the configured addresses from the eUICC for Root SM-DS, and optionally for the default SM-DP+.
- ES10b: Used by the IPAd to transfer a Profile Package to the eUICC.
- ES11: Interface between the IPAe and an SM-DS (Alternative SM-DS or Root SM-DS) for the Event Retrieval.
- ESipa: Interface between an eIM and an IPA used to trigger a Profile download at the IPA and to provide a secure transport for the delivery of eUICC Packages unless the underlying transport provides necessary security.
- ESep: Logical end-to-end interface between the eIM and the eUICC used to transfer eUICC Packages for Profile State Management and eIM configuration by eIM.

#### ***Server to Server Interfaces***

- ES2+: Interface between the Operator and the SM-DP+ used by the Operator to order Profile Package preparation.
- ES12: Interface between the SM-DP+ and an SM-DS (Alternative SM-DS or Root SM-DS) for the Event management.

#### ***Device to RSP Server Interfaces***

- ES9+: Used to provide a secure transport between the SM-DP+ and the IPAd for the delivery of the Profile Package.
- ES11: Interface between the IPAd and an SM-DS (Alternative SM-DS or Root SM-DS) for the Event Retrieval.

#### ***eIM interfaces***

- ES9+': Used to provide a secure transport between the SM-DP+ and the eIM for the delivery of the Bound Profile Package.
- ESipa: Interface between an eIM and an IPA used to trigger a Profile download at the IPA and to provide a secure transport for the delivery of eUICC Packages unless the underlying transport provides necessary security.
- ESep: Logical end-to-end interface between the eIM and the eUICC used to transfer eUICC Packages for Profile State Management and eIM configuration by eIM.
- ES11': Interface between the eIM and the SM-DS (Alternative SM-DS or Root SM-DS) for the Event Retrieval.

The following table presents the normative list of all the functions that are defined in this section.

**Request-Response Functions:**

Interface	Functions	Function provider Role
ES2+	As defined in section 5.3 of SGP.22 [4].	SM-DP+
ES6	As defined in section 5.2 of SGP.22 [4].	ISD-P
ES8+	As defined in section 5.5 of SGP.22 [4].	eUICC
ES9+	As defined in section 5.6 of SGP.22 [4].	SM-DP+
	AuthenticateClient	
ES9+'	InitiateAuthentication	
	GetBoundProfilePackage	
	AuthenticateClient	
	HandleNotification	
	CancelSession	
ES10a	As defined in section 5.7 of SGP.22 [4].	ISD-R (IPA Services)
	GetEuiccConfiguredAddresses	
ES10b	LoadEuiccPackage	ISD-R (IPA Services)
	GetEUICCInfo	
	GetEUICCChallenge	
	AddInitialEim	
	eUICCMemoryReset	
	AuthenticateServer	
	PrepareDownload	
	LoadBoundProfilePackage	
	CancelSession	
	GetCerts	
	GetEID	
	RetrieveNotificationsList	
	RemoveNotificationFromList	

	GetRAT	
	GetProfilesInfo	
	ImmediateEnable	
	ProfileRollback	
	ConfigureImmediateProfileEnabling	
	GetEimConfigurationData	
	ExecuteFallbackMechanism	
	ReturnFromFallback	
	EnableEmergencyProfile	
	DisableEmergencyProfile	
	GetConnectivityParameters	
	SetDefaultDpAddress	
ES11	As defined in section 5.8 of SGP.22 [4].	SM-DS
ES11'	InitiateAuthentication	SM-DS
	AuthenticateClient	
ES12	As defined in section 5.9 of SGP.22 [4].	SM-DS
ESipa	TransferEimPackage	IPA
	InitiateAuthentication	eIM
	GetBoundProfilePackage	
	AuthenticateClient	
	GetEimPackage	
	ProvideEimPackageResult	
	HandleNotification	
	CancelSession	
ESep	Enable	eUICC
	Disable	
	Delete	
	ListProfileInfo	
	GetRAT	
	ConfigureImmediateEnable	
	AddEim	
	UpdateEim	
	DeleteEim	
	ListEim	
	SetDefaultDpAddress	

**Table 7 Request-Response Functions**

**Notification Handler Functions:**

Interface	Notification handler functions	Function handler/recipient
ES2+	As defined in section 5.3 of SGP.22 [4].	Operator
ES9+	HandleNotification	SM-DP+
ES9+'	HandleNotification	SM-DP+
ESipa	HandleNotification	eIM

**Table 8 Notification Handler Functions**

## 5.2 Server to Server Function Commonalities

Each function represents an entry point that is provided by a Role (function provider), and that can be called by other Roles (function requester).

### 5.2.1 Common Data Types

The functions provided in this section deal with management of the eUICC and Profile, so that the common data defined in this section needs to be used in most of the functions.

Type name	Description	Type definition
	As defined in table 21 of section 5.2.1 of SGP.22 [4]	

**Table 9 Common data types**

### 5.2.2 Request-Response Function

As defined in section 5.2.2 of SGP.22 [4].

### 5.2.3 Notification Handler Function

As defined in section 5.2.3 of SGP.22 [4].

### 5.2.4 Functions Input Header

As defined in section 5.2.4 of SGP.22 [4].

### 5.2.5 Functions Output Header

As defined in section 5.2.5 of SGP.22 [4].

### 5.2.6 Status Code

This specification relies on subject codes and reason codes as defined in section 5.2.6 of SGP.22 [4]. In addition, this specification defines additional codes.

#### 5.2.6.1 Subject Code

Hereunder are listed the subject codes used in this specification:

1. Generic (as defined in SGP.02 [6])
  - 1.1. Function Requester (as defined in SGP.02 [6])
  - 1.2. Function Provider (as defined in SGP.02 [6])
  - 1.3. Protocol (as defined in SGP.02 [6])

- 1.3.1. Protocol Format (as defined in SGP.02 [6])
- 1.3.2. Protocol Version (as defined in SGP.02 [6])
- 1.6. Function (as defined in SGP.02 [6])
- 8. eUICC Remote Provisioning (as defined in SGP.02 [6])
  - 8.1. eUICC (as defined in SGP.02 [6])
    - 8.1.1. EID (as defined in SGP.02 [6])
    - 8.1.2. EUM Certificate (as defined in SGP.22 [4])
    - 8.1.3. eUICC Certificate (as defined in SGP.22 [4])
  - 8.2. Profile (as defined in SGP.02 [6])
    - 8.2.1. Profile ICCID (as defined in SGP.02 [6])
    - 8.2.5. Profile Type (as defined in SGP.02 [6])
    - 8.2.6. Matching ID (as defined in SGP.22 [4])
    - 8.2.7. Confirmation Code (as defined in SGP.22 [4])
    - 8.2.8. PPR (as defined in SGP.22 [4])
    - 8.2.9. Profile Metadata (as defined in SGP.22 [4])
    - 8.2.10. Bound Profile Package (as defined in SGP.22 [4])
  - 8.8. SM-DP+ (as defined in SGP.22 [4])
    - 8.8.1. SM-DP+ Address (as defined in SGP.22 [4])
    - 8.8.2. Security configuration (as defined in SGP.22 [4])
    - 8.8.3. Specification Version Number (SVN) (as defined in SGP.22 [4])
    - 8.8.4. SM-DP+ Certificate (as defined in SGP.22 [4])
    - 8.8.5. Download order (as defined in SGP.22 [4])
  - 8.9. SM-DS (as defined in SGP.22 [4])
    - 8.9.1. SM-DS Address (as defined in SGP.22 [4])
    - 8.9.2. Security configuration (as defined in SGP.22 [4])
    - 8.9.3. Specification Version Number (SVN) (as defined in SGP.22 [4])
    - 8.9.4. SM-DS Certificate (as defined in SGP.22 [4])
    - 8.9.5. Event Record (as defined in SGP.22 [4])

- 8.10. RSP Operation (as defined in SGP.22 [4])
  - 8.10.1. TransactionId (as defined in SGP.22 [4])
- 8.11. GSMA CI (as defined in SGP.22 [4])
  - 8.11.1. Public Key (PK) (as defined in SGP.22 [4])
- 8.31. eIM
  - 8.31.1. eIM Package
  - 8.31.2. eIM TransactionId

#### **5.2.6.2 Reason Code**

Hereunder are listed the reason codes used in this specification:

1. Access Error (as defined in SGP.02 [6])
  - 1.1. Unknown (Identification or Authentication) (as defined in SGP.02 [6])
  - 1.2. Not Allowed (Authorisation) (as defined in SGP.02 [6])
2. Format Error (as defined in SGP.02 [6])
  - 2.1. Invalid (as defined in SGP.02 [6])
  - 2.2. Mandatory Element Missing (as defined in SGP.02 [6])
  - 2.3. Conditional Element Missing (as defined in SGP.02 [6])
3. Conditions of Use Not Satisfied (as defined in SGP.02 [6])
  - 3.1. Unsupported (as defined in SGP.02 [6])
  - 3.3. Already in Use (Uniqueness) (as defined in SGP.02 [6])
  - 3.7. Unavailable (as defined in SGP.02 [6])
  - 3.8. Refused (as defined in SGP.02 [6])
  - 3.9. Unknown (as defined in SGP.02 [6])
  - 3.10. Invalid Association (as defined in SGP.22 [6])
  - 3.11. Value has Changed (as defined in SGP.22 [6])
4. Processing Error (as defined in SGP.02 [6])
  - 4.2. Execution Error (as defined in SGP.02 [6])
  - 4.3. Stopped on Warning (as defined in SGP.02 [6])
  - 4.8. Insufficient Memory (as defined in SGP.02 [6])
  - 4.10. Time to Live Expired (as defined in SGP.22 [4])

## 5. Transport Error (as defined in SGP.02 [6])

### 5.1. Inaccessible (as defined in SGP.02 [6])

## 6. Security Error (as defined in SGP.02 [6])

### 6.1. Verification Failed (as defined in SGP.02 [26])

### 6.3. Expired (as defined in SGP.22 [4])

### 6.4. Maximum number of retries exceeded (as defined in SGP.22 [4])

### 5.2.6.3 Common Function Status Code

As defined in SGP.02 [6].

## 5.3 ES2+ (Operator -- SM-DP+)

ES2+ is the interface between:

- The Operator.
- SM-DP+

This interface is used by the Operator to order the Profile Package preparation for specific eUICC(s) and the delivery of the Profile Package from the SM-DP+. This interface is identical to the one defined in section 5.3 of SGP.22 [4].

## 5.4 ES6 (Operator -- eUICC)

ES6 is the interface between:

- The Operator.
- eUICC (Enabled Profile)

This interface is used by the Operator to make modifications on their Profile in the eUICC using legacy OTA mechanisms.

NOTE: SCP80 over SMS might not be supported in ES6 in case of NCD.

### 5.4.1 Function: ConnectivityParametersUpdateByOperator

**Related Procedures:** N/A

**Function Provider entity:** ISD-P

#### **Description:**

This function is used to update Connectivity Parameters on the eUICC.

This function has the following parameter:

- Connectivity Parameters

As this function is provided by the ISD-P, the STORE DATA command message defined hereunder SHALL be preceded by an INSTALL [for personalisation] as defined in section 5.4.2:

**STORE DATA command:**

This command is a STORE DATA command, as described in GlobalPlatform Card Specification [32] section 11.11.2.

Code	Value	Meaning
CLA	'80'	GlobalPlatform Card Specification [32] section 11.11
INS	'E2'	STORE DATA
P1	'88'	Reference Control Parameter P1
P2	'00'	Block Number
Lc	xx	Length of data field
Data	xx	Application Data and MAC (if present)
Le	Not present	

**Table 10 ConnectivityParametersUpdateByOperator Command Message**

Parameter P1 is coded according to the following table:

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
1	=	=	=	=	=	=	=	Last block
=	0	0	=	=	=	=	=	No general encryption information or non-encrypted data
=	=	=	0	1	=	=	=	DGI format of the command data field
							0	Case 3 command as defined in GlobalPlatform [32]
=	=	=	=	=	X	X	=	RFU

**Table 11 ConnectivityParametersUpdateByOperator Reference Control Parameter P1**

**Data Field Sent in the Command Message**

The data field is defined in Table 2 of section 2.4.4, including tag '3A07'.

**Response Message**

**Data Field Returned in the Response Message:**

The data field of the response message SHALL NOT be present.

**Processing State returned in the Response Message:**

See GlobalPlatform Card Specification [32] section 11.11.3.

**5.4.2 Function: UpdateMetadata**

Related Procedures: N/A

Function Provider Entity: ISD-P

## Description:

This function is identical to the ES6.UpdateMetadata function defined in section 5.4.1 of SGP.22 [4], with the addition that this function allows the update of the following Profile Metadata of the targeted Profile:

- fallbackAllowed

NOTE: Currently it is not possible to update the eCallIndication data element using ES6.

The data field of the ES6.UpdateMetadata SHALL follow the ASN.1 message definition in the ES6.UpdateMetadata described in section 5.4.1 and Annex H of SGP.22 [4], renaming the data field 'reserved103' by 'fallbackAllowed' as defined below. Consequently, the following ASN.1 structure applies.

### Data Field

The data field SHALL be coded as defined hereunder.

```
-- ASN1START
UpdateMetadataRequest ::= [42] SEQUENCE { -- Tag 'BF2A'
  serviceProviderName [17] UTF8String (SIZE(0..32)) OPTIONAL, -- Tag '91'
  profileName [18] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '92'
  iconType [19] IconType OPTIONAL, -- Tag '93'
  icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94'
  profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
  serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, -- Tag
'BF22'
  fallbackAllowed [103] BOOLEAN OPTIONAL -- Tag '9F67'
}
-- ASN1STOP
```

'serviceProviderName', 'profileName', 'iconType', 'icon', 'profilePolicyRules', 'serviceSpecificDataStoredInEuicc' and 'fallbackAllowed' fields in UpdateMetadataRequest are optional but at least one field SHALL be present.

'fallbackAllowed' is BOOLEAN type and consequently SHALL only be set to TRUE or FALSE. If absent, value in profile metadata SHALL NOT change.

For the fallbackAllowed flag:

- Upon reception of a UpdateMetadataRequest containing fallbackAllowed set to TRUE:
  - If target Profile has the ecallIndication set to TRUE then following status bytes SHALL be returned in the response message, as described in GlobalPlatform Card Specification [32] section 11.11.3:
    - '6A 80': Incorrect values in command data
- Upon reception of a UpdateMetadataRequest containing fallbackAllowed set to FALSE:
  - If target Profile has the fallbackAllowed set to TRUE, and fallbackAttribute set to TRUE then following status bytes SHALL be returned in the response message:
    - '6A 89': Fallback attribute set

NOTE: As per DER encoding, the field 'fallbackAllowed' cannot be present with an empty value.

### **Processing State Returned in the Response Message**

In addition to the *State Returned in the Response Message* described in section 5.4.1 of SGP.22 [4], the following additional status bytes are defined:

- '6A 89': Fallback attribute set

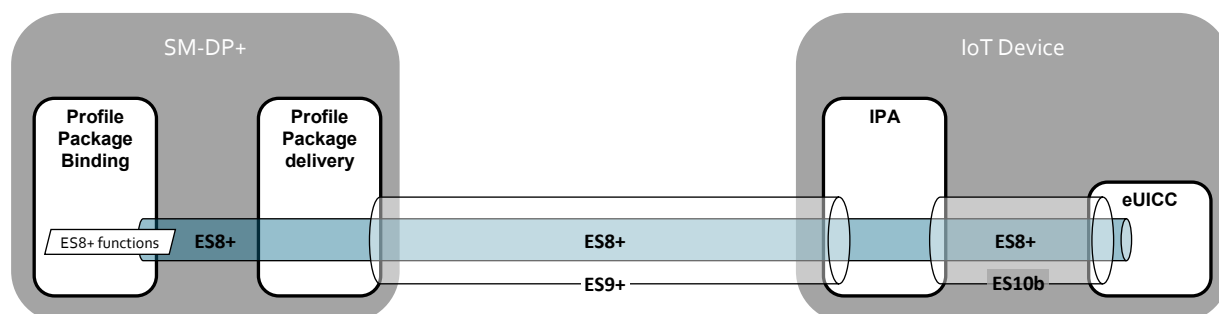
## **5.5 ES8+ (SM-DP+ -- eUICC)**

The ES8+ is the interface between:

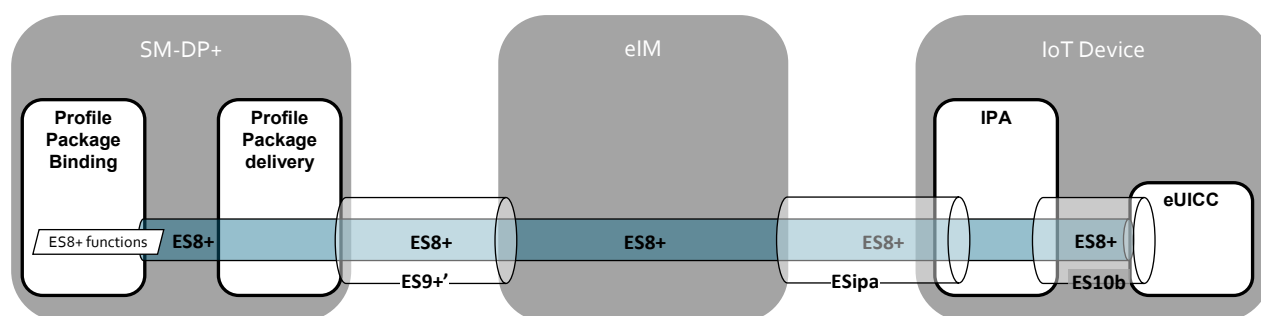
- The SM-DP+ (Profile Package Binding function).
- The eUICC.

This interface is intended to be tunnelled either over:

- the ES9+ and ES10b interfaces for direct profile download (i.e., the IPA directly communicates with the SM-DP+) as shown in Figure 29 or
- ES9+, ESipa, and ES10b for indirect profile download (i.e., the IPA communicates with the SM-DP+ via the eIM) as shown in Figure 30.



**Figure 30 ES8+ interface through ES9+ and ES10b**



**Figure 30 ES8+ through ES9+' ESipa, and ES10b**

The ES8+ functions are addressed to the eUICC through a secure channel established between the Profile Package Binding function of the SM-DP+ and the eUICC.

The ES8+ functions are called by the SM-DP+ and executed by the eUICC.

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4] with the following modifications to the command data structure of the ES8+.StoreMetadata function.

**Command data**

The command data for this function is identified by the data structure defined hereunder.

```
-- ASN1START
StoreMetadataRequest ::= [37] SEQUENCE { -- Tag 'BF25'
    iccid Iccid,
    serviceProviderName [17] UTF8String (SIZE(0..32)), -- Tag '91'
    profileName [18] UTF8String (SIZE(0..64)), -- Tag '92' (corresponds to 'Short
Description' defined in SGP.21 [2])
    iconType [19] IconType OPTIONAL, -- Tag '93' (JPG or PNG)
    icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94' (Data of the icon.
Size 64 x 64 pixel. This field SHALL only be present if iconType is present)
    profileClass [21] ProfileClass DEFAULT operational, -- Tag '95'
    notificationConfigurationInfo [22] SEQUENCE OF
NotificationConfigurationInformation OPTIONAL,
    profileOwner [23] OperatorId OPTIONAL, -- Tag 'B7'
    profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
    serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, -- Tag
'BF22'
    serviceSpecificDataNotStoredInEuicc [35] VendorSpecificExtension OPTIONAL, --
Tag 'BF23'
    ecallIndication [123] BOOLEAN OPTIONAL, -- Tag '9F7B'
    fallbackAllowed [103] BOOLEAN OPTIONAL, -- Tag '9F67'
    iotSpecificMetadata [100] SEQUENCE {
        -- Data objects and their tags
    } OPTIONAL -- Tag 'BF64' IoT specific Data object
```

```
}  
-- ASN1STOP
```

**NOTE:** The structure `iotSpecificMetadata` is introduced for forward compatibility and is empty. It may be used in future versions of this specification.

Upon reception of a `StoreMetadataRequest` containing `ecallIndication` set to `TRUE` the following steps are performed in addition to the procedure defined in SGP.22 [4] section 5.5.3:

**Start Conditions:**

- `ecallIndication` is set to `TRUE` in the `StoreMetadataRequest`.

**Procedure:**

The eUICC SHALL:

1. Verify that the eUICC supports the Emergency Mechanism and that the eUICC does not contain an Emergency Profile (i.e. a Profile with `ecallIndication` set to `TRUE`).
2. Verify that `fallbackAllowed` is not set to `TRUE`.
3. In case of failed verification in the steps above, the eUICC SHALL return error with reason `incorrectInputValues` and the procedure SHALL stop.
4. Set the `ecallIndication` to `TRUE` for the target Profile.
5. Generate as many Notifications as configured in the metadata.
6. Return `ok`.

## 5.6 ES9+ (IPA -- SM-DP+)

ES9+ is the interface between:

- The IPA entity.
- The SM-DP+

The IPA SHALL communicate with the SM-DP+ secured by HTTPS in server authentication mode as described in SGP.22 [4] section 2.6.6.

This interface is identical to the ES9+ interface defined in section 5.6 of SGP.22 [4], where the IPA plays the role of LPA, except Function (ES9+): `AuthenticateClient` which is defined bellow.

### 5.6.1 Function (ES9+): `AuthenticateClient`

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** SM-DP+

**Description:**

This function SHALL be called by the IPA to request the authentication of the eUICC by the SM-DP+.

This function is identical to the ES9+.AuthenticateClient function defined in section 5.6.3 of SGP.22 [4], where the IPA plays the role of LPA and the EUICCInfo2 in AuthenticateResponseOk is according to Section 5.9.2 of this specification.

```
-- ASN1START
AuthenticateClientRequest ::= [59] SEQUENCE { -- Tag 'BF3B'
    transactionId [0] TransactionId,
    authenticateServerResponse [56] AuthenticateServerResponse -- This is the response
from ES10b.AuthenticateServer
}
-- ASN1STOP
```

## 5.7 ES9+' (eIM -- SM-DP+)

ES9+' is the interface between:

- The eIM entity.
- The SM-DP+.

The eIM SHALL communicate with the SM-DP+ secured by HTTPS in server authentication mode as described in SGP.22 [4] section 5.6 using the TLS requirements in SGP.22 [4] section 2.6.6.

### 5.7.1 Function (ES9+): InitiateAuthentication

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** SM-DP+

#### Description:

This function SHALL be called by the eIM to authenticate the SM-DP+.

This function is identical to the ES9+.InitiateAuthentication function defined in section 5.6.1 of SGP.22 [4], where the eIM plays the role of LPA. In addition, in case the IPA has sent all the necessary input data in ESipa.InitiateAuthentication request, the eIM SHALL forward it as ES9+'.InitiateAuthentication request. Otherwise, the eIM SHALL complete the missing input data as described in ES9+.InitiateAuthentication function.

### 5.7.2 Function (ES9+): GetBoundProfilePackage

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** SM-DP+

**Function Caller Entity:** eIM

#### Description:

This function SHALL be called by the eIM to request the delivery and the binding of a Profile Package for the eUICC.

This function is identical to the ES9+.GetBoundProfilePackage function defined in section 5.6.2 of SGP.22 [4], where the eIM plays the role of LPA.

### **5.7.3 Function (ES9+): AuthenticateClient**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** SM-DP+

**Function Caller Entity:** eIM

#### **Description:**

This function SHALL be called by the eIM to request the authentication of the eUICC by the SM-DP+.

This function is identical to the ES9+.AuthenticateClient function defined in section 5.6.3 of SGP.22 [4], where the eIM plays the role of LPA and the EUICCInfo2 in AuthenticateResponseOk is according to Section 5.9.2 of this specification.

### **5.7.4 Function (ES9+): HandleNotification**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** SM-DP+

**Function Caller Entity:** eIM

#### **Description:**

This function SHALL be called by the eIM to notify the SM-DP+ that a Profile Management Operation has successfully been performed on the eUICC. This function is called by the eIM upon the receipt by the eIM of either an ESipa.HandleNotification function call or an ESipa.ProvideEimPackageResult function call sent by the IPA containing a PendingNotification. When the received PendingNotification contains a ProfileInstallationResult that is part of an ongoing Indirect Profile Download session, the eIM SHALL send the ES9+'.HandleNotification call to the SM-DP+ that is participating in that Indirect Profile Download session. When the PendingNotification contains an OtherSignedNotification, the eIM SHALL send the ES9+'.HandleNotification to the NotificationAddress contained in the NotificationMetadata of the OtherSignedNotification.

This function is identical to the ES9+.HandleNotification function defined in section 5.6.4 of SGP.22 [4], where the eIM plays the role of LPA. In addition, in case the IPA has sent all the necessary input data in ESipa.HandleNotification request, the eIM SHALL forward it as ES9+'.HandleNotification request. Otherwise, the eIM SHALL complete the missing input data as described in ES9+.HandleNotification function.

### **5.7.5 Function (ES9+): CancelSession**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** SM-DP+

#### **Description:**

This function SHALL be called by the eIM to request the cancellation of an on-going RSP session.

This function is identical to the ES9+.CancelSession function defined in section 5.6.5 of SGP.22 [4], where the eIM plays the role of LPA. In addition, in case the IPA has sent all the necessary input data in ESipa.CancelSession request, the eIM SHALL forward it as ES9+'.CancelSession request. Otherwise, the eIM SHALL complete the missing input data as described in ES9+.CancelSession function.

## 5.8 ES10a (IPA -- eUICC)

### 5.8.1 Function (ES10a): GetEuiccConfiguredAddresses

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

#### **Description:**

This function is identical to the ES10a.GetEuiccConfiguredAddresses function defined in section 5.7.3 of SGP.22 [4], where the IPA plays the role of LPA.

NOTE : the default SM-DP+ address that is stored in the eUICC can be updated through the functions ConfigureImmediateProfileEnabling (described in 5.9.17 of this specification) and ConfigureImmediateEnable (described in 5.13.6 of this specification) as well as functions ES10b.SetDefaultDpAddress as described in 5.9.25 of this specification and ESep.SetDefaultDpAddress as described in 5.13.11 of this specification.

## 5.9 ES10b (IPA -- eUICC)

### 5.9.1 Function (ES10b): LoadEuiccPackage

**Related Procedures:** Generic eUICC Package Download and Execution

**Function Provider Entity:** ISD-R (IPA Services)

#### **Description:**

This function executes a eUICC Package.

On reception of this package, the eUICC SHALL:

- Retrieve the public key of the Associated eIM using the `eimId` in `euiccPackageSigned`. If the `eimId` is unknown, the eUICC SHALL return an unsigned eUICC Package error result (`euiccPackageErrorUnsigned`) containing the `eimId` received in the `euiccPackageSigned`.
- If the `euiccPackageRequest` size exceeds the eUICC capacity, the eUICC SHOULD return an unsigned eUICC Package error result (`euiccPackageErrorUnsigned`) containing the `eimId` received in the `euiccPackageSigned` and the `sizeOverflow` error code.

- Verify the `eimSignature` created upon `euiccPackageSigned` concatenated with `associationToken` using the retrieved eIM public key. If no association token is configured for the eIM, the `associationToken` data object SHALL be used with the value set to zero for verifying the signature. If the signature is invalid, the eUICC SHALL return an unsigned eUICC Package error result (`euiccPackageErrorUnsigned`) containing the `eimId` received in the `euiccPackageSigned`. If and only if an association token is configured for the eIM, the eUICC SHALL return the `associationToken` in the error result.
- Verify that the `eidValue` in `euiccPackageSigned` matches its own EID. If the EID verification fails, the eUICC SHALL return a signed eUICC error result (`euiccPackageErrorSigned`) containing the `invalidEid` error code.
- Verify that the `counterValue` in `euiccPackageSigned` is greater than the counter value stored in the eUICC for the Associated eIM. If the counter value verification fails, the eUICC SHALL return a signed eUICC Package error result (`euiccPackageErrorSigned`) containing the `replayError` error code.
- Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).
- Execute PSMO(s) or eCO(s) contained in the eUICC Package and generate the corresponding PSMO or eCO result(s) (`euiccResultData`) sequentially.

If the eUICC encounters an error for a PSMO or an eCO, the eUICC SHALL terminate processing of the remaining command sequence and generate `euiccResultData` containing `processingTerminated` with the following error codes:

- For a PSMO 'List Profile Info' request, if the eUICC cannot generate the `euiccResultData` due to the size limit of the response data, the error code SHALL be `resultSizeOverflow`.

NOTE: this could happen, e.g., when retrieving Profile Metadata of multiple Profiles containing large icons.

- If a PSMO or an eCO in the eUICC Package is unknown or unsupported, or the command data cannot be interpreted, the error code SHALL be `unknownOrDamagedCommand`.
- Generate a signed eUICC Package Result (`euiccPackageResultSigned`) with the execution results of the PSMOs/eCOs (`euiccResultData`). The eUICC Package Result SHALL be signed using SK.EUICC.ECDSA. In case there are more than one SK.EUICC.ECDSA available for signing, the SK.EUICC.ECDSA chaining back to the CI public key identified by the `euiccCiPKId` of the eIM Configuration Data (of the Associated eIM for which the eUICC Package Result is intended) SHALL be used. The signed data includes the next (not used) value of the sequence number (defined in SGP.22 [4] for use with Notifications).
- Increment the sequence number in eUICC storage by 1.
- Update the counter value stored in the eUICC for the Associated eIM to the `counterValue` received in the `euiccPackageSigned`.

- Return the signed eUICC Package Result (`euiccPackageResultSigned`).

The `ES10b.LoadEuiccPackage` function SHALL be processed in an atomic way, meaning that in case of non-resumable interruption (e.g., power loss), the eUICC SHALL revert all performed changes related to PSMOs or eCOs within that eUICC Package. The eUICC SHALL restore its original state prior to execution of the `ES10b.LoadEuiccPackage`, including the `counterValue`. Any intermediate result generated for the eUICC Package SHALL be discarded.

NOTE: In case of non-resumable interruption the ISD-R withdraws the previously received eUICC Package.

The IPA MAY call the `ES10b.LoadEuiccPackage` function again, including the same eUICC Package, if it detects that the previous processing was interrupted.

NOTE: How the IPA detects the interruption and how an IPA resubmits the eUICC Package is out of scope.

### **Command Data**

The `EuiccPackageRequest` as defined in section 2.11.1.1.

### **Response Data**

The `EuiccPackageResult` as defined in section 2.11.2.1.

## **5.9.2 Function (ES10b): GetEUICCInfo**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

### **Description:**

This function is identical to the `ES10b.GetEUICCInfo` function defined in section 5.7.8 of SGP.22 [4], where the IPA plays the role of LPA.

The response data for `EUICCInfo2` SHALL follow the ASN.1 message definition below replacing the `EUICCInfo2` described in section 5.7.8 and Annex H of SGP.22 [4].

NOTE: `EUICCInfo2` in this specification is based on `EUICCInfo2` as defined in SGP.22 [4] but has been extended for IoT purposes.

```
-- ASN1START
EUICCInfo2 ::= [34] SEQUENCE { -- Tag 'BF22'
  profileVersion [1] VersionType, -- Base eUICC Profile package version supported
  svn [2] VersionType, -- GSMA SGP.22 version supported (SVN) referenced by SGP.32
  euiccFirmwareVer [3] VersionType, -- eUICC Firmware version
  extCardResource [4] OCTET STRING, -- Extended Card Resource Information
  according to ETSI TS 102 226
  uiccCapability [5] UICCCapability,
  ts102241Version [6] VersionType OPTIONAL,
  globalplatformVersion [7] VersionType OPTIONAL,
  rspCapability [8] RspCapability,
  euiccCipKeyIdListForVerification [9] SEQUENCE OF SubjectKeyIdentifier, -- List of
  CI Public Key Identifiers supported on the eUICC for signature verification
  euiccCipKeyIdListForSigning [10] SEQUENCE OF SubjectKeyIdentifier, -- List of CI
  Public Key Identifier supported on the eUICC for signature creation
}
```

```
euiccCategory [11] INTEGER {
    other(0),
    basicEuicc(1),
    mediumEuicc(2),
    contactlessEuicc(3)
} OPTIONAL,
forbiddenProfilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
ppVersion VersionType, -- Protection Profile version
sasAccreditationNumber UTF8String (SIZE(0..64)),
certificationDataObject [12] CertificationDataObject OPTIONAL,
treProperties [13] BIT STRING {
    isDiscrete(0),
    isIntegrated(1),
    usesRemoteMemory(2) -- refers to the usage of remote memory protected by the
Remote Memory Protection Function described in SGP.21 [4]
} OPTIONAL,
treProductReference [14] UTF8String OPTIONAL, -- Platform_Label as defined in
GlobalPlatform DLOA specification [57]
additionalEuiccProfilePackageVersions [15] SEQUENCE OF VersionType OPTIONAL,
ipaMode [16] IpaMode OPTIONAL, -- active IPA, mandatory within SGP.32
euiccCiPKIdListForSigningV3 [17] SEQUENCE OF SubjectKeyIdentifier OPTIONAL, --
not used by this version of SGP.32.
additionalEuiccInfo [18] OCTET STRING (SIZE(0..32)) OPTIONAL, -- not used by
this version of SGP.32
highestSvn [19] VersionType OPTIONAL, -- not used by this version of SGP.32
iotSpecificInfo [20] IoTSpecificInfo OPTIONAL, -- mandatory within SGP.32
euiccMinimumSecurityLevel [21] OCTET STRING (SIZE (1)) OPTIONAL -- value for
eUICC Minimum Security Level Check
}

-- Definition of IoTSpecificInfo
IoTSpecificInfo ::= SEQUENCE {
    iotVersion [0] SEQUENCE OF VersionType, -- SGP.32 version(s) supported by the
eUICC, at least one must be present
    ecallSupported [1] NULL OPTIONAL, -- if present, eUICC supports the Emergency
Mechanism
    fallbackSupported [2] NULL OPTIONAL, -- if present, eUICC supports the Fallback
Mechanism
    fallbackAllowedUpdateSupported [3] NULL OPTIONAL -- if present, eUICC supports
the update of the Fallback Allowed metadata via ES6
}

-- Definition of IpaMode
IpaMode ::= INTEGER {
    ipad (0), -- IPAd is active
    ipae (1) -- IPAe is active
}
-- ASN1STOP
```

The procedures related to the eUICC Profile Package versioning and associated to profileVersion, additionalEuiccProfilePackageVersions and euiccCapability are described in section 5.7.8 of SGP.22 [4].

**NOTE:** as part of the TCA eUICC Profile Package specification [29], in version v3.3 a new Profile type called "IoT Minimal Profile" has been introduced with the intention to be useful for the provisioning of eUICCs in Network Constrained Devices.

The euiccCiPKIdListForSigningV3, additionalEuiccInfo, and highestSvn fields are not used in this version of the specification. The eUICC SHALL NOT include these fields in euiccInfo2. The SM-DP+ SHOULD ignore these fields if received with iotSpecificInfo being present.

The `iotSpecificInfo` data object SHALL be present.

The `iotSpecificMetadataSupport` bit in `RspCapability` SHALL be set to '0'.

NOTE: `iotSpecificMetadataSupport` is introduced for forward compatibility and may be used in future versions of this specification.

If the eUICC supports the Fallback Mechanism:

- The `fallbackSupported` flag SHALL be present within `IoTSpecificInfo`.
- The `fallbackAllowedUpdateSupported` flag SHALL be present within `IoTSpecificInfo`.

The `ecallSupported` data object SHALL be present within `IoTSpecificInfo`. If the flag is set, the eUICC SHALL support the Emergency Mechanism.

### 5.9.3 Function (ES10b): GetEUICCChallenge

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

#### **Description:**

This function is identical to the ES10b.GetEUICCChallenge function defined in section 5.7.7 of SGP.22 [4], where the IPA plays the role of LPA.

### 5.9.4 Function (ES10b): AddInitialEim

**Related Procedures:** Addition of initial eIM by IPAd

**Function Provider Entity:** eUICC (ISD-R)

#### **Description:**

This function is used by the IPAd to store eIM Configuration Data to the eUICC in case the eUICC does not contain any eIM Configuration Data.

Upon reception of this function call, the eUICC SHALL:

- Check if any eIM Configuration Data is already present in the eUICC. If so, the eUICC SHALL return an error code `associatedEimAlreadyExists`.
- Check if `counterValue` exceeds the maximum value supported by the eUICC. If so, the eUICC SHALL return the error code `counterValueOutOfRange`.
- Check if the eUICC has enough memory to store the list of provided eIM Configuration Data. If not, the eUICC SHALL return an error code `insufficientMemory`.
- Check that the sub-field `eimId` of `EimConfigurationData` in each entry of the `eimConfigurationDataList` is present and that its length does not exceed the allowed maximum value. Otherwise, the eUICC SHALL return an error code `commandError`.
- Check that the sub-field `euiccCiPKId` of `EimConfigurationData` in each entry of the `eimConfigurationDataList`, if present, is a valid entry within

`euiccCiPKIdListForSigning` in `eUICCInfo2`. If not, the eUICC SHALL return an error code `ciPKUnknown`.

- If requested for an Associated eIM, calculate an `associationToken`. If `associationToken` is not set to -1 the eUICC SHALL return an error code `invalidAssociationToken`.
- Store the provided list of eIM Configuration Data, including the `associationToken` (if requested).
- Return the `associationToken` for each Associated eIM where it is requested.

NOTE: Any further addition of eIM Configuration Data can be performed by an Associated eIM using signed eUICC Packages containing eCO(s) as defined in section 3.5.1.1.

The function SHALL be performed in an atomic way, meaning that in case of any error during the function execution, the command SHALL stop and SHALL leave the eIM Configuration Data in their original state prior to function execution.

### **Command Data**

The command data SHALL be coded as follows:

```
-- ASN1START
AddInitialEimRequest ::= [87] SEQUENCE { -- Tag 'BF57'
    eimConfigurationDataList [0] SEQUENCE OF EimConfigurationData
}
-- ASN1STOP
```

The following sub-fields of `EimConfigurationData` SHALL be present in each entry of the `eimConfigurationDataList`:

- `eimId`;
- `counterValue`; and
- either `eimPublicKey` or `eimCertificate`.

The `eimIdType`, the `associationToken` (with the value set to -1 to request the calculation of an association token for the new Associated eIM) and `eimFqdn` are optional for the IPA to provide in each `EimConfigurationData`. If the `eimIdType` is not provided, the eUICC SHALL assign the value `eimIdTypeProprietary` before storing the eIM Configuration Data.

The `eimSupportedProtocol` is optional for the IPA to provide in `EimConfigurationData`. If not provided, the eUICC SHALL assign the value where only the `eimProprietary` bit is set before storing the eIM Configuration Data.

The `euiccCiPKId` is optional for the IPA to provide in `EimConfigurationData`. If not provided, the eUICC SHALL assign the value of first entry of `euiccCiPKIdListForSigning` in `eUICCInfo2` before storing the eIM Configuration Data.

The `trustedPublicKeyDataTls` is optional for the IPAd to provide in each `EimConfigurationData` data object.

NOTE: It is IPA's responsibility to correctly formulate each entry, including the uniqueness of `eimId`, and the eUICC is not mandated to check the validity.

### **Response Data**

The response data SHALL be coded as follows:

```
-- ASN1START
AddInitialEimResponse ::= [87] CHOICE { -- Tag 'BF57'
  addInitialEimOk SEQUENCE OF CHOICE {
    associationToken [4] INTEGER,
    addOk NULL
  },
  addInitialEimError INTEGER {
    insufficientMemory(1),
    associatedEimAlreadyExists (2),
    ciPKUnknown(3),
    invalidAssociationToken(5),
    counterValueOutOfRange(6),
    commandError(7),
    undefinedError(127)
  }
}
-- ASN1STOP
```

## **5.9.5 Function (ES10b): eUICCMemoryReset**

**Related Procedures:** eUICC Memory Reset

**Function Provider Entity:** ISD-R (IPA Services)

### **Description:**

NOTE: ES10c.eUICCMemoryReset function defined in SGP.22 [4] has a tag number 'BF34'.

This function deletes selected subsets of the Profiles stored on an eUICC regardless of their enabled status or any Profile Policy Rules. The following subsets are defined:

- Operational Profiles
- Test Profiles that were not pre-installed (i.e., Test Profiles that were "field loaded")
- Test Profiles that were pre-installed (i.e., Test Profiles that were loaded in the factory)
- Provisioning Profiles

NOTE: The identification of pre-installed Test Profiles is out of the scope of this specification.

The eUICC returns a status in `resetResult` indicating whether any Profiles were deleted.

- The eUICC returns `ok` if at least one Profile is deleted.
- The eUICC returns `nothingToDelete` if no Profile is deleted.

This function can also be used to:

- Reset the default SM-DP+ address to its initial value. In this context both `ok` and `nothingToDelete` response in `resetResult` indicates a successful execution of resetting the default SM-DP+ address.

- Remove all eIM Configuration Data. The execution result of this operation is returned in `resetEimResult` as described below.
- Reset the configuration of the immediate Profile enabling. The execution result of this operation is returned in `resetImmediateEnableConfigResult` as described below.

Any combination MAY be specified in the request.

If the eUICC does not support Test Profiles, then a request to delete them is ignored.

The eUICC Memory Reset SHALL be performed in an atomic and non-reversible way in case of external interruptions (e.g., power loss): the eUICC SHALL continue the processing of that command upon the next eUICC power on. In case of any other error during the function execution, the command SHALL stop and SHALL leave the eUICC and the involved Profiles in their original states prior to function execution.

NOTE: How this function is protected from misuse is implementation specific.

Upon reception of this function call, the eUICC SHALL:

- If the Emergency Profile is enabled, terminate the command and return error code `ecallActive`.
- If the `eUICCMemoryReset` function call indicates at least one of `deleteOperationalProfiles`, `deleteFieldLoadedTestProfiles`, `deletePreLoadedTestProfiles` or `deleteProvisioningProfiles`, perform the following:
  - If there is an Enabled Profile with a proactive session ongoing, the eUICC SHALL perform one of the following:
    - terminate the command and return error code `catBusy`.
    - internally terminate the proactive session and prepare to send the REFRESH proactive command as the next proactive command. If a TERMINAL RESPONSE is still outstanding, the REFRESH proactive command SHALL only be sent after reception of the TERMINAL RESPONSE.

NOTE: In case of the `catBusy` error, the Device MAY take implementation-dependent action to terminate the proactive command session, and MAY send the eUICC Memory Reset command again without any further End User interaction.

- Delete all the selected ISD-Ps with their Profiles regardless of their enabled status or Profile Policy Rules and all related Profiles Metadata stored in the ISD-R and prepare `resetResult`.
- Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).
- If there is any information recorded of the previously enabled Profile that was enabled prior to enabling the Fallback Profile, the eUICC SHALL remove this recorded information.

- For each deleted Profile, the eUICC SHALL generate as many Notifications as configured in its Profile Metadata (notificationConfigurationInfo) in the format of OtherSignedNotification.
- If the eUICCMemoryReset function call indicates resetDefaultSmdpAddress: reset the default SM-DP+ address to its default value.
- If the eUICCMemoryReset function call indicates resetEimConfigData: remove all eIM Configuration Data of all Associated eIMs and prepare resetEimResult.
- If there is no eIM Configuration Data to be removed, the eUICC SHALL return error code nothingToDelete.
- If the command is not supported by the eUICC, the eUICC SHALL return error code eimResetNotSupported.
- If the eUICCMemoryReset function call indicates resetImmediateEnableConfig: reset the configuration of the immediate Profile enabling by deactivating immediate Profile enabling and removing any immediate enabling data for use with immediate Profile enabling. The eUICC SHALL also prepare resetImmediateEnableConfigResult.
  - If the command is not supported by the eUICC, the eUICC SHALL return error code resetIECNotSupported.
- Return EuiccMemoryResetResponse including resetResult. In addition, if resetResult indicates either ok or nothingToDelete:
  - If resetEimConfigData bit was set in the command data, return EuiccMemoryResetResponse additionally including resetEimResult.
  - If resetImmediateEnableConfig bit was set in the command data, return EuiccMemoryResetResponse additionally including resetImmediateEnableConfig.
- If an Enabled Profile was deleted, the ISD-R SHALL send a REFRESH proactive command to the Device with mode "UICC Reset".

### **Command Data**

The command data SHALL be coded as follows:

```
-- ASN1START
EuiccMemoryResetRequest ::= [100] SEQUENCE { -- Tag 'BF64'
  resetOptions [2] BIT STRING {
    deleteOperationalProfiles (0),
    deleteFieldLoadedTestProfiles (1),
    resetDefaultSmdpAddress (2),
    deletePreLoadedTestProfiles (3),
    deleteProvisioningProfiles (4),
    resetEimConfigData (5),
    resetImmediateEnableConfig (6)
  }
}
-- ASN1STOP
```

### **Response Data**

The response data SHALL be coded as follows:

```
-- ASN1START
EuiccMemoryResetResponse ::= [100] SEQUENCE { -- Tag 'BF64'
```

```
    resetResult INTEGER {ok(0), nothingToDelete(1), catBusy(5), ecallActive(104),
undefinedError(127)},
    resetEimResult INTEGER {ok(0), nothingToDelete(1), eimResetNotSupported(2),
undefinedError(127)} OPTIONAL,
    resetImmediateEnableConfigResult INTEGER {ok(0), resetIECNotSupported(1),
undefinedError(127)} OPTIONAL
}
-- ASN1STOP
```

### 5.9.6 Function (ES10b): AuthenticateServer

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

#### Description:

This function is identical to the ES10b.AuthenticateServer function defined in section 5.7.13 of SGP.22 [4], where the IPA plays the role of LPA and the EUICCInfo2 in AuthenticateResponseOk is according to Section 5.9.2 of this specification. The following additional eUICC behaviour applies after attaching the received RSP Server Certificate in the RSP session context:

- If immediate Profile enabling is activated, the eUICC SHALL, in case the MatchingID is missing in CtxParams1, verify that the OID in the RSP Server Certificate and FQDN in serverSigned1 match to the default SM-DP+ data (OID and FQDN) stored in the eUICC. If the verification is successful, the eUICC SHALL grant immediate enabling of the Profile. Regardless of the verification result, the eUICC continues the processing of this function by generating euiccSigned1.

### 5.9.7 Function (ES10b): PrepareDownload

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

#### Description:

This function is identical to the ES10b.PrepareDownload function defined in section 5.7.5 of SGP.22 [4], where the IPA plays the role of LPA.

### 5.9.8 Function (ES10b): LoadBoundProfilePackage

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

#### Description:

This function is identical to the ES10b.LoadBoundProfilePackage function defined in section 5.7.6 of SGP.22 [4], where the IPA plays the role of LPA.

### 5.9.9 Function (ES10b): CancelSession

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

### Description:

This function is identical to the ES10b.CancelSession function defined in section 5.7.14 of SGP.22 [4], where the IPA plays the role of LPA, with the following additional eUICC behaviour:

- The eUICC SHALL revoke any grant to immediately enable a Profile.

NOTE: The following values of CancelSessionReason are not used in this specification: endUserRejection(0), postponed(1), timeout(2).

### 5.9.10 Function (ES10b): GetCerts

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

### Description:

This function is used by the IPA to retrieve the eUICC Certificate and the EUM Certificate from the eUICC. This function can be used at any time by the IPA.

In order to select between multiple of EUM certificates and eUICC certificates within the eUICC the IPA MAY provide the CI Public Key Identifier (euiccCiPKId). If euiccCiPKId is not provided, the first entry of euiccCiPKIdListForSigning in eUICCInfo1 / eUICCInfo2 is used as euiccCiPKId when selecting the EUM certificate and/or eUICC certificate to return.

### Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
GetCertsRequest ::= [86] SEQUENCE { -- Tag 'BF56'
    euiccCiPKId SubjectKeyIdentifier OPTIONAL -- CI Public Key Identifier supported on
the eUICC for signature creation
}
-- ASN1STOP
```

### Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
GetCertsResponse ::= [86] CHOICE { -- Tag 'BF56'
    certs SEQUENCE {
        eumCertificate [5] Certificate, -- Tag 'A5'
        euiccCertificate [6] Certificate -- Tag 'A6'
    },
    getCertsError INTEGER {invalidCiPKId(1), undefinedError(127)}
}
-- ASN1STOP
```

### 5.9.11 Function (ES10b): RetrieveNotificationsList

**Related Procedures:** Generic eUICC Package Download and Execution

**Function Provider Entity:** ISD-R (IPA Services)

## Description:

This function retrieves the list of pending Notifications for installed Profiles and/or eUICC Package Results.

## Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
RetrieveNotificationsListRequest ::= [43] SEQUENCE { -- Tag 'BF2B'
  searchCriteria CHOICE {
    seqNumber [0] INTEGER,
    profileManagementOperation [1] NotificationEvent,
    euiccPackageResults [2] NULL
  } OPTIONAL
}
-- ASN1STOP
```

The `searchCriteria` data object can be used to filter the list of Notifications or eUICC Package Results that the eUICC SHALL return.

- `seqNumber` indicates that the eUICC SHALL return either a Notification or an eUICC Package Result that has the sequence number.
- A bit set to 1 in the `profileManagementOperation` indicates that the eUICC SHALL return all the Notifications corresponding to this type. The type `notificationInstall` SHALL include `ProfileInstallationResult`.
- `euiccPackageResults` indicates that the eUICC SHALL return all the eUICC Package Results.

If `searchCriteria` data object is omitted, the eUICC SHALL return all stored Notifications that are in the format of `PendingNotificationList`.

## Response Data

The response data SHALL contain the list of `PendingNotification` data objects or `EuiccPackageResult` data objects. The list SHALL be filtered according to the command data. The eUICC MAY provide the Notifications and/or eUICC Package Results in any order within each list. The list SHALL be empty if there are no pending Notifications or eUICC Package Results matching the filtering criteria.

The following is the definition of the `RetrieveNotificationsListResponse` data object

```
-- ASN1START
RetrieveNotificationsListResponse ::= [43] CHOICE { -- Tag 'BF2B'
  notificationList PendingNotificationList, -- Tag 'A0'
  notificationsListResultError INTEGER { undefinedError(127)}, -- Tag '81'
  euiccPackageResultList EuiccPackageResultList -- Tag 'A2'
}
-- ASN1STOP
```

The `PendingNotification` is either a `ProfileInstallationResult` or an `OtherSignedNotification` according to definition in section 5.7.10 of SGP.22 [4].

### **5.9.12 Function (ES10b): RemoveNotificationFromList**

**Related Procedures:** Profile Download and Installation, Generic eUICC Package Download and Execution

**Function Provider Entity:** ISD-R (IPA Services)

#### **Description:**

This function is identical to the ES10b.RemoveNotificationFromList function defined in section 5.7.11 of SGP.22 [4], where the IPA plays the role of LPA, with the addition that besides Notifications the function is also used to delete eUICC Package Results. Hence, the provided `seqNumber` in `NotificationSentRequest` provided by IPA is either a `seqNumber` of a Notification or a `seqNumber` of an eUICC Package Result.

### **5.9.13 Function (ES10b): GetRAT**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

#### **Description:**

This function is identical to the ES10b.GetRAT function defined in section 5.7.22 of SGP.22 [4], where the IPA plays the role of LPA.

### **5.9.14 Function (ES10b): GetProfilesInfo**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

#### **Description:**

This function is based on the ES10c.GetProfilesInfo function defined in section 5.7.15 of SGP.22 [4], where the IPA plays the role of LPA.

The ProfileInfo data object as defined in SGP.22[4] SHALL be replaced by the definition of the section 4.4 of this document. The section 4.4 describes the additional tags.

**NOTE:** In this version of the specification, `iotSpecificMetadataSupport` bit in `RspCapability` is set to '0', therefore tag `iotSpecificTagList` need not to be supported as described in SGP.22 [4].

### **5.9.15 Function (ES10b): ImmediateEnable**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

#### **Description:**

This function is used by the IPA to request immediate Profile enabling without eIM involvement. The eUICC MAY be configured to support the immediate enabling.

On reception of this function call, the eUICC SHALL:

- Verify that, except for Notification retrieval commands, the previous ES10 function was an ES10b.LoadBoundProfilePackage. If not, the eUICC SHALL return an error code `noSessionContext`.
- Verify that immediate enabling was granted for the newly installed Profile. If not, the eUICC SHALL return an error code `immediateEnableNotAvailable`.
- Verify that the target Profile is not an Emergency Profile. If it is an Emergency Profile with `ecallIndication` set to TRUE, the eUICC SHALL return an error code `immediateEnableNotAvailable`.

If the `refreshFlag` is not set, the eUICC SHALL then:

- Check whether there is a proactive session ongoing (which the Device did not terminate). If so:
  - the eUICC MAY terminate the ImmediateEnable command with error `'catBusy'`.
  - If the eUICC does not terminate the ImmediateEnable command with error `'catBusy'`, the eUICC SHALL internally terminate the proactive session and ignore any incoming TERMINAL RESPONSE from that proactive session.
- Close all logical channels which still have an application of the currently Enabled Profile selected (which the IoT Device did not close), without generating an error.
- Reset the PIN status.
- Disable the currently Enabled Profile (if any).
- Enable the downloaded Profile.
- Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).
- Implicitly select the MF on the basic logical channel.
- Return `'ok'` to the IPA.

If the `refreshFlag` is set:

- If there is a proactive session ongoing;
  - The eUICC MAY terminate the ImmediateEnable command with error `'catBusy'`.
  - If the eUICC does not terminate the ImmediateEnable command with error `'catBusy'`, the eUICC SHALL internally terminate the proactive session and send the REFRESH command as the next proactive command. If a TERMINAL RESPONSE is still outstanding, the REFRESH command SHALL only be sent after reception of the TERMINAL RESPONSE.
- Otherwise, the eUICC SHALL:
  - Mark the currently Enabled Profile as "to be disabled" (if any).
  - Mark the target Profile as "to be enabled".
  - Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).
  - Return `'ok'` to the IPA.
  - Send the REFRESH command in "eUICC Profile State Change" mode (if supported by the IoT Device) or "UICC Reset" mode to the IoT Device according to ETSI TS 102 223 [5].

- Upon reception of the Terminal Response with result "command performed successfully" or upon reset of the eUICC, disable the currently Enabled Profile (if any) and enable the downloaded Profile.
- Upon reception of the Terminal Response with result "temporary problem with executing command" or "permanent problem with executing command", the eUICC SHALL NOT disable the currently Enabled Profile (if any) nor enable the downloaded Profile. Subsequent actions are implementation specific.

### **Command Data**

The command data SHALL be encoded as follows:

```
-- ASN1START
ImmediateEnableRequest ::= [90] SEQUENCE { -- Tag 'BF5A'
    refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
-- ASN1STOP
```

### **Response Data**

The response data SHALL be encoded as follows:

```
-- ASN1START
ImmediateEnableResponse ::= [90] SEQUENCE { -- Tag 'BF5A'
    immediateEnableResult [0] INTEGER {
        ok(0),
        immediateEnableNotAvailable(1),
        noSessionContext(4),
        catBusy(5),
        undefinedError(127)
    }
}
-- ASN1STOP
```

## **5.9.16 Function (ES10b): ProfileRollback**

**Related Procedures:** Generic eUICC Package Download and Execution, Profile State Management.

**Function Provider Entity:** ISD-R (IPA Services)

### **Description:**

This function is used by the IPA to request to roll back to the previously Enabled Profile, if any. If the usage of the Rollback Mechanism had been previously granted by the eIM (i.e., in the last enable PSMO), this function disables the currently Enabled Profile and enables the Profile, if any, that was enabled before the processing of the last eUICC Package. This SHALL be performed in an atomic way, meaning that in case of any error during the function execution, the command SHALL stop and SHALL leave the involved Profiles in their original states (prior to function execution).

**NOTE:** Before calling the ProfileRollback function with the `refreshFlag` not being set, the IoT Device has the responsibility to ensure that the relevant conditions for use are met, as indicated for the ES10c.EnableProfile function in section 3.2.1 of SGP.22 [4].

In the following procedure, the target Profile designates the Profile (if any) that was enabled before the processing of the last eUICC Package, and which is to be re-enabled.

Upon reception of the ProfileRollback command, the eUICC SHALL:

- Check whether the usage of the Rollback Mechanism had been previously granted by the eIM. If not, the procedure SHALL stop, and the result of this function SHALL indicate an error ('rollbackNotAllowed').
- Check whether a Profile is currently enabled. If not, the procedure SHALL stop, and the result of this function shall indicate an error ('commandError').

NOTE: If the Fallback Mechanism has recorded information on the currently Enabled Profile, it remains unchanged by the ProfileRollback Mechanism.

If the `refreshFlag` is not set, the eUICC SHALL:

- Check whether there is a proactive session ongoing (which the Device did not terminate). If so:
  - the eUICC MAY terminate the "ES10b.ProfileRollback" function with error 'catBusy'.
  - If the eUICC does not terminate the "ES10b.ProfileRollback" function with error 'catBusy', the eUICC SHALL internally terminate the proactive session and ignore any incoming TERMINAL RESPONSE from that proactive session.
- Close all logical channels which still have an application of the currently Enabled Profile selected (which the Device did not close), without generating an error.
- Reset the PIN status.
- Disable the currently Enabled Profile.
- Enable the target Profile (if any).
- Implicitly select the MF on the basic logical channel.
- Generate a new eUICC Package Result including an additional ProfileRollbackResult (see Response Data below) and remove the previous eUICC Package Result (which is now obsolete).
- Return 'ok' and the newly generated eUICC Package Result to the IPA.

NOTE: For subsequent actions by the Device following the enablement of the target Profile, see ES10c.EnableProfile function in section 3.2.1 of SGP.22 [4].

If the `refreshFlag` is set:

- If there is a proactive session ongoing;
  - The eUICC MAY terminate the "ES10b.ProfileRollback" function with error 'catBusy'.
  - If the eUICC does not terminate the "ES10b.ProfileRollback" function with error 'catBusy', the eUICC SHALL internally terminate the proactive session and send the REFRESH command as the next proactive command. If a TERMINAL RESPONSE is still outstanding, the REFRESH command SHALL only be sent after reception of the TERMINAL RESPONSE.
- Otherwise, the eUICC SHALL:
  - Mark the currently Enabled Profile as "to be disabled".
  - Mark the target Profile (if any) as "to be enabled".

- Generate a new eUICC Package Result including an additional ProfileRollbackResult (see Response Data below) and remove the previous eUICC Package Result (which is now obsolete).
- Return 'ok' and the newly generated eUICC Package Result to the IPA.
- Send the REFRESH command in "eUICC Profile State Change" mode (if supported by the Device) or "UICC Reset" mode to the Device according to ETSI TS 102 223 [5].
- Upon reception of the Terminal Response with result "command performed successfully" or upon reset of the eUICC, disable the currently Enabled Profile and enable the target Profile (if any).
- Upon reception of the Terminal Response with result "temporary problem with executing command" or "permanent problem with executing command", the eUICC SHALL NOT disable the currently Enabled Profile nor Enable the target Profile (if any). Subsequent actions are implementation specific.

The eUICC MAY start a new proactive UICC session only after a new TERMINAL PROFILE command is executed.

- If the currently Enabled Profile was successfully disabled, the eUICC SHALL generate as many Notifications as configured in its metadata (notificationConfigurationInfo) in the format of OtherSignedNotification.
- If the target Profile is successfully enabled, the eUICC SHALL generate as many Notifications as configured in its metadata (notificationConfigurationInfo) in the format of OtherSignedNotification.

### **Command Data**

The command data SHALL be coded as follows:

```
-- ASN1START
ProfileRollbackRequest ::= [88] SEQUENCE { -- Tag 'BF58'
  refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
-- ASN1STOP
```

### **Response Data**

The response data SHALL be coded as follows:

```
-- ASN1START
ProfileRollbackResponse ::= [88] SEQUENCE { -- Tag 'BF58'
  cmdResult INTEGER {
    ok(0),
    rollbackNotAllowed(1), -- Usage of rollback was not granted by the eIM
    catBusy(5),
    commandError(7),
    undefinedError(127)
  },
  eUICCPackageResult [81] EuiccPackageResult OPTIONAL
}
-- ASN1STOP
```

The eUICCPackageResult SHALL be present if the cmdResult is 'ok' (i.e., a Rollback Mechanism has been executed by the eUICC). In this case, the eUICC SHALL:

- Create the `eUICCPackageResult` by including:
  - The same `eimId`, `counterValue`, `eimTransactionId` as the result generated for the previous eUICC Package;
  - The `seqNumber` value as the eUICC sequence number (not yet used); and
  - `eUICCResultData` with the same PSMO results as the result generated for the previous eUICC Package followed by an additional `RollbackProfileResult` (see section 2.11.2.1).
- Increment the sequence number in eUICC storage.

### 5.9.17 Function (ES10b): ConfigureImmediateProfileEnabling

**Related Procedures:** Configure by IPA of Immediate Profile Enabling.

**Function Provider Entity:** eUICC (ISD-R)

#### Description:

This function is used by the IPA to activate or deactivate immediate Profile enabling in the eUICC and to add or update immediate enabling data used in the immediate Profile enabling.

Upon reception of this function call, the eUICC SHALL:

- Check if any eIM Configuration Data is already present in the eUICC. If so, the eUICC SHALL return an error code `associatedEimAlreadyExists`.
- Check if `immediateEnableFlag` is present in the command data. If `immediateEnableFlag` is present and immediate Profile enabling is not activated, the eUICC SHALL activate immediate Profile enabling. If `immediateEnableFlag` is not present and immediate Profile enabling is activated, the eUICC SHALL deactivate immediate Profile enabling. In all other cases, the immediate Profile enabling state is left unchanged.
- Check if default SM-DP+ data (`defaultSmdpOid` and/or `defaultSmdpAddress`) is present in the command data. If so, and if the eUICC has enough memory to store it, the eUICC SHALL store it for use in the immediate Profile enabling. If there is not enough memory, the eUICC SHALL return an error code `insufficientMemory`.

The function SHALL be performed in an atomic way, meaning that in case of any error during the function execution, the command SHALL stop and SHALL leave the immediate Profile enabling configuration in its original state prior to function execution.

#### **Command Data**

The command data SHALL be coded as follows:

```
-- ASN1START
ConfigureImmediateProfileEnablingRequest ::= [89] SEQUENCE { -- Tag 'BF59'
    immediateEnableFlag [0] NULL OPTIONAL,
    defaultSmdpOid [1] OBJECT IDENTIFIER OPTIONAL,
    defaultSmdpAddress [2] UTF8String OPTIONAL
}
-- ASN1STOP
```

#### **Response Data**

The response data SHALL be coded as follows:

```
-- ASN1START
ConfigureImmediateProfileEnablingResponse ::= [89] SEQUENCE { -- Tag 'BF59'
  configImmediateEnableResult [0] INTEGER {
    ok(0),
    insufficientMemory(1),
    associatedEimAlreadyExists(2),
    undefinedError(127)
  }
}
-- ASN1STOP
```

### 5.9.18 Function (ES10b): GetEimConfigurationData

**Related Procedures:** Reading eIM Configuration Data by IPA

**Function Provider Entity:** eUICC (ISD-R)

#### **Description:**

This function is used by the IPA to read eIM Configuration Data stored in the eUICC.

#### **Command Data**

The command data SHALL be coded as follows:

```
-- ASN1START
GetEimConfigurationDataRequest ::= [85] SEQUENCE { -- Tag 'BF55'
  searchCriteria CHOICE {
    eimId [0] UTF8String (SIZE(1..128))
  } OPTIONAL
}
-- ASN1STOP
```

The searchCriteria data object can be used to filter the list of eIM Configuration Data that the eUICC SHALL return.

- eimId is the eIM identifier defined in the section 4.3 of this document. It indicates that the eUICC SHALL return the eIM Configuration Data having the specified eimId.

If searchCriteria data object is omitted, the eUICC SHALL return all stored eIM Configuration Data.

#### **Response Data**

The response data SHALL be coded as follows:

```
-- ASN1START
GetEimConfigurationDataResponse ::= [85] SEQUENCE { -- Tag 'BF55'
  eimConfigurationDataList [0] SEQUENCE OF EimConfigurationData
}
-- ASN1STOP
```

In each EimConfigurationData entry of the eimConfigurationDataList, the eUICC SHALL provide eimId, eimFqdn (if present), eimIdType, trustedPublicKeyDataTls (if present), eimSupportedProtocol, euiccCipKeyId and, if configured, associationToken, and MAY additionally provide either eimPublicKey or eimCertificate. The eUICC SHALL NOT provide counterValue. If the

`associationToken` is provided, it SHALL be the token generated by the eUICC during the association process between the eUICC and that specific eIM.

If there is no eIM Configuration Data stored in the eUICC, or if there is no eIM Configuration Data matching to the `searchCriteria` (if received), the eUICC SHALL return an empty list in the `eimConfigurationDataList`.

### 5.9.19 Function (ES10b): GetEID

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** ISD-R (IPA Services)

#### Description:

This function is identical to the ES10c.GetEID function defined in section 5.7.20 of SGP.22 [4], where the IPA plays the role of LPA.

### 5.9.20 Function (ES10b): ExecuteFallbackMechanism

**Function Provider:** ISD-R (IPA Services)

**Function Caller:** IPA

#### Description:

This function is OPTIONAL for the eUICC and MAY be used by the IPA to instruct the eUICC to enable the Fallback Profile and to disable the currently Enabled Profile.

This function SHALL be performed in an atomic way, meaning that in case of any error during the function execution, the command SHALL stop and SHALL leave the involved Profiles in their original states (prior to function execution).

Upon reception of this function, the ISD-R SHALL:

- verify that there is an enabled Profile. If no, the procedure SHALL stop and the result of this command SHALL indicate error ('`commandError(7)`').
- verify that the `fallbackAttribute` is set for a Profile on the eUICC. If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('`fallbackNotAvailable(6)`').
- verify that the Emergency Profile is disabled. If it is enabled, the procedure SHALL stop and the result of this command SHALL indicate error ('`ecallActive(104)`').
- verify that the Fallback Profile is disabled. If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('`profileNotInDisabledState(2)`').

If the `refreshFlag` is not set, the eUICC SHALL:

- Check whether there is a proactive session ongoing (which the Device did not terminate). If so:
  - the eUICC MAY terminate the "ES10b.ExecuteFallbackMechanism" function with error '`catBusy`'.

- If the eUICC does not terminate the "ES10b.ExecuteFallbackMechanism" function with error 'catBusy', the eUICC SHALL internally terminate the proactive session and ignore any incoming TERMINAL RESPONSE from that proactive session.
- Close all logical channels which still have an application of the currently Enabled Profile selected (which the Device did not close), without generating an error.
- Reset the PIN status.
- Record information for identification of the currently Enabled Profile
- Disable the currently Enabled Profile.
- Enable the Fallback Profile.
- Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).
- Implicitly select the MF on the basic logical channel.
- Return 'ok' to the IPA.

If the `refreshFlag` is set:

- If there is a proactive session ongoing;
  - The eUICC MAY terminate the "ES10b.ExecuteFallbackMechanism" function with error 'catBusy'.
  - If the eUICC does not terminate the "ES10b.ExecuteFallbackMechanism" function with error 'catBusy', the eUICC SHALL internally terminate the proactive session and send the REFRESH command as the next proactive command. If a TERMINAL RESPONSE is still outstanding, the REFRESH command SHALL only be sent after reception of the TERMINAL RESPONSE.
- Otherwise, the eUICC SHALL:
  - Mark the currently Enabled Profile as "to be disabled".
  - Mark the target Fallback Profile as "to be enabled".
  - Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).
  - Return 'ok' to the IPA.
  - Send the REFRESH command in "eUICC Profile State Change" mode (if supported by the IoT Device) or "UICC Reset" mode to the IoT Device according to ETSI TS 102 223 [5].
  - Upon reception of the Terminal Response with result "command performed successfully" or upon reset of the eUICC, record information for identification of the currently Enabled Profile, disable that currently Enabled Profile and enable the target Fallback Profile.
  - Upon reception of the Terminal Response with result "temporary problem with executing command" or "permanent problem with executing command", the eUICC SHALL NOT disable the currently Enabled Profile nor Enable the targeted Fallback Profile. Subsequent actions are implementation specific.

NOTE: Recorded information on the currently Enabled Profile is used when ES10b.ReturnFromFallback is executed.

The eUICC MAY start a new proactive UICC session only after a new TERMINAL PROFILE command is executed.

- If the currently Enabled Profile was successfully disabled, the eUICC SHALL generate as many Notifications as configured in its metadata (notificationConfigurationInfo) in the format of OtherSignedNotification.
- If the Fallback Profile is successfully enabled, the eUICC SHALL generate as many Notifications as configured in its metadata (notificationConfigurationInfo) in the format of OtherSignedNotification.

### **Command Data**

The command data SHALL be coded as follows:

```
-- ASN1START
ExecuteFallbackMechanismRequest ::= [93] SEQUENCE { -- Tag 'BF5D'
  refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
-- ASN1STOP
```

### **Response Data**

The response data SHALL be coded as follows:

```
-- ASN1START
ExecuteFallbackMechanismResponse ::= [93] SEQUENCE { -- Tag ' BF5D'
  executeFallbackMechanismResult [0] INTEGER {
    ok(0),
    profileNotInDisabledState(2),
    catBusy(5),
    fallbackNotAvailable(6), -- Fallback Attribute not set
    commandError(7),
    ecallActive(104),
    undefinedError(127)
  }
}
-- ASN1STOP
```

NOTE: It's IoT Device implementation specific to decide when the IPA calls the ReturnFromFallback function.

## **5.9.21 Function (ES10b): ReturnFromFallback**

**Function Provider:** ISD-R (IPA Services)

**Function Caller:** IPA

### **Description:**

This function is OPTIONAL for the eUICC and MAY be used by the IPA to instruct the eUICC to return from the Fallback Profile. It disables the currently enabled Fallback Profile and enables the Profile that was previously enabled prior to enabling the Fallback Profile. If "ES10b.ExecuteFallbackMechanism" function is supported, this function SHALL also be supported.

This function SHALL be performed in an atomic way, meaning that in case of any error during the function execution, the command SHALL stop and SHALL leave the involved Profiles in their original states (prior to function execution).

Upon reception of this function, the ISD-R SHALL:

- check if the currently enabled Profile is the Fallback Profile. If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('fallbackNotAvailable(6)').
- Check if information for the identification of the previously enabled Profile has been recorded. If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('fallbackNotAvailable(6)').
- Check if the previously enabled Profile is installed (e.g. it was not deleted using ES10b.eUICCMemoryReset). If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('fallbackNotAvailable(6)').
- Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).

If the `refreshFlag` is not set, the eUICC SHALL:

- Check whether there is a proactive session ongoing (which the Device did not terminate). If so:
  - the eUICC MAY terminate the “ES10b.ReturnFromFallback” function with error 'catBusy'.
  - If the eUICC does not terminate the “ES10b.ReturnFromFallback” function with error 'catBusy', the eUICC SHALL internally terminate the proactive session and ignore any incoming TERMINAL RESPONSE from that proactive session.
- Close all logical channels which still have an application of the currently Enabled Fallback Profile selected (which the Device did not close), without generating an error.
- Reset the PIN status.
- Disable the currently enabled Fallback Profile.
- Enable the Profile that was previously enabled (prior to enabling the Fallback Profile).
- Implicitly select the MF on the basic logical channel.
- Return 'ok' to the IPA.

If the `refreshFlag` is set:

- If there is a proactive session ongoing;
  - The eUICC MAY terminate the “ES10b.ReturnFromFallback” function with error 'catBusy'.
  - If the eUICC does not terminate the “ES10b.ReturnFromFallback” function with error 'catBusy', the eUICC SHALL internally terminate the proactive session and send the REFRESH command as the next proactive command. If a TERMINAL RESPONSE is still outstanding, the REFRESH command SHALL only be sent after reception of the TERMINAL RESPONSE.
- Otherwise, the eUICC SHALL:
  - Mark the currently enabled Fallback Profile as "to be disabled".
  - Mark the previously enabled Profile as "to be enabled".
  - Return 'ok' to the IPA.
  - Send the REFRESH command in "eUICC Profile State Change" mode (if supported by the IoT Device) or "UICC Reset" mode to the IoT Device according to ETSI TS 102 223 [5].

- Upon reception of the Terminal Response with result "command performed successfully" or upon reset of the eUICC, disable the currently enabled Fallback Profile and enable the previously enabled Profile.
- Upon reception of the Terminal Response with result "temporary problem with executing command" or "permanent problem with executing command", the eUICC SHALL NOT disable the currently enabled Fallback Profile nor enable the previously enabled Profile. Subsequent actions are implementation specific.

The eUICC MAY start a new proactive UICC session only after a new TERMINAL PROFILE command is executed.

- If the currently enabled Fallback Profile was successfully disabled, the eUICC SHALL generate as many Notifications as configured in its metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.
- If the previously enabled Profile is successfully enabled, the eUICC SHALL generate as many Notifications as configured in its metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.

### **Command Data**

The command data SHALL be coded as follows:

```
-- ASN1START
ReturnFromFallbackRequest ::= [94] SEQUENCE { -- Tag 'BF5E'
  refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
-- ASN1STOP
```

### **Response Data**

The response data SHALL be coded as follows:

```
-- ASN1START
ReturnFromFallbackResponse ::= [94] SEQUENCE { -- Tag 'BF5E'
  returnFromFallbackResult [0] INTEGER {
    ok(0),
    catBusy(5),
    fallbackNotAvailable(6), -- Fallback Attribute not set
    commandError(7),
    undefinedError(127)
  }
}
-- ASN1STOP
```

## **5.9.22 Function (ES10b): EnableEmergencyProfile**

**Function Provider:** eUICC

### **Description:**

This function is OPTIONAL for eUICC and IPA, and MAY be used by the IPA to instruct the eUICC to enable the Emergency Profile with `ecallIndication` set to TRUE.

This function disables the currently Enabled Profile and enables the Emergency Profile. This function SHALL be performed in an atomic way, meaning that in case of any error during the

function execution, the command SHALL stop and SHALL leave the involved Profiles in their original states (prior to function execution).

Upon enabling the Emergency Profile, the eUICC SHALL NOT generate any Notifications.

The Emergency Profile SHALL remain enabled until the IoT Device disables it again using the ES10b.DisableEmergencyProfile function, i.e. it remains enabled even after restart of the IoT Device.

Upon reception of this function, the eUICC SHALL:

- check if the Emergency feature is supported. If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('ecallNotAvailable').
- verify that the Emergency Profile (Profile with the `ecallIndication` set to TRUE) is available. If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('ecallNotAvailable').
- verify that the selected Emergency Profile is disabled. If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('profileNotInDisabledState').

If the `refreshFlag` is not set, the eUICC SHALL:

- Check whether there is a proactive session ongoing (which the Device did not terminate). If so:
  - the eUICC MAY terminate the “ES10b.EnableEmergencyProfile” function with error 'catBusy'.
  - If the eUICC does not terminate the “ES10b.EnableEmergencyProfile” function with error 'catBusy', the eUICC SHALL internally terminate the proactive session and ignore any incoming TERMINAL RESPONSE from that proactive session.
- Close all logical channels which still have an application of the currently Enabled Profile selected (which the IoT Device did not close), without generating an error.
- Reset the PIN status.
- Disable the currently Enabled Profile.
- Enable the Emergency Profile.
- Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).
- Implicitly select the MF on the basic logical channel.
- Return 'ok' to the IPA.

If the `refreshFlag` is set:

- If there is a proactive session ongoing;
  - The eUICC MAY terminate the “ES10b.EnableEmergencyProfile” function with error 'catBusy'.
  - If the eUICC does not terminate the “ES10b.EnableEmergencyProfile” function with error 'catBusy', the eUICC SHALL internally terminate the proactive session and send the REFRESH command as the next proactive command. If a TERMINAL RESPONSE is still outstanding, the REFRESH command SHALL only be sent after reception of the TERMINAL RESPONSE.

- Otherwise, the eUICC SHALL:
  - Mark the currently Enabled Profile as "to be disabled".
  - Mark the target Emergency Profile as "to be enabled".
  - Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).
  - Return 'ok' to the IPA.
  - Send the REFRESH command in "eUICC Profile State Change" mode (if supported by the IoT Device) or "UICC Reset" mode to the IoT Device according to ETSI TS 102 223 [5].
  - Upon reception of the Terminal Response with result "command performed successfully" or upon reset of the eUICC, disable the currently Enabled Profile and enable the Emergency Profile.
  - Upon reception of the Terminal Response with result "temporary problem with executing command" or "permanent problem with executing command", the eUICC SHALL NOT disable the currently Enabled Profile nor enable the Emergency Profile. Subsequent actions are implementation specific.

The eUICC MAY start a new proactive UICC session only after a new TERMINAL PROFILE command is executed.

When the Emergency Profile is enabled, the eUICC SHALL reject an incoming `EnableEmergencyProfileRequest` with the appropriate error code as defined in section 2.11.2.1.

### **Command Data**

The command data SHALL be coded as follows:

```
-- ASN1START
EnableEmergencyProfileRequest ::= [91] SEQUENCE { -- Tag 'BF5B'
  refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
-- ASN1STOP
```

### **Response Data**

The response data SHALL be coded as follows:

```
-- ASN1START
EnableEmergencyProfileResponse ::= [91] SEQUENCE { -- Tag 'BF5B'
  enableEmergencyProfileResult [0] INTEGER {
    ok(0),
    profileNotInDisabledState(2),
    catBusy(5),
    ecallNotAvailable(8), -- Emergency Profile does not exist
    undefinedError(127)
  }
}
-- ASN1STOP
```

## **5.9.23 Function (ES10b): DisableEmergencyProfile**

**Function Provider:** eUICC

### **Description:**

This function is OPTIONAL for eUICC and IPA, and MAY be used by the IPA to instruct the eUICC to disable the Emergency Profile. If "ES10b.EnableEmergencyProfile" function is supported, this function SHALL also be supported.

This function disables the currently enabled Emergency Profile and enables the Profile that was previously enabled prior to enabling the Emergency Profile. This function SHALL be performed in an atomic way, meaning that in case of any error during the function execution, the command SHALL stop and SHALL leave the involved Profiles in their original states (prior to function execution).

Upon reception of this function, the eUICC SHALL:

- check if the Emergency feature is supported. If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('eCallNotAvailable').
- verify that the Emergency Profile (Profile with the `eCallIndication` set to TRUE) is available. If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('eCallNotAvailable').
- verify that the selected Emergency Profile is enabled. If not, the procedure SHALL stop, and the result of this command SHALL indicate an error ('profileNotInEnabledState').

If the `refreshFlag` is not set, the eUICC SHALL:

- Check whether there is a proactive session ongoing (which the IoT Device did not terminate). If so:
  - the eUICC MAY terminate the "ES10b.DisableEmergencyProfile" function with error 'catBusy'.
  - If the eUICC does not terminate the "ES10b.DisableEmergencyProfile" function with error 'catBusy', the eUICC SHALL internally terminate the proactive session and ignore any incoming TERMINAL RESPONSE from that proactive session.
- Close all logical channels which still have an application of the currently Enabled Profile selected (which the IoT Device did not close), without generating an error.
- Reset the PIN status.
- Disable the Emergency Profile.
- Enable the previously Enabled Profile.
- Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).
- Implicitly select the MF on the basic logical channel.
- Return 'ok' to the IPA.

If the `refreshFlag` is set:

- If there is a proactive session ongoing;
  - The eUICC MAY terminate the "ES10b.DisableEmergencyProfile" function with error 'catBusy'.
  - If the eUICC does not terminate the "ES10b.DisableEmergencyProfile" function with error 'catBusy', the eUICC SHALL internally terminate the proactive session and send the REFRESH command as the next proactive command. If a TERMINAL RESPONSE is still outstanding, the REFRESH command SHALL only be sent after reception of the TERMINAL RESPONSE.
- Otherwise, the eUICC SHALL:
  - Mark the Emergency Profile as "to be disabled".

- Mark the previously Enabled Profile as "to be enabled".
- Reset any authorisation previously granted to use the Profile Rollback Mechanism (and any related data).
- Return 'ok' to the IPA.
- Send the REFRESH command in "eUICC Profile State Change" mode (if supported by the IoT Device) or "UICC Reset" mode to the IoT Device according to ETSI TS 102 223 [5].
- Upon reception of the Terminal Response with result "command performed successfully" or upon reset of the eUICC, disable the Emergency Profile and enable the previously Enabled Profile.
- Upon reception of the Terminal Response with result "temporary problem with executing command" or "permanent problem with executing command", the eUICC SHALL NOT disable the Emergency Profile nor enable the previous Enabled Profile. Subsequent actions are implementation specific.

The eUICC MAY start a new proactive UICC session only after a new TERMINAL PROFILE command is executed.

- If the Emergency Profile is successfully disabled, the eUICC MAY generate as many Notifications as configured in its Profile Metadata (notificationConfigurationInfo) in the format of OtherSignedNotification.
- If the previously enabled Profile is successfully enabled, the eUICC MAY generate as many Notifications as configured in its Profile Metadata (notificationConfigurationInfo) in the format of OtherSignedNotification.

### **Command Data**

```
-- ASN1START
DisableEmergencyProfileRequest ::= [92] SEQUENCE { -- Tag 'BF5C'
  refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
-- ASN1STOP
```

### **Response Data**

```
-- ASN1START
DisableEmergencyProfileResponse ::= [92] SEQUENCE { -- Tag 'BF5C'
  disableEmergencyProfileResult [0] INTEGER {
    ok(0),
    profileNotInEnabledState(2),
    catBusy(5),
    ecallNotAvailable(8), -- Emergency Profile does not exist or is not supported
    undefinedError(127)
  }
}
-- ASN1STOP
```

## **5.9.24 Function (ES10b): GetConnectivityParameters**

**Related Procedures: eIM Package Retrieval, Indirect Profile Download**

**Function Provider Entity: eUICC (ISD-R)**

**Function Caller Entity: IPA**

### **Description:**

This function is used by the IPA to read the HTTP/CoAP Connectivity Parameters of the currently enabled Profile, if available. The Connectivity Parameters for HTTP protocol SHOULD be provided as part of an Profile Package as described in the TCA Profile Interoperability Technical Specification [29]. They can be updated by the operator using ES6 interface as described in section 5.4 (ES6 (Operator -- eUICC)).

If available, the Connectivity Parameters for HTTP protocol SHALL be returned in the `httpParams` field described below. The format of this field is identical to that described in table 3 in section 2.4.4.

If not available, the `parametersNotAvailable` error SHALL be returned.

### **Command Data**

The command data SHALL be coded as follows:

```
-- ASN1START
GetConnectivityParametersRequest ::= [95] SEQUENCE { -- Tag 'BF5F'
}
-- ASN1STOP
```

### **Response Data**

The response data object SHALL be coded as follows:

```
-- ASN1START
GetConnectivityParametersResponse ::= [95] CHOICE { -- Tag 'BF5F'
  connectivityParameters ConnectivityParameters,
  connectivityParametersError ConnectivityParametersError
}

ConnectivityParameters ::= SEQUENCE {
  httpParams [1] OCTET STRING OPTIONAL -- Also used for CoAP
}

ConnectivityParametersError ::= INTEGER {
  parametersNotAvailable(1),
  undefinedError(127)
}
-- ASN1STOP
```

## **5.9.25 Function (ES10b): SetDefaultDpAddress**

**Function Provider Entity:** ISD-R (IPA Services)

### **Description:**

NOTE: ES10a.SetDefaultDpAddress function defined in SGP.22 [4] has a tag number 'BF3F'.

This function is used to update the default SM-DP+ address.

ES10b.SetDefaultDpAddress SHALL only be used if immediate enable is not configured. In case it is configured and the eUICC receives an ES10b.SetDefaultDpAddress, the eUICC SHALL reject the command with error code "undefinedError".

If the provided UTF8 string is not empty, it SHALL constitute the new default SM-DP+ address.

If the provided UTF8 string is empty, an existing default SM-DP+ address SHALL be removed.

### **Command Data**

The command data SHALL be coded as follows:

```
-- ASN1START
SetDefaultDpAddressRequest ::= [101] SEQUENCE { -- Tag 'BF65'
    defaultDpAddress UTF8String -- Default SM-DP+ address as an FQDN
}
-- ASN1STOP
```

### **Response Data**

The response data SHALL be coded as follows:

```
-- ASN1START
SetDefaultDpAddressResponse ::= [101] SEQUENCE { -- Tag 'BF65'
    setDefaultDpAddressResult INTEGER { ok (0), undefinedError (127) }
}
-- ASN1STOP
```

## **5.10 ES11 (IPA -- SM-DS)**

ES11 is the interface to retrieve Event Records (see section 3.9.2.1) between:

- The IPA entity.
- The SM-DS.

The IPA SHALL communicate with the SM-DS secured by TLS in server authentication mode as described in section 2.6.4 of this document.

This interface is identical to the ES11 interface defined in section 5.8 of SGP.22 [4], where the IPA plays the role of LPA, except Function (ES11): AuthenticateClient where the EUICInfo2 in AuthenticateResponseOk is according to Section 5.9.2 of this specification.

## **5.11 ES11' (eIM -- SM-DS)**

ES11' is the interface to retrieve Event Records (see section 3.9.2.2) between:

- The eIM entity.
- The SM-DS.

The eIM SHALL communicate with the SM-DS secured by TLS in server authentication mode as described in section 2.6.4 of this document.

### **5.11.1 Function (ES11'): InitiateAuthentication**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** SM-DS

**Function Caller Entity:** eIM

### **Description:**

This function SHALL be called by the eIM to authenticate the SM-DS.

This function is identical to the "ES11.InitiateAuthentication" function defined in section 5.8.1 of SGP.22 [4], where the eIM plays the role of LPA. In addition, in case the IPA has sent all the necessary input data in the "ESipa.InitiateAuthentication" function request, the eIM SHALL forward it as an "ES11'.InitiateAuthentication" function request. Otherwise, the eIM SHALL complete the missing input data as described in "ES11.InitiateAuthentication" function as defined in section 5.8.1 of SGP.22 [4].

### **5.11.2 Function (ES11'): AuthenticateClient**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** SM-DS

**Function Caller Entity:** eIM

### **Description:**

This function SHALL be called by the eIM to request the authentication of the eUICC by the SM-DS.

This function is identical to the "ES11.AuthenticateClient" function defined in section 5.8.2 of SGP.22 [4], where the eIM plays the role of LPA and the `EuiccInfo2` in `AuthenticateResponseOk` is according to Section 5.9.2 of this specification.

### **5.12 ES12 (SM-DS -- SM-DP+)**

The ES12 is used by the SM-DP+ to manage Event Registration (see section 3.9.1) and Event Deletion (see section 3.9.3). This interface is identical to the one defined in section 5.9 of SGP.22 [4].

### **5.13 ESep (eIM -- eUICC)**

ESep is the interface between:

- The eIM entity.
- The eUICC.

The ESep interface is based on the eIM sending a signed `EuiccPackageRequest` to the eUICC containing PSMO(s) or eCO(s) and receiving a signed `EuiccPackageResult` in return from the eUICC containing the execution results. The ESep functions are the PSMOs and eCOs in the eUICC Package.

#### **5.13.1 Function (ESep): Enable**

**Related Procedures:** Profile State Management

**Function Provider Entity:** eUICC

### **Description:**

This function enables an installed Profile in the eUICC. The input data is the `enable` structure defined in section 2.11.1.1.3 and the output is the `enableResult` structure defined in section 2.11.2.1. The function is further described in the Enable Profile procedure defined in section 3.4.1. In addition, the eUICC behaviour with `refreshFlag` set as defined in SGP.22 [4] section 5.7.16 applies.

### 5.13.2 Function (ESep): Disable

**Related Procedures:** Profile State Management

**Function Provider Entity:** eUICC

#### **Description:**

This function disables an enabled Profile in the eUICC. The input data is the `disable` structure defined in section 2.11.1.1.3 and the output is the `disableResult` structure defined in section 2.11.2.1. The function is further described in the Disable Profile procedure defined in section 3.4.2. In addition, the eUICC behaviour with `refreshFlag` set as defined in SGP.22 [4] section 5.7.17 applies.

### 5.13.3 Function (ESep): Delete

**Related Procedures:** Profile State Management

**Function Provider Entity:** eUICC

#### **Description:**

This function deletes an installed Profile in the eUICC. The input data is the `delete` structure defined in section 2.11.1.1.3 and the output is the `deleteResult` structure defined in section 2.11.2.1. The function is further described in the Delete Profile procedure defined in section 3.4.3. In addition, the eUICC behaviour as defined in SGP.22 [4] section 5.7.18 applies.

### 5.13.4 Function (ESep): ListProfileInfo

**Related Procedures:** Indirect Profile Download

**Function Provider Entity:** eUICC

#### **Description:**

This function allows the eIM to retrieve the list of Profile information for installed Profiles including their current state (Enabled/Disabled) and their associated Profile Metadata. The input data is the `listProfileInfo` structure defined in section 2.11.1.1.3 and the output is the `listProfileInfoResult` structure defined in section 2.11.2.1.

### 5.13.5 Function (ESep): GetRAT

**Related Procedures:** Indirect Profile Download

**Function Provider Entity:** eUICC

#### **Description:**

This function allows the eIM to retrieve the Rules Authorisation Table (RAT) from the eUICC. The input data is the `getRAT` structure defined in section 2.11.1.1.3 and the output is the `getRATResult` structure defined in section 2.11.2.1.

### 5.13.6 Function (ESep): ConfigureImmediateEnable

**Related Procedures:** Configure by eIM of Immediate Profile Enabling

**Function Provider Entity:** eUICC

#### **Description:**

This function configures the immediate enabling of a Profile in the eUICC. The input data is the `configureImmediateEnable` structure defined in section 2.11.1.1.3 and the output is the `configureImmediateEnableResult` structure defined in section 2.11.2.1. The function is further described in the Configuration by eIM of Immediate Profile Enabling procedure defined in section 3.4.4.

### 5.13.7 Function (ESep): AddEim

**Related Procedures:** Addition of eIM Configuration Data

**Function Provider Entity:** eUICC

#### **Description:**

This function adds an Associated eIM to the eUICC by providing its eIM Configuration Data including the `eimID` to the eUICC. The input data is the `addEim` structure defined in section 2.11.1.1.2 and the output is the `addEimResult` structure defined in section 2.11.2.1. The function is further described in the Addition of eIM Configuration Data procedure defined in section 3.5.1.1.

### 5.13.8 Function (ESep): UpdateEim

**Related Procedures:** Update of eIM Configuration Data

**Function Provider Entity:** eUICC

#### **Description:**

This function updates eIM Configuration Data, i.e., the public key or Certificate and the related anti-replay counter value of an Associated eIM with a given `eimID` within the eUICC while keeping the same `eimID`. The input data is the `updateEim` structure defined in section 2.11.1.1.2 and the output is the `updateEimResult` structure defined in section 2.11.2.1. The function is further described in the Update of eIM Configuration Data procedure defined in section 3.5.1.3.

### 5.13.9 Function (ESep): DeleteEim

**Related Procedures:** Deletion of eIM Configuration Data

**Function Provider Entity:** eUICC

### **Description:**

This function deletes an Associated eIM identified by its `eimID` from the eUICC. If the successfully deleted Associated eIM was the last available Associated eIM, the eUICC SHALL allow `ES10b.AddInitialEim` again. The input data is the `deleteEim` structure defined in section 2.11.1.1.2 and the output is the `deleteEimResult` structure defined in section 2.11.2.1. The function is further described in the Deletion of eIM Configuration Data procedure defined in section 3.5.1.2.

### **5.13.10 Function (ESep): ListEim**

**Related Procedures:** Request for a list of Associated eIMs

**Function Provider Entity:** eUICC

### **Description:**

This function requests the eUICC to provide a list of all currently configured Associated eIMs to the eIM. The input data is the `listEim` structure defined in section 2.11.1.1.2 and the output is the `listEimResult` structure defined in section 2.11.2.1. The function is further described in the Request for a list of Associated eIMs procedure defined in section 3.5.1.4.

### **5.13.11 Function (ESep): SetDefaultDpAddress**

**Function Provider Entity:** eUICC

### **Description:**

This function allows the eIM to update the Default SM-DP+ address configured within the eUICC. The input data is the `setDefaultDpAddress` structure defined in section 2.11.1.1.3 and the output is the `setDefaultDpAddressResult` structure defined in section 2.11.2.1.

## **5.14 ESipa (eIM -- IPA)**

This ESipa interface is used between:

- The IPA entity
- The eIM entity.

This section defines the different functions used through the ESipa interface.

### **5.14.1 Function (ESipa): InitiateAuthentication**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** eIM

**Function Caller Entity:** IPA

### **Description:**

This function requests the SM-DP+/SM-DS authentication via the eIM. This follows the `ES10b.GetEUICCChallenge` by the IPA, where the IPA retrieves the relevant information from

the eUICC and provides it to the eIM. The eIM SHALL send this information to the SM-DP+/SM-DS to initiate mutual authentication between the eUICC and the SM-DP+/SM-DS.

- On reception of this function call, the eIM SHALL perform the following: If an `eimTransactionId` is received: identify the session by using the `eimTransactionId`. If the received `eimTransactionId` is unknown, the eIM SHALL return a status code "8.31.2 eIM TransactionId – 3.10 Invalid Association".
- Identify any missing input data (e.g., `euiccInfo1` or `smdpAddress`) in the function call and restore them by using the data already available to the eIM for the corresponding eUICC.
- Call `ES9+'.InitiateAuthentication` or `ES11'.InitiateAuthentication` function.
- The input parameters of this function are identical to those of the `ES9+.InitiateAuthentication` / `ES11.InitiateAuthentication` defined in section 5.6.1/5.8.1 of SGP.22 [4], with the following exceptions: `euiccInfo1` and `smdpAddress` are OPTIONAL.
  - An IPA with IPA Capability `minimizeEsipaBytes` SHOULD NOT send `euiccInfo1` and `smdpAddress`.
  - In case an Activation Code is used, an IPA with IPA Capability `eimDownloadDataHandling` SHALL NOT send `smdpAddress`.
- If the eIM has sent `eimTransactionId` in `ProfileDownloadTriggerRequest`, the IPA SHALL include the same `eimTransactionId`.

**Additional Input Data**

Input data name	Description	Type	No.	MOC
<code>euiccChallenge</code>	<code>euiccChallenge</code> generated by the eUICC. IPA can retrieve the <code>euiccChallenge</code> from the eUICC by calling "ES10b.GetEUICCCChallenge" (section 5.9.3).	Binary[16]	1	M
<code>euiccInfo1</code>	<code>euiccInfo1</code> of the eUICC. IPA can retrieve the <code>euiccInfo1</code> by calling "ES10b.GetEUICCInfo" (section 5.9.2).	Binary <sup>(1)</sup>	1	O
<code>smdpAddress</code>	The SM-DP+/SM-DS Address as known and provided by the IPA.	FQDN	1	O
<code>eimTransactionId</code>	The transaction ID received from the eIM in <code>ProfileDownloadTriggerRequest</code>	Binary[1-16]	1	C
NOTE1: <code>euiccInfo1</code> SHALL be provided as an encoded <code>EuiccInfo1</code> data object.				

**Table 12 InitiateAuthentication Additional Input Data**

Upon receiving the `ES9+'.InitiateAuthentication` / `ES11'.InitiateAuthentication` response, the eIM SHALL perform the following:

- If the `ES9+'.InitiateAuthentication` or `ES11'.InitiateAuthentication` returns any error, the eIM SHALL return the same error to the IPA.
- If the `smdpAddress` provided as input to `ES9+'.InitiateAuthentication` / `ES11'.InitiateAuthentication` did not come from IPA (not present in `ESipa.InitiateAuthentication` e.g., due to IPA has capability

`eimDownloadDataHandling`), the eIM SHALL verify that the `serverAddress` in `serverSigned1` returned by the SM-DP+/SM-DS matches the `smdpAddress` that the eIM provided as input in `ES9+.InitiateAuthentication`. If not, the eIM SHALL return a status code "8.8.1 SMDP+ Address – 3.10 Invalid Association".

- If the Activation Code contains the SM-DP+ OID and is available to the eIM (e.g., due to IPA has capability `eimDownloadDataHandling`), the eIM SHALL check that the SM-DP+ OID from the Activation Code matches the SM-DP+ OID of the SM-DP+ Certificate (`serverCertificate`). If not, the eIM SHALL return a status code "8.8 SM-DP+ – 3.10 Invalid Association".
- If IPA does not generate `ctxParams1` (see IPA capability `eimCtxParams1Generation`), the eIM SHALL generate `ctxParams1` based on `matchingId` (if available) and `DeviceInfo`. `DeviceInfo` MAY be common for several IoT Devices and MAY have been retrieved from one IoT Device and re-used for other IoT Devices.

The eIM SHALL prepare the `ESipa.InitiateAuthentication` response. The output parameters of this function are identical to those of `ES9+.InitiateAuthentication` / `ES11'.InitiateAuthentication` defined in section 5.6.1/5.8.1 of SGP.22 [4] with the following modifications:

- `transactionId` is conditional since it is also part of `serverSigned1` and SHALL NOT be sent by the eIM to an IPA with IPA Capability `minimizeEsipaBytes` (i.e., IPA is capable of extracting `transactionId` from `serverSigned1`).
- `euiccCiPKIdentifierToBeUsed` SHALL be sent in truncated form (instead of the full CI Public Key Identifier) to an IPA with IPA Capability `minimizeEsipaBytes` to minimize the number of transmitted bytes.
- `ctxParams1` SHALL be provided to IPA if IPA is not capable of generating it (IPA capability `eimCtxParams1Generation`).
- `matchingId` SHALL be provided to the IPA if IPA generates `ctxParams1` and the `matchingId` is available to the eIM but not available to the IPA.

**Additional Output Data**

Output data name	Description	Type	No.	MOC
<code>transactionId</code>	Transaction ID as generated by the SM-DP+/SM-DS (section 3.1.1.4 of SGP.22 [4]).	Binary[1-16]	1	C
<code>serverSigned1</code>	The data object signed by the SM-DP+.	Binary <sup>(1)</sup>	1	M
<code>serverSignature1</code>	SM-DP+/SM-DS signature	Binary <sup>(1)</sup>	1	M
<code>euiccCiPKIdentifierToBeUsed</code>	CI Public Key Identifier to be used by the eUICC for signature.	Binary <sup>(1,2)</sup>	1	M
<code>serverCertificate</code>	SM-DP+/SM-DS Certificate (CERT.DPauth.ECDSA/ CERT.DSauth.ECDSA)	Binary <sup>(1)</sup>	1	M
<code>matchingId</code>	MatchingID/ EventID as defined in SGP.22 [4] identifying a specific	UTF8String	1	C <sup>(3)</sup>

	management order at the SM-DP+/SM-DS.			
ctxParams1	Data structure used in Common Mutual Authentication.	Binary <sup>(1)</sup>	1	C <sup>(3)</sup>

NOTE 1: serverSigned1, serverSignature1, euiccCiPKIdentifierToBeUsed, serverCertificate, and ctxParams1 are data objects defined in section 5.7.13 (function "ES10b.AuthenticateServer") of SGP.22 [4]. They SHALL be returned as encoded data objects including the tags defined for them in the AuthenticateServerRequest data object.

NOTE 2: euiccCiPKIdentifierToBeUsed contains either the full CI Public Key Identifier or a truncated version of the CI Public Key Identifier where only enough leading bytes of the CI Public Key Identifier are provided for IPA to be able to uniquely locate and extract the chosen CI Public Key Identifier from the euiccCiPKIdListForSigning of eUICCInfo1. If the euiccCiPKIdListForSigning only contains one CI Public Key Identifier, the euiccCiPKIdentifierToBeUsed encoded data object in truncated form indicates a zero length CI Public Key Identifier. The euiccCiPKIdentifierToBeUsed SHALL be sent in truncated form to an IPA with IPA Capability minimizeEsipaBytes. Otherwise, the full CI Public Key Identifier SHALL be used.

NOTE 3: matchingId and ctxParams1 are conditional since either the IPA or the eIM generates the ctxParams1. If the IPA generates ctxParams1 and the matchingId is available to the eIM but not available to the IPA (i.e. IPA sets in IpaCapabilities the value of eimDownloadDataHandling to 1 and the value of eimCtxParams1Generation to 0), the eIM SHALL provide the Matching ID to IPA.

**Table 13 InitiateAuthentication Additional Output Data**

The error codes returned by ESipa.InitiateAuthentication SHALL be the same as those of ES9+'.InitiateAuthentication / ES11'.InitiateAuthentication with the following additions:

**Specific Status Codes**

Subject Code	Subject	Reason code	Reason	Description
8.31.2	eIM TransactionId	3.10	Invalid Association	The provided eIM TransactionId is not valid. (maps to invalidEimTransactionId)
8.8.1	SM-DP+ Address	3.10	Invalid Association	SM-DP+ or SM-DS address returned by the RSP Server does not match the expected value. (maps to smdpAddressMismatch)
8.8	SM-DP+	3.10	Invalid Association	SM-DP+ OID in an Activation Code does not match the SM-DP+ OID in the SM-DP+ Certificate. (maps to smdpOidMismatch)

**Table 13a InitiateAuthentication Specific Status Codes**

**5.14.2 Function (ESipa): GetBoundProfilePackage**

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** eIM

**Function Caller Entity:** IPA

**Description:**

This function requests the delivery and the binding of a Profile Package for the eUICC.

This function is correlated to a previous normal execution of an ESipa.AuthenticateClient function through a TransactionID delivered by the SM-DP+.

On reception of this function call, the eIM SHALL perform the following:

- If the IPA indicates the IPA Capability minimizeEsipaBytes: restore the prepareDownloadResponse to contain downloadResponseOk by using smdpSigned2 and hashCc already available to the eIM for the corresponding eUICC.
- Call ES9+'.GetBoundProfilePackage' function.

The input parameters of this function are identical to the those of ES9+'.GetBoundProfilePackage' defined in section 5.6.2 of SGP.22 [4], with the following exceptions:

- prepareDownloadResponse is extended according to the below ASN.1 structure where a compactDownloadResponseOk is added.
  - An IPA with IPA Capability minimizeEsipaBytes SHOULD re-encode a downloadResponseOk contained in prepareDownloadResponse from eUICC into compactDownloadResponseOk.

**Additional Input Data**

Input data name	Description	Type	No.	MOC
transactionId	Transaction ID as generated by the SM-DP+ (section 3.1.1.4 of SGP.22 [4]).	Binary[1-16]	1	M
prepareDownloadResponse	PrepareDownloadResponse data object defined in section 5.7.5 of SGP.22 extended with compactDownloadResponseOk (see structure below). IPA retrieves prepareDownloadResponse from the eUICC by calling "ES10b.PrepareDownload" (section 5.9.7).	Binary <sup>(1)</sup>	1	M
NOTE 1: prepareDownloadResponse SHALL be provided as an encoded PrepareDownloadResponse data object.				

**Table 14 GetBoundProfilePackage Additional Input Data**

```
-- ASN1START
PrepareDownloadResponse ::= [33] CHOICE { -- Tag 'BF21'
    downloadResponseOk PrepareDownloadResponseOk,
    downloadResponseError PrepareDownloadResponseError,
    compactDownloadResponseOk CompactPrepareDownloadResponseOk
}
CompactPrepareDownloadResponseOk ::= SEQUENCE {
    compactEuiccSigned2 CompactEuiccSigned2, -- Compact version of EuiccSigned2
```

```

    euiccSignature2 [APPLICATION 55] OCTET STRING -- tag '5F37' signature on
    EuiccSigned2
}
CompactEuiccSigned2 ::= SEQUENCE {
    euiccOtpk [APPLICATION 73] OCTET STRING OPTIONAL, -- otPK.EUICC.ECKA, tag '5F49'
    euiccOtpk is always present except if bppEuiccOtpk was chosen by the eUICC
    hashCc Octet32 OPTIONAL -- Hash of confirmation code, if not received from eIM
}
-- ASN1STOP
    
```

Upon receiving the ES9+'.GetBoundProfilePackage response, the eIM SHALL perform the following:

- If the ES9+'.GetBoundProfilePackage returns any error, the eIM SHALL return the same error to the IPA.
- If the eIM verified the Profile Metadata in ESipa.AuthenticateClient (due to IPA Capability eimProfileMetadataVerification), the eIM SHOULD parse the boundProfilePackage received from SM-DP+, extract the Profile Metadata (encoded in the StoreMetadataRequest field), and check that the Profile Metadata has not changed. If it has changed, the eIM SHALL return a status code "8.2.9 Profile Metadata – 3.11 Value has Changed".

The eIM SHALL prepare the ESipa.GetBoundProfilePackage response. The output parameters of this function are identical to the those of ES9+'.GetBoundProfilePackage defined in section 5.6.2 of SGP.22 [4], with the following exceptions:

- The transactionId is conditional. The eIM SHALL NOT send the transactionId to an IPA with IPA Capability minimizeEsipaBytes. Otherwise, the eIM SHALL send the transactionId.

**Additional Output Data**

Output data name	Description	Type	No.	MOC
transactionId	Transaction ID as generated by the SM-DP+ (section 3.1.1.4 of SGP.22 [4]).	Binary[1-16]	1	C
boundProfilePackage	Bound Profile Package data object to be transferred to the eUICC using "ES10b.LoadBoundProfilePackage" (section 5.9.8).	Binary <sup>(1)</sup>	1	M

NOTE 1: boundProfilePackage SHALL be provided as an encoded BoundProfilePackage data object.

**Table 15 GetBoundProfilePackage Additional Output Data**

The error codes returned by ESipa.GetBoundProfilePackage SHALL be the same as those of ES9+'.GetBoundProfilePackage with the following additions:

**Specific Status Codes**

Subject Code	Subject	Reason code	Reason	Description
8.2.9	Profile Metadata	3.11	Value has Changed	The Profile Metadata in Bound Profile Package does not match the Profile Metadata received in ES9+'.AuthenticateClient response

				(maps to metadataMismatch)
--	--	--	--	----------------------------

**Table 15a GetBoundProfilePackage Specific Status Codes**

### 5.14.3 Function (ESipa): AuthenticateClient

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** eIM

**Function Caller Entity:** IPA

**Description:**

This function SHALL be called by the IPA to request the authentication of the eUICC by the SM-DP+/SM-DS.

This function is correlated to a previous normal execution of an ESipa.InitiateAuthentication through a TransactionID delivered by the SM-DP+/SM-DS.

On reception of this function call, the eIM SHALL perform the following:

- If the IPA indicates the IPA Capability minimizeEsipaBytes: restore the authenticateServerResponse to contain authenticateResponseOk by using serverSigned1, eUICCInfo2, ctxParams1, euiccCertificate, and eumCertificate already available to the eIM for the corresponding eUICC.
- Call ES9+'.AuthenticateClient or ES11'.AuthenticateClient function.

The input parameters of this function are identical to the those of the ES9+'.AuthenticateClient / ES11'.AuthenticateClient defined in section 5.6.3/5.8.2 of SGP.22 [4], with the following exceptions:

- authenticateServerResponse is extended according to the below ASN.1 structure where a compactAuthenticateResponseOk is added.
- An IPA with IPA Capability minimizeEsipaBytes SHOULD re-encode an authenticateResponseOk contained in authenticateServerResponse from eUICC into compactAuthenticateResponseOk.

**NOTE:** the use of this compactAuthenticateResponseOk reduces the number of transmitted bytes over ESipa by omitting either or both of eUICC Certificate and EUM Certificate, and/or sending only non-static parts of euiccInfo2.

**Additional Input Data**

Input data name	Description	Type	No.	MOC
transactionId	Transaction ID as generated by the SM-DP+/SM-DS (section 3.1.1.4 of SGP.22 [4]).	Binary[1-16]	1	M
authenticateServerResponse	AuthenticateServerResponse data object defined in section 5.7.13 of	Binary <sup>(1)</sup>	1	M

	SGP.22 with EUICCInfo2 according to Section 5.9.2 of this specification and extended with compactAuthenticateResponseOk (see structure below). IPA retrieves authenticateServerResponse from the eUICC by calling "ES10b.AuthenticateServer" (section 5.9.6).		
NOTE 1: authenticateServerResponse SHALL be provided as an encoded AuthenticateServerResponse data object.			

**Table 16 AuthenticateClient Additional Input Data**

```

-- ASN1START
EuiccSigned1 ::= SEQUENCE {
    transactionId [0] TransactionId,
    serverAddress [3] UTF8String,
    serverChallenge [4] Octet16, -- The RSP Server Challenge
    euiccInfo2 [34] EUICCInfo2,
    ctxParams1 CtxParams1
}
AuthenticateResponseOk ::= SEQUENCE {
    euiccSigned1 EuiccSigned1, -- Signed information
    euiccSignature1 [APPLICATION 55] OCTET STRING, --EUICC_Sign1, tag 5F37
    euiccCertificate Certificate, -- eUICC Certificate (CERT.EUICC.ECDSA) signed by
the EUM
    eumCertificate Certificate -- EUM Certificate (CERT.EUM.ECDSA) signed by the
requested CI
}
AuthenticateServerResponse ::= [56] CHOICE { -- Tag 'BF38'
    authenticateResponseOk AuthenticateResponseOk,
    authenticateResponseError AuthenticateResponseError,
    compactAuthenticateResponseOk CompactAuthenticateResponseOk
}
CompactAuthenticateResponseOk ::= SEQUENCE {
    signedData CHOICE {
        euiccSigned1 EuiccSigned1,
        compactEuiccSigned1 [0] CompactEuiccSigned1 -- Compact version of EuiccSigned1
    },
    euiccSignature1 [APPLICATION 55] OCTET STRING, -- tag 5F37 signature on
EuiccSigned1
    euiccCertificate [1] Certificate OPTIONAL, -- eUICC Certificate (CERT.EUICC.ECDSA)
    eumCertificate [2] Certificate OPTIONAL -- EUM Certificate (CERT.EUM.ECDSA)
}
CompactEuiccSigned1 ::= SEQUENCE {
    extCardResource [4] OCTET STRING, -- Extended Card Resource Information according
to ETSI TS 102 226 extracted from euiccInfo2,
    ctxParams1 [2] CtxParams1 OPTIONAL -- ctxParams1 may be left out by IPA if eIM
ctxParams1 was received from the eIM
}
-- ASN1STOP
    
```

Upon receiving the ES9+'.AuthenticateClient response, the eIM SHALL perform the following:

- If the ES9+'.AuthenticateClient returns any error, the eIM SHALL return the same error to the IPA.
- If the IPA is not capable of verifying the Profile Metadata (see IPA Capability eimProfileMetadataVerification), the eIM SHALL verify the Profile Metadata according to steps 7a, b, c and 8 of section 3.1.3 (Profile Download and Installation)

of SGP.22 [4]. For this verification, the eIM SHALL use Rules Authorisation Table and information on installed Profiles available to the eIM, e.g., obtained prior to the start of this procedure using the PSMOs `getRAT` and `listProfileInfo`. If the verification fails, the eIM SHALL return a status code "8.2.8 PPR – 1.2 Not Allowed".

Upon receiving the ES11'.AuthenticateClient response, the eIM SHALL perform the following:

- If the ES11'.AuthenticateClient returns any error, the eIM SHALL return the same error to the IPA.
- Extract Event entries from the ES11'.AuthenticateClient response. Optionally, prepare and provide a `ProfileDownloadTriggerRequest` as part of ES11'.AuthenticateClient response to trigger the IPA to initiate download of a new Profile where information obtained from the Event entries are leveraged.

The eIM SHALL prepare the ES11'.AuthenticateClient response. The output parameters of this function are identical to the those of ES9+.AuthenticateClient / ES11'.AuthenticateClient defined in section 5.6.3 / 5.8.2 of SGP.22 [4], with the following exceptions:

- If `smdpSign2` is provided, `transactionId` is optional. The eIM SHALL NOT send the `transactionId` to an IPA with IPA capability `minimizeEsipaBytes`.
- The eIM SHALL NOT send `profileMetadata` (from ES9+'.AuthenticateClient) to an IPA that indicates IPA Capability `eimProfileMetadataVerification`.
- `hashCc` (Hash of the Confirmation Code) from ES9+'.AuthenticateClient MAY be provided if a confirmation code is requested by the SM-DP+ and the confirmation code is available to the eIM.
- `eventEntries` from ES11'.AuthenticateClient are not forwarded to the IPA.
- a `ProfileDownloadTriggerRequest` data object MAY be provided in case the ES11'.AuthenticateClient originates from ES11'.AuthenticateClient.

### **Additional Output Data**

Output data name	Description	Type	No.	MOC
<code>transactionId</code>	Transaction ID as generated by the SM-DP+ (section 3.1.1.4 of SGP.22 [4]).	Binary[1-16]	1	C
<code>profileMetadata</code>	Profile Metadata to check against eUICC settings.	Binary <sup>(1)</sup>	1	C
<code>smdpSigned2</code>	The data to be signed by the SM-DP+.	Binary <sup>(1)</sup>	1	C
<code>smdpSignature2</code>	SM-DP+ signature	Binary <sup>(1)</sup>	1	C
<code>smdpCertificate</code>	SM-DP+ Certificate (CERT.DPpb.ECDSA)	Binary <sup>(1)</sup>	1	C
<code>hashCc</code>	Hash of the confirmation code if confirmation code is requested and available to eIM	Binary <sup>(1)</sup>	1	O
<code>profileDownloadTrigger</code>	Profile download trigger (see section 2.11.2.3 <code>ProfileDownloadTriggerRequest</code> )	Binary	1	O

NOTE 1: `profileMetadata` is the data object `StoreMetadataRequest` defined in section 5.5.3 (function "ES8+.StoreMetadata"); `smdpSigned2`, `smdpSignature2` and `smdpCertificate` are data objects defined in section 5.7.5 (function "ES10b.PrepareDownload") of SGP.22 [4]. They SHALL be returned

as encoded data objects including the tags defined for them in the StoreMetadataRequest/PrepareDownloadRequest data object.

**Table 17 AuthenticateClient Additional Output Data**

The error codes returned by ESipa.AuthenticateClient SHALL be the same as those of ES9+'.AuthenticateClient with the following additions:

**Specific Status Codes**

Subject Code	Subject	Reason code	Reason	Description
8.2.8	PPR	1.2	Not Allowed	The corresponding ES9+'.AuthenticateClient function returned PPR not allowed in RAT. (maps to pprNotAllowed)

**Table 17a AuthenticateClient Specific Status Codes**

**5.14.4 Function (ESipa): TransferEimPackage**

**Related Procedures:** eIM Package injection

**Function Provider Entity:** IPA

**Function Caller Entity:** eIM

**Description:**

This function is used by the eIM to transfer single eIM Package to the IPA.

On reception of the function call:

- In case the eIM Package contains an eUICC Package (EuiccPackageRequest), the IPA SHALL:
  - Call the "ES10b.LoadEuiccPackage" function of the ISD-R with its relevant input data.
  - Collect the eUICC Package execution result from the eUICC to notify the eIM about the result of the operation.
  - In case the IPA detects that the eIM cannot be notified about the result of the eIM Package execution then the IPA SHOULD call the "ES10b.ProfileRollback" function (see section 5.9.16) of the ISD-R as defined in section 3.3.1. The response to the eIM Package execution SHALL be then collected again after the Rollback operation.

**NOTE:** In case the IPA detects that the eIM cannot be notified about the result, the IPA can retry notifying the eIM at a later stage prior to calling the function "ES10b.ProfileRollback".

- If the IPA sends Notifications together with the eUICC Package Result in the eIM Package Result and if the eUICC Package contains PSMO(s): retrieve pending Notifications by calling ES10b.RetrieveNotificationsList and, if a non-empty list of pending Notifications, include the list of pending Notifications

(PendingNotificationList as defined in section 2.11.2.2) after the signed eUICC Package Result in the eIM Package. In case of an IPA with IPA Capability minimizeEsipaBytes, the IPA SHOULD include each Notification in the list in compact format as described in section 5.14.7.

- In case the eIM Package contains a request for IPA and/or eUICC data (IpaEuiccDataRequest), the IPA SHALL collect the requested data and return the IpaEuiccDataResponse structure.
- In case the eIM Package contains a Profile download trigger request (ProfileDownloadTriggerRequest), the IPA SHALL:
  - Parse ProfileDownloadTriggerRequest to obtain any necessary data needed for the Profile download (e.g., an Activation Code). If the contained data is not valid to proceed with the Profile download, the IPA SHALL return a status code "8.31.1 eIM Package – 2.1 Invalid".
    - If the ProfileDownloadTriggerRequest indicates direct Profile download, the IPA SHALL return the eimPackageReceivedWithCid structure, which SHOULD include correlationId representing either eimTransactionId (if received in the eIM Package) or EID (otherwise).
    - If the ProfileDownloadTriggerRequest indicates indirect Profile download, the IPA MAY return the eimPackageReceivedWithCid structure, which SHOULD include correlationId representing either eimTransactionId (if received in the eIM Package) or EID (otherwise).

NOTE: After sending ProfileDownloadTriggerRequest indicating indirect Profile download to the IPA, the eIM expects either eimPackageReceived structure followed by the ESipa.InitiateAuthentication function call or directly the ESipa.InitiateAuthentication function call. In the latter case successful reception of the ProfileDownloadTriggerRequest is implicit.

- Proceed with the suitable Profile download and installation procedure as described in section 3.2.
- In case the eIM Package contains the EimAcknowledgements structure with one or more sequence numbers of Notifications and/or eUICC Package Results, the IPA SHALL:
  - Return the eimPackageReceivedWithCid structure, which SHOULD include correlationId representing EID.
  - For each sequence number, call the ES10b.RemoveNotificationFromList function where the eUICC deletes the Notification/eUICC Package Result identified by SequenceNumber from its memory.
- In case the eIM Package contains a data object other than those defined in table 18, the IPA SHALL return a status code "8.31.1 eIM Package – 1.1 Unknown".

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the eIM Package has been successfully executed and an eIM Package Result is generated.

- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table. In ASN.1 binding, the IPA SHALL include the status code in `eimPackageErrorWithCid` data object, which SHOULD include `correlationId` representing either `eimTransactionId` (if received in the eIM Package) or EID (otherwise).

**Additional Input Data**

Input data name	Description	Type	No.	MOC
EuiccPackageRequest	The eUICC Package Request	Binary	1	C <sup>(1)</sup>
IpaEuiccDataRequest	The request for IPA and/or eUICC data	Binary	1	C <sup>(1)</sup>
ProfileDownloadTriggerRequest	The request for IPA to trigger profile download	Binary	1	C <sup>(1)</sup>
EimAcknowledgements	Sequence number(s) of Notifications and/or eUICC Package Results to be acknowledged	Binary	1	C <sup>(1)</sup>
NOTE 1: The eIM Package SHALL either contain an EuiccPackageRequest data object, an IpaEuiccDataRequest data object, ProfileDownloadTriggerRequest data object, or an EimAcknowledgements data object.				

**Table 18 TransferEimPackage Additional Input Data**

**Additional Output Data**

Output data name	Description	Type	No.	MOC
EuiccPackageResult	The result of eUICC Package processing	Binary	1	C <sup>(1)</sup>
IpaEuiccDataResponse	The response containing requested IPA and/or eUICC data	Binary	1	C <sup>(1)</sup>
PendingNotificationList	List of pending Notifications	Binary	1	C <sup>(1,2)</sup>
EimPackageReceived	Acknowledgement that an eIM Package was received successfully	Binary	1	C <sup>(1,3)</sup>
EimPackageReceivedWithCid	Acknowledgement with a correlation ID indicating that an eIM Package was received successfully	Binary	1	C <sup>(1,3)</sup>

EimPackageError	The response indicating that an eIM Package was received in error	Binary	1	C <sup>(1,4)</sup>
EimPackageErrorWithCid	The response with a correlation ID indicating that an eIM Package was received in error	Binary	1	C <sup>(1,4)</sup>
<p>NOTE 1: The eIM Package Result MAY contain either:</p> <ul style="list-style-type: none"> <li>• an EuiccPackageResult data object</li> <li>• an EuiccPackageResult data object concatenated with a PendingNotificationList data object</li> <li>• an IpaEuiccDataResponse data object</li> <li>• a simple acknowledgement that the eIM Package was successfully received (EimPackageReceived or EimPackageReceivedWithCid – see NOTE 3)</li> <li>• a response that the eIM Package was received in error (EimPackageError or EimPackageErrorWithCid – see NOTE 4).</li> </ul> <p>NOTE 2: The PendingNotificationList data object, if included, SHALL be sent together with an EuiccPackageResult in the same eIM Package Result.</p> <p>NOTE 3: EimPackageReceived is used by IPAs compliant to earlier versions of the specification, and is superseded by EimPackageReceivedWithCid.</p> <p>NOTE 4: EimPackageError is used by IPAs compliant to earlier versions of the specification, and is superseded by EimPackageErrorWithCid.</p>				

**Table 19 TransferEimPackage Additional Output Data**

**Specific Status Codes**

Subject Code	Subject	Reason code	Reason	Description
8.31.1	eIM Package	2.1	Invalid	The eIM Package contains an invalid Profile download trigger. (maps to <code>invalidPackageFormat</code> )
8.31.1	eIM Package	1.1	Unknown	The eIM Package is not supported by the IPA. (maps to <code>unknownPackage</code> )

**Table 19a TransferEimPackage Specific Status Codes**

**5.14.5 Function (ESipa): GetEimPackage**

**Related Procedures:** eIM Package retrieval

**Function Provider Entity:** eIM

**Function Caller Entity:** IPA

**Description:**

This function is used by the IPA to retrieve an eIM Package.

The IPA provides the EID as an input to the function call.

The IPA MAY notify the eIM that one or several changes occurred in the eUICC (e.g. list of Profiles, Profiles states, ...). In that case, the eIM SHOULD update its information about the eUICC.

The IPA MAY include the identity of the last PLMN the IoT Device registered to, in order for the eIM to detect a roaming situation.

On reception of the function call the eIM SHALL:

- Identify the eUICC by using the received EID.
  - If EID format is not valid, the eIM SHALL return a status code "8.1.1 EID – 2.1 Invalid".
  - If EID is not provided, the eIM SHALL return a status code "8.1.1 EID – 2.2 Mandatory Element Missing".
  - If EID is unknown to the eIM, the eIM SHALL return a status code "8.1.1 EID – 3.9 Unknown".
- Check if there is any eIM Package available for the eUICC identified by the EID.
  - If so, the eIM SHALL return a single eIM Package. If multiple eIM Packages are pending for the eUICC, it is eIM implementation-specific to select one of the pending eIM Packages.
  - Otherwise, the eIM SHALL return a status code "8.1.1 EID – 3.7 Unavailable".

NOTE: In case the IPA detects that the eIM cannot be notified about the result, the IPA can retry notifying the eIM at a later stage prior to calling the function "ES10b.ProfileRollback".

The eIM MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that an eIM Package has been successfully identified.
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

**Additional Input Data**

Input data name	Description	Type	No.	MOC
Eid	The EID of the eUICC for which the eIM Package SHALL be retrieved	EID	1	M
NotifyStateChange	Notification to the eIM that it SHOULD update its information about the eUICC (e.g. list of Profiles, Profile states...)	(empty)	1	O
stateChangeCause	Provide the cause of the notified state change(s)	Integer	1	C <sup>(1)</sup>
RPLMN	Identifier of the last Registered PLMN (RPLMN), as defined in 3GPP TS 23.122 [19] and coded	Binary	1	O

	as defined in 3GPP TS 24.008 [22]			
--	-----------------------------------	--	--	--

NOTE 1: mandatory if NotifyStateChange is present.

**Table 20 GetEimPackage Additional Input Data**

**Additional Output Data**

Output data name	Description	Type	No.	MOC
EuiccPackageRequest	The eUICC Package Request	Binary	1	C <sup>(1)</sup>
IpaEuiccDataRequest	The request for IPA and/or eUICC data	Binary	1	C <sup>(1)</sup>
ProfileDownloadTriggerRequest	The request for IPA to trigger profile download	Binary	1	C <sup>(1)</sup>

NOTE 1: Either one and only one of EuiccPackageRequest, IpaEuiccDataRequest or ProfileDownloadTriggerRequest can be present.

**Table 21 GetEimPackage Additional Output Data**

**Specific Status Codes**

Subject Code	Subject	Reason code	Reason	Description
8.1.1	EID	3.7	Unavailable	No pending eIM Package for the EID. (maps to noEimPackageAvailable)
8.1.1	EID	2.1	Invalid	EID parameter is invalid. See EID format defined in [6] section 2.2.2 (maps to invalidEid)
8.1.1	EID	2.2	Mandatory Element Missing	Mandatory EID parameter is missing. (maps to missingEid)
8.1.1	EID	3.9	Unknown	No eUICC found with specified EID. (maps to eidNotFound)

**Table 22 GetEimPackage Specific Status Codes**

**5.14.6 Function (ESipa): ProvideEimPackageResult**

**Related Procedures:** eIM Package Retrieval

**Function Provider Entity:** eIM

**Function Caller Entity:** IPA

**Description:**

This function is used by the IPA to deliver an eIM Package Result optionally including one or more Notifications to the eIM in the same function call.

On reception of the function call the eIM SHALL:

- Process the eIM Package Result. For that, the eIM SHALL:
  - Identify the eUICC by using the received EID or data already available to the eIM.
    - If EID is received but the format is not valid, the eIM SHALL return a status code "8.1.1 EID – 2.1 Invalid".
    - If EID is received but it is unknown to the eIM, the eIM SHALL return a status code "8.1.1 EID – 3.9 Unknown".
    - If EID is not received and the eIM cannot identify the eUICC, the eIM SHALL return a status code "8.1.1 EID – 2.2 Mandatory Element Missing".
  - If the eIM Package Result contains `EuiccPackageResult` (possibly within `ePRAndNotifications` data object):
    - Verify the eUICC signature using `PK.EUICC.ECDSA` of the eUICC. If the signature is invalid, the eIM SHALL discard the input.
    - Check if the sequence number of the `EuiccPackageResult` is greater than the expected sequence number of the eUICC. If not, the eIM SHALL discard the input.
    - Manage the eIM Packages pending for the eUICC.
    - Update the expected sequence number of the eUICC to the received sequence number.
  - If the eIM Package Result contains `IpaEuiccDataResponse`:
    - Store the received IPA and/or eUICC information for future use.
  - If the eIM Package Result contains a `notificationList` data object within `ePRAndNotifications`:
    - If the `notificationList` contains compact format of Notification(s), restore the Notification(s) in `notificationList` by using the data already available to the eIM.
    - Forward the (restored) Notifications to the corresponding Notification Receiver(s).
  - If the eIM Package Result contains `profileDownloadTriggerResult` or `eimPackageResultResponseError`:
    - Manage the eIM Packages pending for the eUICC.
- Return the sequence numbers of the eUICC Package Result and optionally Notifications that were processed (including discarded results) and/or forwarded above.

The eIM MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the eIM Package Result has been successfully delivered.
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

**Additional Input Data**

Input data name	Description	Type	No.	MOC
EidValue	The EID of the involved eUICC	EID	1	C <sup>(1)</sup>

EimPackageResult	The result of an eIM Package processing, as defined in section 6.3.2.7.	Binary	1	M <sup>(2)</sup>
NOTE 1: EID is required if there is no means for eIM to identify the eUICC (e.g., via ESipa underlying transport (see Figure 8) or eimTransactionId). Otherwise it is optional. NOTE 2: CompactOtherSignedNotification within EimPackageResult does not require EID; see section 5.14.7.				

**Table 23 ProvideEimPackageResult Additional Input Data**

**Additional Output Data**

Output data name	Description	Type	No.	MOC
EimAcknowledgements	Sequence number(s) of eUICC Package Result and/or Notification(s) acknowledged by the eIM.	Binary	1	0

**Table 24 ProvideEimPackageResult Additional Output Data**

**Specific Status Codes**

Subject Code	Subject	Reason code	Reason	Description
8.1.1	EID	2.1	Invalid	EID parameter is invalid. See EID format defined in [6] section 2.2.2 (maps to <code>invalidEid</code> )
8.1.1	EID	2.2	Mandatory Element Missing	Mandatory EID parameter is missing. (maps to <code>missingEid</code> )
8.1.1	EID	3.9	Unknown	No eUICC found with specified EID. (maps to <code>eidNotFound</code> )

**Table 25 ProvideEimPackageResult Specific Status Codes**

**5.14.7 Function (ESipa): HandleNotification**

**Related Procedures:** Profile Download and Installation, eIM Package Retrieval

**Function Provider Entity:** eIM

**Function Caller Entity:** IPA

**Description:**

This function SHALL be called by the IPA to notify the eIM that a Profile has been successfully installed on the eUICC or that a Profile has been successfully enabled, disabled, or deleted on the eUICC. This function is also used in case a Fallback Profile was enabled, disabled or deleted on the eUICC. This function MAY also be used in case of disabling an Emergency Profile. This function MAY also be used to notify the eIM that a Profile has been successfully installed on the eUICC by sending either a ProfileInstallationResult Notification, compactProfileInstallationResult Notification or a ProfileDownloadTriggerResult.

In the context of Indirect Profile Download and if the Notification is a ProfileInstallationResult or compactProfileInstallationResult, this function is correlated to a previous execution of an ESipa.GetBoundProfilePackage through a TransactionID delivered by the SM-DP+. If the Notification is an OtherSignedNotification or compactOtherSignedNotification, then the Notification Receiver address is the NotificationAddress contained in the NotificationMetaData of the OtherSignedNotification or compactOtherSignedNotification.

On reception of this function call, the eIM SHALL:

- If the function call contains a pendingNotification data object:
  - If the pendingNotification contains compact format of Notification, restore the Notification by using the data already available to the eIM.
  - Forward the (restored) Notification to the corresponding Notification Receiver.
- If the function call contains a provideEimPackageResult data object: process the eIM Package Result. For that, the eIM SHALL:
  - If the eIM Package Result contains EuiccPackageResult:
    - Verify the eUICC signature using PK.EUICC.ECDSA of the eUICC. If the signature is invalid, the eIM SHALL discard the input.
    - Check if the sequence number of the EuiccPackageResult is greater than the expected sequence number of the eUICC. If not, the eIM SHALL discard the input.
    - Manage the eIM Packages pending for the eUICC.
    - Update the expected sequence number of the eUICC to the received sequence number.
  - If the eIM Package Result contains IpaEuiccDataResponse:
    - Store the received IPA and/or eUICC information for future use.
  - If the eIM Package Result contains profileDownloadTriggerResult or eimPackageError:
    - Manage the eIM Packages pending for the eUICC.

The eIM MAY perform additional operations, which are out of scope of this specification.

**Additional Input Data**

Input data name	Description	Type	No.	MOC
pendingNotification	PendingNotification data object as defined in section 5.7.10 of SGP.22 extended with compactProfileInstallationResult and compactOtherSignedNotification (see structures below).	Binary <sup>(1)</sup>	1	O
provideEimPackageResult	ProvideEimPackageResult as defined in section 6.3.2.7.	Binary <sup>(2,3)</sup>	1	C
NOTE 1: pendingNotification SHALL be provided as an encoded PendingNotification data object. NOTE 2: In the context of this function call, the provideEimPackageResult data object SHALL NOT contain ePRAndNotifications. NOTE 3: Either only one of pendingNotification or provideEimPackageResult SHALL be present.				

**Table 26 HandleNotification Additional Input Data**

### **Compact format of Notifications**

This specification extends the `pendingNotification` data object defined in section 5.6.4 of SGP.22 [4] with compact formats according to the below ASN.1 structure. The compact format reduces the number of bytes transmitted over ESipa.

An IPA with IPA Capability `minimizeEsipaBytes` SHOULD re-encode `ProfileInstallationResult` and `OtherSignedNotification` into `CompactProfileInstallationResult` and `CompactOtherSignedNotification`, respectively, where the eUICC signature is kept unchanged.

Before calling the `ES9+'.HandleNotification` function, an eIM receiving compact format of Notifications SHALL restore the `pendingNotification` by using information already available at the eIM (e.g., `StoreMetadataRequest` (e.g., from `BoundProfilePackage`), `serverSigned1`, `serverCertificate/smdpCertificate`, `euiccCertificate`, and `eumCertificate`, etc.).

```
-- ASN1START
PendingNotification ::= CHOICE {
  profileInstallationResult [55] ProfileInstallationResult, -- tag 'BF37'
  otherSignedNotification OtherSignedNotification,
  compactProfileInstallationResult [0] CompactProfileInstallationResult,
  compactOtherSignedNotification [1] CompactOtherSignedNotification
}
ProfileInstallationResult ::= [55] SEQUENCE { -- Tag 'BF37'
  profileInstallationResultData [39] ProfileInstallationResultData,
  euiccSignPIR EuiccSignPIR
}
CompactProfileInstallationResult ::= SEQUENCE {
  compactProfileInstallationResultData [0] CompactProfileInstallationResultData,
  euiccSignPIR EuiccSignPIR
}
CompactProfileInstallationResultData ::= SEQUENCE {
  transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
  seqNumber INTEGER,
  iccidPresent BOOLEAN DEFAULT TRUE,
  compactFinalResult [2] CHOICE {
    compactSuccessResult CompactSuccessResult,
    errorResult ErrorResult
  }
}
CompactSuccessResult ::= SEQUENCE {
  compactAid [APPLICATION 15] OCTET STRING (SIZE (2)), -- Byte 14 and 15 of ISD-P
  AID
  simaResponse OCTET STRING OPTIONAL -- MUST be present if the simaResponse value
  (EUICCResponse) is different from the 9-byte value '30 07 A0 05 30 03 80 01 00'
  representing success
}
CompactOtherSignedNotification ::= SEQUENCE {
  eidValue [APPLICATION 26] Octet16 OPTIONAL, -- Tag '5A'
  tbsOtherNotification NotificationMetadata,
  euiccNotificationSignature [APPLICATION 55] OCTET STRING, -- eUICC signature of
  tbsOtherNotification, Tag '5F37'
  euiccCiPKIdentifierToBeUsed OCTET STRING OPTIONAL -- CI Public Key Identifier
  (possibly truncated) that was used by the eUICC to sign OtherSignedNotification
}
-- ASN1STOP
```

NOTE: Truncation of eUICC CI Public Key Identifier (`euiccCiPKIdentifierToBeUsed`) is specified in section 5.14.1.

NOTE: If an underlying protocol is used for the transport of ESipa messages as shown in Figure 8, the underlying protocol typically provides information to identify the EID to the eIM. Then the IPA does not have to include the `eidValue` as part of the `CompactOtherSignedNotification` structure. If there is no underlying protocol like in Figure 7, it is recommended that the IPA includes the `eidValue` in the `CompactOtherSignedNotification` structure in the case a `pendingNotification` is sent using `ESipa.HandleNotification` function.

### 5.14.8 Function (ESipa): CancelSession

**Related Procedures:** Profile Download and Installation

**Function Provider Entity:** eIM

**Function Caller Entity:** IPA

**Description:**

This function SHALL be called by the IPA to request the cancellation of an on-going RSP session. This decision MAY originate from the eIM, or IPA based on where the failure occurred. The eIM MAY return error codes in `ESipa.AuthenticateClient` and `ESipa.GetBoundProfilePackage` that triggers IPA to cancel the session.

This function is correlated to a previous execution of an `ESipa.AuthenticateClient` or `ESipa.GetBoundProfilePackage` through a `TransactionID` delivered by the SM-DP+.

On reception of this function call, the eIM SHALL:

- If the IPA indicates the IPA Capability `minimizeEsipaBytes`: restore the `cancelSessionResponse` to contain `cancelSessionResponseOk` by using `transactionId` (e.g., from `serverSigned1`), and `serverCertificate` or `smdpCertificate` already available to the eIM for the corresponding eUICC.
- Call `ES9+'.CancelSession` function.

The input parameters of this function are identical to the those of `ES9+'.CancelSession` defined in section 5.6.5 of SGP.22 [4] with the following exceptions:

- `cancelSessionResponse` is extended according to the below ASN.1 structure where a `compactCancelSessionResponseOk` is added.
- An IPA with IPA Capability `minimizeEsipaBytes` SHOULD re-encode a `cancelSessionResponseOk` contained in `cancelSessionResponse` from eUICC into `compactCancelSessionResponseOk`.

**Additional Input Data**

Input data name	Description	Type	No.	MOC
<code>transactionId</code>	Transaction ID as generated by the SM-DP+ (section 3.1.1.4 of SGP.22 [4]).	Binary[1-16]	1	M
<code>cancelSessionResponse</code>	CancelSessionResponse data object defined in section 5.7.14 of SGP.22 extended with	Binary <sup>(1)</sup>	1	M

	compactCancelSessionResponseOk (see structure below). IPA retrieves authenticateServerResponse from the eUICC by calling "ES10b.CancelSession" (section 5.9.9).			
NOTE 1: cancelSessionResponse SHALL be provided as an encoded CancelSessionResponse data object.				

**Table 27 CancelSession Additional Input Data**

```

-- ASN1START
CancelSessionResponse ::= [65] CHOICE { -- Tag 'BF41'
    cancelSessionResponseOk CancelSessionResponseOk,
    cancelSessionResponseError INTEGER {invalidTransactionId(5), undefinedError(127)},
    compactCancelSessionResponseOk CompactCancelSessionResponseOk
}
CompactCancelSessionResponseOk ::= SEQUENCE {
    compactEuiccCancelSessionSigned CompactEuiccCancelSessionSigned, -- Compact
version of euiccCancelSessionSigned
    euiccCancelSessionSignature [APPLICATION 55] OCTET STRING -- tag 5F37 signature on
euiccCancelSessionSigned
}
CompactEuiccCancelSessionSigned ::= SEQUENCE {
    reason CancelSessionReason OPTIONAL
}
-- ASN1STOP
    
```

The eIM SHALL forward the status/error code of ES9+'.CancelSession as the status/error code of ESipa.CancelSession.

## 6 Interface Binding

This section defines how to use HTTP with TLS as the transport layer to exchange ES2+, ES9+, ES9+', ES11, ES11', ES12, and ESipa function requests and responses.

This section also defines how to use CoAP with DTLS as the transport layer to exchange ESipa function requests and responses.

For the ES2+, ES9+, ES9+', ES11 and ES11', the interface binding as described in section 6 of SGP.22 [4] SHALL be used; where for ES9+ and ES11, IPA plays the role of the LPA and for ES9+' and ES11' eIM plays the role of the LPA.

For ESipa, the interface binding when using HTTP with TLS and CoAP with DTLS are described in section 6.1 and section 6.2, respectively. This interface binding leverages either the ASN.1 function binding described in section 6.3 of this specification, or the JSON function binding described in section 6.4 of this specification. The ASN.1 function binding MAY also be leveraged by other protocols such as LwM2M (see Annex B) used in the communication between eIM and IPA.

For better IPA interoperability, the eIM SHALL support both the ASN.1 binding and the JSON binding on ESipa.

For better IPA interoperability:

- the SM-DP+ SHOULD support both the JSON binding and the ASN.1 binding on ES9+.

- the SM-DS SHOULD support both the JSON binding and the ASN.1 binding on ES11.

## 6.1 ESipa interface binding over HTTP

The ESipa interface binding over HTTP SHALL follow the interface binding over HTTP in SGP.22 [4] section 6 relevant for the ES9+ interface with the following exceptions:

- TLS requirements SHALL follow section 2.6.3 in this document.
- User-Agent header (<User Agent>) SHALL be set to either gsma-rsp-ipad or gsma-rsp-ipae. The "User-Agent" field MAY contain additional information after a semicolon.
- The ASN.1 message definition SHALL follow section 6.1.1 below replacing section 6.6.1 of SGP.22 [4] and the ASN.1 list of functions SHALL follow section 6.3 replacing section 6.6.2 of SGP.22 [4].
- A JSON binding SHALL be indicated by the value "application/json; charset=UTF-8", which also mandates UTF-8 encoding. The JSON message definition SHALL follow section 6.1.2 below.
- "X-Admin-Protocol" header field SHALL be set to highest version of SGP.32 [This document] supported by the sender. When the sender is the IoT Device, this indicates the highest version supported by the IPA.

### 6.1.1 ASN.1 message definition

The Function requester and the Function provider SHALL exchange the DER encoded ASN.1 objects in HTTP messages as follows.

HTTP Request SHALL have the following format:

```
POST /gsma/rsp2/asn1 HTTP/1.1
Host: <Server Address>
User-Agent: <User Agent>
X-Admin-Protocol: gsma/rsp/v<x.y.z>
Content-Type: application/x-gsma-rsp-asn1
Content-Length: <Length of the ASN.1 EsipaMessageFromIpaToEim>

<ASN.1 EsipaMessageFromIpaToEim>
```

Any function execution request using ASN.1 binding SHALL be sent to the generic HTTP path '/gsma/rsp2/asn1'.

The body part of the HTTP POST request SHALL contain one EsipaMessageFromIpaToEim data object as defined in section 6.3. If the HTTP POST request is a HandleNotificationEsipa, the HTTP POST response body for a normal notification function execution SHALL be empty and the status code SHALL be '204' (No Content).

HTTP Response SHALL have the following format:

```
HTTP/1.1 <HTTP Status Code>
X-Admin-Protocol: gsma/rsp/v<x.y.z>
Content-Type: application/x-gsma-rsp-asn1
Content-Length: <Length of the ASN.1 EsipaMessageFromEimToIpa>

<ASN.1 EsipaMessageFromEimToIpa>
```

The body part of the HTTP POST response SHALL contain one `EsipaMessageFromEimToIpa` data object as defined in section 6.3.

## 6.1.2 JSON message definition

The Function requester and the Function provider SHALL exchange the JSON objects in HTTP messages as follows.

HTTP Request SHALL have the following format:

```
POST <HTTP Path> HTTP/1.1
Host: <Server Address>
User-Agent: <User Agent>
X-Admin-Protocol: gsma/rsp/v<x.y.z>
Content-Type: application/json;charset=UTF-8
Content-Length: <Length of the JSON requestMessage>

<JSON requestMessage>
```

<JSON requestMessage> is the combination of

- <JSON requestHeader>
- <JSON requestBody> which depends on the function called

<JSON requestHeader> is defined in section 6.5.1.3 of SGP.22 [4]. HTTP messages for ESipa SHALL NOT contain the <JSON requestHeader>. The <HTTP Path> is used to indicate which function execution is requested by the HTTP client. The list of defined <HTTP Path> and <JSON requestBody> are described in section 6.4.1.

HTTP Response SHALL have the following format:

```
HTTP/1.1 <HTTP Status Code>
X-Admin-Protocol: gsma/rsp/v<x.y.z>
Content-Type: application/json;charset=UTF-8
Content-Length: <Length of the JSON responseMessage>

<JSON responseMessage>
```

<JSON responseMessage> is the combination of

- <JSON responseHeader>
- <JSON responseBody> which depends on the function called

<JSON responseHeader> is defined in section 6.5.1.4 of SGP.22 [4]. The HTTP POST response body SHALL be empty for Notification message (see section 6.5.1.2 of SGP.22 [4]). The list of defined <HTTP Path> and <JSON responseBody> are described in section 6.4.1.

## 6.2 ESipa interface binding over CoAP

The ESipa interface binding over CoAP SHALL follow the ESipa interface binding over HTTP described in section 6.1. The ASN.1 function binding SHALL be used. In the ASN.1 message binding in section 6.1.1, the HTTP headers SHALL be translated to CoAP options as described by RFC 7252 [7].

- The <HTTP Path> SHALL be converted to a sequence of corresponding Uri-Path options. URI-Query is not used.
- The <Server Address> SHALL be converted to a Uri-Host option if necessary. If the server address is simply the IP address of the eIM server, this option SHALL be omitted.
- URI-Port option SHALL be added in case nonstandard port is used.
- The X-Admin-Protocol content SHALL be sent using the following CoAP option

Option	Number	Type	Value
X-Admin-Protocol	2056	opaque	XXYYZZ

For new 'X-Admin-Protocol' option, the following properties are defined:

- Elective (non-critical) class according RFC 7252 section 5.4.1
- Safe-to-Forward
- Cache-key
- Not Repeatable
- Format: opaque
- Length: 3 bytes

The Value above SHALL be set to highest version of SGP.32 [This document] supported by the sender. When the sender is the IoT Device, this indicates the highest version supported by the IPA. The Value is coded as major/minor/revision version as binary value on three bytes 1/2/3, e.g. '01 03 00' for v1.3.0.

This option SHALL be used from version 1.3 upward. It MAY be omitted in previous versions.

- The <User Agent> SHALL NOT be included in the CoAP transport layer.
- The Content-Format option SHOULD be omitted. The Content Format to be used is determined by the application context. If the Content-Format option is included, it SHALL be set to 42 (application/octet-stream). The payload format is defined by the X-Admin-Protocol option and the corresponding version of this specification.
- Content-Length is not relevant to CoAP and SHALL be omitted. If the payload is too large to be sent in a single CoAP UDP packet, then block-wise CoAP SHALL be used as defined in RFC 7959 [12]. In this case the CoAP Request-Tag option SHALL be used as defined in RFC 9175 [13], in order to associate the separate blocks of a CoAP request together.

NOTE: It is RECOMMENDED to send CoAP/UDP messages in a single UDP Packet.

- The CoAP Echo Option SHALL be used to enable lightweight freshness verifications as defined in RFC 9175 [13]

### 6.3 ESipa function binding in ASN.1

This section presents the ASN.1 function binding for ESipa. The ASN.1 structures for ESipa messages between IPA and eIM, that are used in 6.1.1 and 6.2 respectively for the HTTP and CoAP interface bindings to select between ESipa functions, are presented in section

6.3.1. The request and response ASN.1 structures for each ESipa function are presented in sections 6.3.2 and 6.3.3. The ASN.1 structures in this section MAY be leveraged by other interface bindings, see for example Annex B.

Both eIM and IPAd SHALL support the ASN.1 encoding/decoding attribute "EXTENSIBILITY IMPLIED". This is useful when processing data definitions from a newer specification and to help for interoperability between entities of various vendors (to handle proprietary tag values).

### 6.3.1 ASN.1 structures for ESipa messages between IPA and eIM

The `EsipaMessageFromIpaToEim` and `EsipaMessageFromEimToIpa` ASN.1 structures that contain the ESipa function requests and responses are defined as follows:

```
-- ASN1START
EsipaMessageFromIpaToEim ::= CHOICE {
  initiateAuthenticationRequestEsipa [57] InitiateAuthenticationRequestEsipa, --
  Tag 'BF39'
  authenticateClientRequestEsipa [59] AuthenticateClientRequestEsipa, -- Tag 'BF3B'
  getBoundProfilePackageRequestEsipa [58] GetBoundProfilePackageRequestEsipa, --
  Tag 'BF3A'
  cancelSessionRequestEsipa [65] CancelSessionRequestEsipa, -- Tag 'BF41'
  handleNotificationEsipa [61] HandleNotificationEsipa, -- Tag 'BF3D'
  transferEimPackageResponse [78] TransferEimPackageResponse, -- Tag 'BF4E'
  getEimPackageRequest [79] GetEimPackageRequest, -- Tag 'BF4F'
  provideEimPackageResult [80] ProvideEimPackageResult -- Tag 'BF50'
}
EsipaMessageFromEimToIpa ::= CHOICE {
  initiateAuthenticationResponseEsipa [57] InitiateAuthenticationResponseEsipa, --
  Tag 'BF39'
  authenticateClientResponseEsipa [59] AuthenticateClientResponseEsipa, -- Tag
  'BF3B'
  getBoundProfilePackageResponseEsipa [58] GetBoundProfilePackageResponseEsipa, --
  Tag 'BF3A'
  cancelSessionResponseEsipa [65] CancelSessionResponseEsipa, -- Tag 'BF41'
  transferEimPackageRequest [78] TransferEimPackageRequest, -- Tag 'BF4E'
  getEimPackageResponse [79] GetEimPackageResponse, -- Tag 'BF4F'
  provideEimPackageResultResponse [80] ProvideEimPackageResultResponse -- Tag 'BF50'
}
-- ASN1STOP
```

### 6.3.2 ESipa functions provided by eIM

#### 6.3.2.1 ESipa.InitiateAuthentication

The request and response structures of `ESipa.InitiateAuthentication` are defined as follows:

```
-- ASN1START
InitiateAuthenticationRequestEsipa ::= [57] SEQUENCE { -- Tag 'BF39'
  euiccChallenge [1] Octet16, -- random eUICC challenge
  smdpAddress [3] UTF8String OPTIONAL,
  euiccInfo1 EUICCInfo1 OPTIONAL,
  eimTransactionId [2] TransactionId OPTIONAL
}

InitiateAuthenticationResponseEsipa ::= [57] CHOICE { -- Tag 'BF39'
  initiateAuthenticationOkEsipa InitiateAuthenticationOkEsipa,
  initiateAuthenticationErrorEsipa INTEGER {
    invalidDpAddress(1),
    euiccVersionNotSupportedByDp(2),
    ciPKIdNotSupported(3),
    smdpAddressMismatch(50),
    smdpOidMismatch(51),
  }
}
```

```

        invalidEimTransactionId(52),
        undefinedError(127)
    }
}
InitiateAuthenticationOkEsipa ::= SEQUENCE {
    transactionId [0] TransactionId OPTIONAL, -- The TransactionID generated by the
SM-DP+
    serverSigned1 ServerSigned1, -- Signed information
    serverSignature1 [APPLICATION 55] OCTET STRING, -- Server Sign1, Tag '5F37'
    euiccCiPKIdentifierToBeUsed OCTET STRING, -- Key identifier (possibly truncated)
of the CI Public Key to be used as required by ES10b.AuthenticateServer
    serverCertificate Certificate,
    matchingId UTF8String OPTIONAL,
    ctxParams1 [2] CtxParams1 OPTIONAL
}
-- ASN1STOP

```

### 6.3.2.2 ESipa.AuthenticateClient

The request and response structures of ESipa.AuthenticateClient are defined as follows:

```

-- ASN1START
AuthenticateClientRequestEsipa ::= [59] SEQUENCE { -- Tag 'BF3B'
    transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+/SM-
DS
    authenticateServerResponse [56] AuthenticateServerResponse -- This is the response
from ES10b.AuthenticateServer, possibly in compact format
}
AuthenticateClientResponseEsipa ::= [59] CHOICE { -- Tag 'BF3B'
    authenticateClientOkDPesipa AuthenticateClientOkDPesipa,
    authenticateClientOkDSEsipa AuthenticateClientOkDSEsipa,
    authenticateClientErrorEsipa INTEGER {
        eumCertificateInvalid(1),
        eumCertificateExpired(2),
        euiccCertificateInvalid(3),
        euiccCertificateExpired(4),
        euiccSignatureInvalid(5),
        matchingIdRefused(6),
        eidMismatch(7),
        noEligibleProfile(8),
        ciPKUnknown(9),
        invalidTransactionId(10),
        insufficientMemory(11), -- Note: values 12-17 are reserved
        downloadOrderExpired(18),
        pprNotAllowed(50),
        eventIdUnknown(56),
        undefinedError(127)
    }
}
AuthenticateClientOkDPesipa ::= SEQUENCE {
    transactionId [0] TransactionId OPTIONAL, -- The TransactionID generated by the
SM-DP+
    profileMetaData [37] StoreMetadataRequest OPTIONAL,
    smdpSigned2 SmdpSigned2, -- Signed information
    smdpSignature2 [APPLICATION 55] OCTET STRING, -- Tag '5F37'
    smdpCertificate Certificate, -- CERT.DPpb.ECDSA
    hashCc Octet32 OPTIONAL -- Hash of confirmation code
}
AuthenticateClientOkDSEsipa ::= SEQUENCE {
    transactionId [0] TransactionId, -- The TransactionID generated by the SM-DS
    profileDownloadTrigger [84] ProfileDownloadTriggerRequest OPTIONAL -- Tag 'BF54'
}
-- ASN1STOP

```

### 6.3.2.3 ESipa.GetBoundProfilePackage

The request and response structures of ESipa.GetBoundProfilePackage are defined as follows:

```
-- ASN1START
GetBoundProfilePackageRequestEsipa ::= [58] SEQUENCE { -- Tag 'BF3A'
    transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
    prepareDownloadResponse [33] PrepareDownloadResponse -- This is the response from
    ES10b.PrepareDownload, possibly in compact format
}
GetBoundProfilePackageResponseEsipa ::= [58] CHOICE { -- Tag 'BF3A'
    getBoundProfilePackageOkEsipa GetBoundProfilePackageOkEsipa,
    getBoundProfilePackageErrorEsipa INTEGER {
        euiccSignatureInvalid(1),
        confirmationCodeMissing(2),
        confirmationCodeRefused(3),
        confirmationCodeRetriesExceeded(4),
        bppRebindingRefused(5),
        deprecated(6), -- this value is no longer used
        metadataMismatch(50),
        invalidTransactionId(95),
        undefinedError(127)
    }
}
GetBoundProfilePackageOkEsipa ::= SEQUENCE {
    transactionId [0] TransactionId OPTIONAL, -- The TransactionID generated by the
    SM-DP+
    boundProfilePackage [54] BoundProfilePackage
}
-- ASN1STOP
```

### 6.3.2.4 ESipa.HandleNotification

The request structure of ESipa.HandleNotification is defined as follows (there is no response structure defined):

```
-- ASN1START
HandleNotificationEsipa ::= [61] CHOICE { -- Tag 'BF3D'
    pendingNotification [0] PendingNotification, -- A Notification to be delivered to
    a Notification Receiver, possibly in compact format
    provideEimPackageResult [80] ProvideEimPackageResult -- Tag 'BF50'
}
-- ASN1STOP
```

### 6.3.2.5 ESipa.CancelSession

The request and response structures of ESipa.CancelSession are defined as follows:

```
-- ASN1START
CancelSessionRequestEsipa ::= [65] SEQUENCE { -- Tag 'BF41'
    transactionId TransactionId, -- The TransactionID generated by the SM-DP+
    cancelSessionResponse CancelSessionResponse -- This is the response from ES10b.
    CancelSession function, possibly in compact format
}
CancelSessionResponseEsipa ::= [65] CHOICE { -- Tag 'BF41'
    cancelSessionOk CancelSessionOk,
    cancelSessionError INTEGER {
        invalidTransactionId(1),
        euiccSignatureInvalid(2),
        undefinedError(127)
    }
}
CancelSessionOk ::= SEQUENCE { -- This function has no output data
}
-- ASN1STOP
```

### 6.3.2.6 ESipa.GetEimPackage

The request and response structures of ESipa.GetEimPackage are defined as follows:

```
-- ASN1START
StateChangeCause ::= INTEGER {
    otherEim(0), -- for change(s) requested by another registered eIM
    fallback(1), -- for change(s) following fallback requested by IPA
    emergencyProfile(2), -- for change(s) following Emergency Profile swap requested
    by IPA
    local(3), -- for change(s) applied locally (RFU, not supported in this version)
    reset(4), -- for change(s) following an euicc reset
    immediateEnableProfile(5), -- for change(s) following immediate Profile enabling
    requested by IPA
    deviceChange(6), -- for change(s) detected by IP Ae following a IoT Device change
    (removable eSIMs)
    undefined(127) -- unknown cause
}

GetEimPackageRequest ::= [79] SEQUENCE { -- Tag 'BF4F'
    eidValue [APPLICATION 26] Octet16, -- Tag '5A'
    notifyStateChange [0] NULL OPTIONAL, -- Notification to the eIM that it should
    update its information about the eUICC (e.g. list of Profiles, Profile states...)
    stateChangeCause [1] StateChangeCause OPTIONAL, -- Provide the cause of the
    notified state change(s)
    rPLMN [2] OCTET STRING (SIZE(3)) OPTIONAL -- MCC and MNC of the last registered
    PLMN, coded as defined in 3GPP TS 24.008 [22]
}

GetEimPackageResponse ::= [79] CHOICE { -- Tag 'BF4F'
    euiccPackageRequest [81] EuiccPackageRequest, -- Tag 'BF51'
    ipaEuiccDataRequest [82] IpaEuiccDataRequest, -- Tag 'BF52'
    profileDownloadTriggerRequest [84] ProfileDownloadTriggerRequest, -- Tag 'BF54'
    eimPackageError INTEGER {
        noEimPackageAvailable(1),
        eidNotFound(2),
        invalidEid(3),
        missingEid(4),
        undefinedError(127)
    }
}
-- ASN1STOP
```

### 6.3.2.7 ESipa.ProvideEimPackageResult

The request and response structures of ESipa.ProvideEimPackageResult are defined as follows:

```
-- ASN1START
EimPackageResultErrorCode ::= INTEGER {
    invalidPackageFormat(1),
    unknownPackage(2),
    undefinedError(127)
}

EimPackageResultResponseError ::= SEQUENCE {
    eimTransactionId [0] TransactionId OPTIONAL,
    eimPackageResultErrorCode EimPackageResultErrorCode
}

EimPackageResult ::= CHOICE {
    euiccPackageResult [81] EuiccPackageResult, -- Tag 'BF51'
    ePRAndNotifications SEQUENCE {
        euiccPackageResult [81] EuiccPackageResult, -- Tag 'BF51'
        notificationList [0] PendingNotificationList -- Tag 'A0'
    },
    ipaEuiccDataResponse [82] IpaEuiccDataResponse, -- Tag 'BF52'
    profileDownloadTriggerResult [84] ProfileDownloadTriggerResult, -- Tag 'BF54'
}
```

```

    eimPackageResultResponseError [0] EimPackageResultResponseError
  }

ProvideEimPackageResult ::= [80] SEQUENCE { -- Tag 'BF50'
  eidValue [APPLICATION 26] Octet16 OPTIONAL, -- Tag '5A'
  eimPackageResult EimPackageResult
}

ProvideEimPackageResultResponse ::= [80] CHOICE { -- Tag 'BF50'
  eimAcknowledgements [83] EimAcknowledgements, -- Tag 'BF53'
  emptyResponse SEQUENCE {}, -- because EimAcknowledgements was OPTIONAL
  provideEimPackageResultError INTEGER {
    eidNotFound(2),
    invalidEid(3),
    missingEid(4),
    undefinedError(127)
  }
}
-- ASN1STOP

```

If `eimTransactionId` was present in the eIM Package, the IPA SHALL return the same `eimTransactionId` in the `ProvideEimPackageResult`.

### 6.3.3 ESipa functions provided by IPA

#### 6.3.3.1 ESipa.TransferEimPackage

The request and response structures of `ESipa.TransferEimPackage` are defined as follows:

```

-- ASN1START
TransferEimPackageRequest ::= [78] CHOICE { -- Tag 'BF4E'
  euiccPackageRequest [81] EuiccPackageRequest, -- Tag 'BF51'
  ipaEuiccDataRequest [82] IpaEuiccDataRequest, -- Tag 'BF52'
  eimAcknowledgements [83] EimAcknowledgements, -- Tag 'BF53'
  profileDownloadTriggerRequest [84] ProfileDownloadTriggerRequest -- Tag 'BF54'
}

TransferEimPackageResponse ::= [78] CHOICE { -- Tag 'BF4E'
  euiccPackageResult [81] EuiccPackageResult, -- Tag 'BF51'
  ePRAndNotifications SEQUENCE {
    euiccPackageResult [81] EuiccPackageResult, -- Tag 'BF51'
    notificationList [0] PendingNotificationList -- Tag 'A0'
  },
  ipaEuiccDataResponse [82] IpaEuiccDataResponse, -- Tag 'BF52'
  eimPackageReceived NULL,
  eimPackageReceivedWithCid [96] EimPackageReceivedWithCid,
  eimPackageError INTEGER {
    invalidPackageFormat(1),
    unknownPackage(2),
    undefinedError(127)
  },
  [97] EimPackageErrorWithCid
}

EimPackageReceivedWithCid ::= SEQUENCE {
  correlationId CHOICE {
    eimTransactionId [0] TransactionId, -- if received in the EimPackage
    eidValue [APPLICATION 26] Octet16 -- otherwise
  } OPTIONAL
}

EimPackageErrorWithCid ::= SEQUENCE {
  correlationId CHOICE {
    eimTransactionId [0] TransactionId, -- if received in the EimPackage
    eidValue [APPLICATION 26] Octet16 -- otherwise
  } OPTIONAL,
  eimPackageError EimPackageResultErrorCode
}
-- ASN1STOP

```

END  
 -- ASN1STOP

## 6.4 ESipa function binding in JSON

"format": "base64": unless specified otherwise below, the value of a JSON field of this format SHALL contain the base64 coding defined in RFC 4648 [71] of the DER encoded ASN.1 data object (including its tag and length fields), referenced in "description".

NOTE In most of the cases, the ASN.1 data object is defined in ES10x request/responses. Otherwise, the 'description' of the base64 field references the section where the ASN.1 type is specified.

"pattern": "^[0-9,A-F]{n,m}\$": specifies the hexadecimal representation of the data referred to in "description".

### 6.4.1 List of functions

Function	Path	MXP
InitiateAuthentication	/gsma/rsp2/esipa/initiateAuthentication	Synchronous
AuthenticateClient	/gsma/rsp2/esipa/authenticateClient	Synchronous
GetBoundProfilePackage	/gsma/rsp2/esipa/getBoundProfilePackage	Synchronous
TransferEimPackage	/gsma/rsp2/esipa/transferEimPackage	Synchronous
GetEimPackage	/gsma/rsp2/esipa/getEimPackage	Synchronous
ProvideEimPackageResult	/gsma/rsp2/esipa/provideEimPackageResult	Synchronous
HandleNotification	/gsma/rsp2/esipa/handleNotification	Notification
CancelSession	/gsma/rsp2/esipa/cancelSession	Synchronous

Table 28 List of functions

#### 6.4.1.1 ESipa.InitiateAuthentication

Only the characters that are mandatory to escape (Quotation mark, reverse solidus, and the control characters) SHALL be escaped in JSON string value as specified in RFC 7159 [36] section 7. Other characters SHALL NOT be escaped.

Hereunder is the definition of the JSON schema for the <JSON requestBody> corresponding to the "ESipa.InitiateAuthentication" function:

```
{
  "type": "object",
  "properties": {
    "euiccChallenge": {
      "type": "string",
      "format": "base64",
      "description": "base64 encoding of the value field of the eUICC Challenge as defined in SGP.22 [4] (without tag and length fields)"
    },
    "euiccInfo1": {
      "type": "string",
      "format": "base64",
      "description": "euiccinfo1 as defined in SGP.22 [4]"
    },
    "smdpAddress": {
      "type": "string",

```

```
    "description": "SM-DP+ Address as defined in SGP.22 [4]"
  },
  "eimTransactionId": {
    "type": "string",
    "pattern": "^[0-9,A-F]{2,32}$",
    "description": "Transaction ID received from the eIM in
ProfileDownloadTriggerRequest"
  }
},
"required": ["euiccChallenge"]
}
```

Hereunder is the definition of the JSON schema for the <JSON responseBody> corresponding to the "ESipa.InitiateAuthentication" function:

```
{
  "type": "object",
  "properties": {
    "transactionId": {
      "type": "string",
      "pattern": "^[0-9,A-F]{2,32}$",
      "description": "TransactionID as defined in SGP.22 [4]"
    },
    "serverSigned1": {
      "type": "string",
      "format": "base64",
      "description": "The data object as required by ES10b.AuthenticateServer"
    },
    "serverSignature1": {
      "type": "string",
      "format": "base64",
      "description": "The signature as required by ES10b.AuthenticateServer"
    },
    "euiccCiPKIdentifierToBeUsed": {
      "type": "string",
      "format": "base64",
      "description": "The Public Key Identifier (possibly truncated) of the CI
Public Key to be used as required by ES10b.AuthenticateServer"
    },
    "serverCertificate": {
      "type": "string",
      "format": "base64",
      "description": "The server Certificate as required by
ES10b.AuthenticateServer"
    },
    "matchingId": {
      "type": "string",
      "description": "MatchingID as defined in SGP.22 [4]"
    },
    "ctxParams1": {
      "type": "string",
      "format": "base64",
      "description": "ctxParams1 as required by ES10b.AuthenticateServer"
    }
  },
  "required": ["serverSigned1", "serverSignature1", "serverCertificate",
"euiccCiPKIdentifierToBeUsed"]
}
```

NOTE: IPA is in charge of transcoding the transactionId.

#### 6.4.1.2 ESipa.AuthenticateClient

Hereunder is the definition of the JSON schema for the <JSON requestBody> corresponding to the "ESipa.AuthenticateClient" function:

```
{
  "type": "object",
  "properties": {
    "transactionId": {
      "type": "string",
      "pattern": "^[0-9,A-F]{2,32}$",
      "description": "TransactionID as defined in SGP.22 [4]"
    },
    "authenticateServerResponse": {
      "type": "string",
      "format": "base64",
      "description": "AuthenticateServerResponse as provided by ES10b.AuthenticateServer, possibly in compact format"
    }
  },
  "required": ["transactionId", "authenticateServerResponse"]
}
```

Hereunder is the definition of the JSON schema for the <JSON responseBody> corresponding to the "ESipa.AuthenticateClient" function:

```
{
  "type": "object",
  "oneOf": [
    {
      "properties": {
        "transactionId": {
          "type": "string",
          "pattern": "^[0-9,A-F]{2,32}$",
          "description": "TransactionID as defined in SGP.22 [4]"
        },
        "profileMetadata": {
          "type": "string",
          "format": "base64",
          "description": "StoreMetadataRequest as defined SGP.22 [4]"
        },
        "smdpSigned2": {
          "type": "string",
          "format": "base64",
          "description": "SmdpSigned2 encoded data object"
        },
        "smdpSignature2": {
          "type": "string",
          "format": "base64",
          "description": "SM-DP+ signature as required by ES10b.PrepareDownload"
        },
        "smdpCertificate": {
          "type": "string",
          "format": "base64",
          "description": "The Certificate as required by ES10b.PrepareDownload"
        }
      }
    }
  ]
}
```

```

        "hashCc": {
            "type": "string",
            "format": "base64",
            "description": "hashCc as required by ES10b.PrepareDownload"
        },
        "required" : ["smdpSigned2", "smdpSignature2", "smdpCertificate"]
    }, {
        "properties": {
            "transactionId": {
                "type": "string",
                "pattern": "^[0-9,A-F]{2,32}$",
                "description": "TransactionID as defined in SGP.22 [4]"
            },
            "profileDownloadTrigger": {
                "type": "string",
                "format": "base64",
                "description": "ProfileDownloadTriggerRequest as defined in section
2.11.1.3"
            }
        },
        "required" : ["transactionId"]
    }
]
}
    
```

#### 6.4.1.3 ESipa.GetBoundProfilePackage

Hereunder is the definition of the JSON schema for the <JSON requestBody> corresponding to the "ESipa.GetBoundProfilePackage" function:

```

{
    "type": "object",
    "properties": {
        "transactionId": {
            "type": "string",
            "pattern": "^[0-9,A-F]{2,32}$",
            "description": "TransactionID as defined in SGP.22 [4]"
        },
        "prepareDownloadResponse": {
            "type": "string",
            "format": "base64",
            "description": "PrepareDownloadResponse as provided by
ES10b.PrepareDownload, possibly in compact format"
        }
    },
    "required": ["transactionId", "prepareDownloadResponse"]
}
    
```

Hereunder is the definition of the JSON schema for the <JSON responseBody> corresponding to the "ESipa.GetBoundProfilePackage" function:

```

{
    "type": "object",
    "properties": {
        "transactionId": {
            "type": "string",
            "pattern": "^[0-9,A-F]{2,32}$",
            "description": "TransactionID as defined in SGP.22 [4]"
        }
    }
}
    
```

```

    },
    "boundProfilePackage": {
      "type": "string",
      "format": "base64",
      "description": "Bound Profile Package required by
ES10b.LoadBoundProfilePackage"
    }
  },
  "required": ["boundProfilePackage"]
}

```

#### 6.4.1.4 ESipa.TransferEimPackage

Hereunder is the definition of the JSON schema for the <JSON requestBody> corresponding to the "ESipa.TransferEimPackage" function:

```

{
  "type": "object",
  "oneOf": [
    {
      "properties": {
        "euiccPackageRequest": {
          "type": "string",
          "format": "base64",
          "description": "EuiccPackageRequest as defined in section 2.11.1.1.1"
        }
      },
      "required": ["euiccPackageRequest"]
    },
    {
      "properties": {
        "ipaEuiccDataRequest": {
          "type": "string",
          "format": "base64",
          "description": "IpaEuiccDataRequest as defined in section 2.11.1.2"
        }
      },
      "required": ["ipaEuiccDataRequest"]
    },
    {
      "properties": {
        "eimAcknowledgements": {
          "type": "string",
          "format": "base64",
          "description": "EimAcknowledgements as defined in section 2.11.1.4"
        }
      },
      "required": ["eimAcknowledgements"]
    },
    {
      "properties": {
        "profileDownloadTriggerRequest": {
          "type": "string",
          "format": "base64",
          "description": "ProfileDownloadTriggerRequest as defined in section
2.11.2.3"
        }
      },
      "required": ["profileDownloadTriggerRequest"]
    }
  ]
}

```

Hereunder is the definition of the JSON schema for the <JSON responseBody> corresponding to the "ESipa.TransferEimPackage" function:

```
{
  "type": "object",
  "oneOf": [
    {
      "properties": {
        "euiccPackageResult": {
          "type": "string",
          "format": "base64",
          "description": "EuiccPackageResult as defined in section 2.11.2.1"
        }
      },
      "required": ["euiccPackageResult"]
    }, {
      "properties": {
        "ePRAndNotifications": {
          "type": "object",
          "properties": {
            "euiccPackageResult": {
              "type": "string",
              "format": "base64",
              "description": "EuiccPackageResult as defined in section
2.11.2.1"
            },
            "notificationList": {
              "type": "string",
              "format": "base64",
              "description": "PendingNotificationList as defined in section
2.11.2.2"
            }
          },
          "required": ["euiccPackageResult", "notificationList"]
        }
      },
      "required": ["ePRAndNotifications"]
    }, {
      "properties": {
        "ipaEuiccDataResponse": {
          "type": "string",
          "format": "base64",
          "description": "IpaEuiccDataResponse as defined in section 2.11.2.2"
        }
      },
      "required": ["ipaEuiccDataResponse"]
    }, {
      "properties": {
        "eimPackageReceived": {
          "type": "object",
          "description": "NULL response as defined in section 6.3.3.1",
          "properties": {
          }
        }
      },
      "required": ["eimPackageReceived"]
    }, {
      "properties": {
        "eimPackageReceivedWithCid": {
```

```

        "type": "object",
        "description": "eimPackageReceivedWithCid as defined in section
6.3.3.1",
        "properties": {
            }
        },
        "required": ["eimPackageReceivedWithCid"]
    },{
        "properties": {
            "eimPackageError": {
                "type": "integer",
                "description": "eimPackageError code as defined in section 6.3.3.1"
            }
        },
        "required": ["eimPackageError"]
    },{
        "properties": {
            "eimPackageErrorWithCid": {
                "type": "string",
                "format": "base64",
                "description": "eimPackageErrorWithCid as defined in section 6.3.3.1"
            }
        },
        "required": ["eimPackageErrorWithCid"]
    }
}
    ]
}

```

#### 6.4.1.5 ESipa.GetEimPackage

Hereunder is the definition of the JSON schema for the <JSON requestBody> corresponding to the "ESipa.GetEimPackage" function:

```

{
    "type": "object",
    "properties": {
        "eidValue": {
            "type": "string",
            "pattern": "^([0-9]{32})$",
            "description": "EID as described in SGP.22 [4]"
        },
        "notifyStateChange": {
            "type": "boolean",
            "description": "Notification to the eIM that it should update its
information about the eUICC (e.g. list of profiles, profile states...)"
        },
        "stateChangeCause": {
            "type": "integer",
            "description": "Provide the cause of the notified state change(s), as
defined at section 6.3.2.6"
        },
        "rPlmn": {
            "type": "string",
            "format": "base64",
            "description": "MCC and MNC of the last registered PLMN, coded as defined
in 3GPP TS 24.008 [22]"
        }
    },
}

```

```
"required": ["eidValue"]  
}
```

Hereunder is the definition of the JSON schema for the <JSON responseBody> corresponding to the "ESipa.GetEimPackage" function:

```
{  
  "type": "object",  
  "oneOf": [  
    {  
      "properties": {  
        "euiccPackageRequest": {  
          "type": "string",  
          "format": "base64",  
          "description": "EuiccPackageRequest as defined in section 2.11.1.1.2"  
        }  
      },  
      "required": ["euiccPackageRequest"]  
    },  
    {  
      "properties": {  
        "ipaEuiccDataRequest": {  
          "type": "string",  
          "format": "base64",  
          "description": "IpaEuiccDataRequest as defined in section 2.11.1.2"  
        }  
      },  
      "required": ["ipaEuiccDataRequest"]  
    },  
    {  
      "properties": {  
        "profileDownloadTriggerRequest": {  
          "type": "string",  
          "format": "base64",  
          "description": "ProfileDownloadTriggerRequest as defined in section  
2.11.2.3"  
        }  
      },  
      "required": ["profileDownloadTriggerRequest"]  
    },  
    {  
      "properties": {  
        "eimPackageError": {  
          "type": "integer",  
          "description": "eimPackageError code as defined in section 6.3.2.6"  
        }  
      },  
      "required": ["eimPackageError"]  
    }  
  ]  
}
```

#### 6.4.1.6 ESipa.ProvideEimPackageResult

Hereunder is the definition of the JSON schema for the <JSON requestBody> corresponding to the "ESipa.ProvideEimPackageResult" function:

```
{  
  "type": "object",  
  "properties": {  
    "eidValue": {
```

```
    "type" : "string",
    "pattern" : "^[0-9]{32}$",
    "description" : "EID as described in SGP.22 [4]"
  },
  "eimPackageResult": {
    "type": "string",
    "format": "base64",
    "description": "EimPackageResult as defined in section 6.3.2.7"
  }
},
"required": ["eimPackageResult"]}
```

Hereunder is the definition of the JSON schema for the <JSON responseBody> corresponding to the "ESipa.ProvideEimPackageResult" function:

```
{
  "type": "object",
  "properties": {
    "eimAcknowledgements": {
      "type": "string",
      "format": "base64",
      "description": "EimAcknowledgements as defined in section 2.11.1.4"
    }
  }
}
```

#### 6.4.1.7 ESipa.HandleNotification

Hereunder is the definition of the JSON schema for the <JSON requestBody> corresponding to the "ESipa.HandleNotification" function:

```
{
  "type" : "object",
  "oneOf": [
    {
      "properties" : {
        "pendingNotification" : {
          "type" : "string",
          "format" : "base64",
          "description" : "PendingNotification as defined in section 5.14.7"
        }
      },
      "required" : ["pendingNotification"]
    },{
      "properties": {
        "provideEimPackageResult": {
          "type": "string",
          "format": "base64",
          "description": "ProvideEimPackageResult as defined in section
6.3.2.7"
        }
      },
      "required": ["provideEimPackageResult"]
    }
  ]
}
```

### 6.4.1.8 ESipa.CancelSession

Hereunder is the definition of the JSON schema for the <JSON requestBody> corresponding to the "ESipa.CancelSession" function:

```
{
  "type": "object",
  "properties": {
    "transactionId": {
      "type": "string",
      "pattern": "^[0-9,A-F]{2,32}$",
      "description": "TransactionID as defined in SGP.22 [4]"
    },
    "cancelSessionResponse": {
      "type": "string",
      "format": "base64",
      "description": "CancelSessionResponse as provided by ES10b.CancelSession"
    }
  },
  "required": ["transactionId", "cancelSessionResponse"]
}
```

"ESipa.CancelSession" function has no <JSON responseBody>.

## Annex A IoT Device Requirements (Normative)

### A.1 Functional IoT Device Requirements

Functional IoT Device Requirements No.	Requirement
IoTDEV1	The IoT Device SHALL support the following proactive command: REFRESH Command (At least mode 4 – "UICC Reset"), as defined in ETSI TS 102 223 [5].

**Table 29 IoT Device requirements**

### A.2 Support for CAT Mechanisms

Dependent on the deployment, the IoT Device SHALL support at least the CAT mechanisms (ETSI TS 102 223 [5]) indicated in the table below.

CAT mechanism	IPAd	IP Ae
TERMINAL PROFILE	X	X
SET UP EVENT LIST	X	X
REFRESH with "UICC Reset" or "eUICC Profile State Change" mode	X	X
PROVIDE LOCAL INFORMATION (IMEI)		X
PROVIDE LOCAL INFORMATION (Location Information) EVENT DOWNLOAD - Location status		X
PROVIDE LOCAL INFORMATION (current date, time and time zone)		X
SEND SHORT MESSAGE ENVELOPE (SMS-PP DOWNLOAD)	X	X

TIMER MANAGEMENT ENVELOPE (TIMER EXPIRATION)		X
OPEN CHANNEL related to packet data service bearer	X	X
CLOSE CHANNEL RECEIVE DATA SEND DATA GET CHANNEL STATUS EVENT DOWNLOAD - Data available EVENT DOWNLOAD - Channel status	X	X

**Table 30 CAT Mechanisms**

NOTE: The table also includes requirements for ES6.

## Annex B Mapping of procedures to IoT protocols (Informative)

### B.1 LwM2M

In case of LwM2M implementation of the ESipa it is expected that the eIM is a component that is integrated into Device Management platform and the IPA is a component that is part of the Device Management Client functionality in the IoT Device. For efficient message transfer LwM2M 1.1 and higher protocol version is RECOMMENDED.

#### B.1.1 Secure connection

A secure connection needs to be ensured so that ESipa security requirements are fulfilled. Specifically, LwM2M unsecured mode is not used unless appropriate lower-layer provided security is used, that the LwM2M server is in the same security realm as the element terminating the lower-layer security between it and the IoT Device, and it is ensured that only elements within the same security realm can access the unsecured traffic.

#### B.1.2 LwM2M object

The following object has been defined and registered to handle the ESipa implementation in LwM2M, see OMA LwM2M Object and Resource Registry [30].

#### Object definition

Name	Object ID	Object Version	LWM2M Version
eSIM IoT	3443	1.0	1.0
Object URN		Instances	Mandatory
urn:oma:lwm2m:ext:3443		Single	Optional

**Table 31 Object Definition**

#### Resource definition

ID	Name	Operations	Instances	Mandatory	Type	Range or Enumeration	Units	Description
0	State	R	Single	Mandatory	Integer			Status 0 – ready to receive eIM message

							1 – not able to receive eIM message  Note: In case value is >0 then writing to resource 2 SHALL NOT be possible
1	EID	R	Single	Mandatory	String		EID of the eUICC that is target of the object
2	eIM message	W	Single	Mandatory	Opaque		the message from eIM to IPA. The message SHALL be in the ASN.1 DER format defined in section 6.3.1
3	IPA message	R	Single	Mandatory	Opaque		The message from IPA to eIM. The message SHALL be in the ASN.1 DER format defined in section 6.3.1
4	Clear IPA message	E	Single	Mandatory			In case Send or Confirmable Notifications is not used to deliver IPA message then execution of this resource will confirm delivery of the IPA message and clear the resource 3.

**Table 32 Resource Definition**

### B.1.3 Procedures

For purpose of simplification, the IPA in this section will represent LwM2M client integrating also IPA functionality and eIM will represent LwM2M server integrating also eIM functionality.

#### B.1.3.1 Transferring messages between eIM and IPA (LwM2M)

Following procedure is an example generic implementation of transfer of an eIM message to IPA and IPA message to eIM using LwM2M. This example uses functions introduced in LwM2M version 1.1.

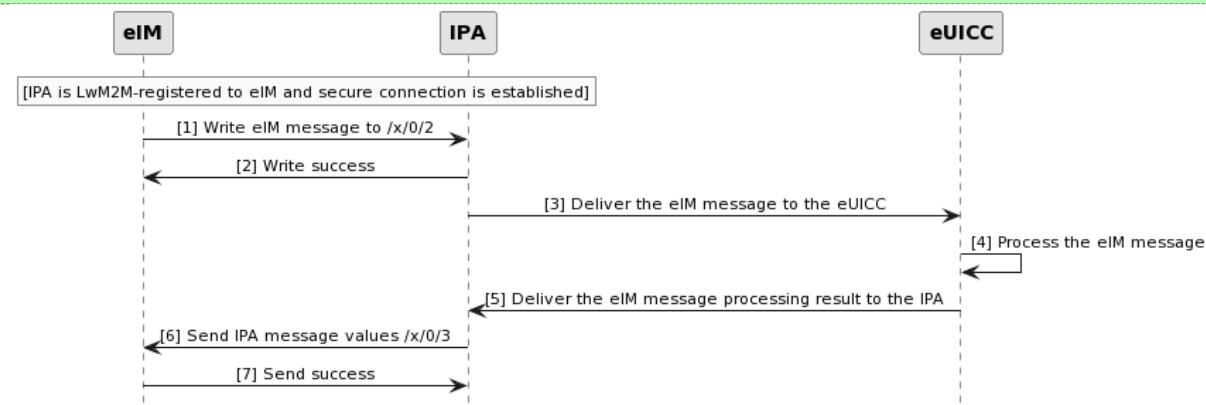
```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>eIM" as EIM
participant "<b>IPA" as IPA
participant "<b>eUICC" as E

rnote over EIM, IPA : [IPA is LwM2M-registered to eIM and secure connection is established]

EIM -> IPA : [1] Write eIM message to /x/0/2
IPA -> EIM : [2] Write success
IPA -> E : [3] Deliver the eIM message to the eUICC
E -> E : [4] Process the eIM message
E -> IPA : [5] Deliver the eIM message processing result to the IPA
IPA -> EIM : [6] Send IPA message /x/0/3
EIM -> IPA : [7] Send success
```

@endum1



**Figure 31 Transferring messages between eIM and IPA (LwM2M)**

**Start Conditions:**

The LwM2M client in IPA is registered to the LwM2M server in eIM and secure connection between the client and the server is established.

**Procedure:**

1. The eIM sends Write request with the eIM message in resource /x/0/2.
2. The IPA confirms delivery of the Write request.
3. The IPA provides to the eUICC the message from eIM with provided parameters.
4. The eUICC processes the eIM message.
5. The eUICC returns the result of message processing to the IPA.
6. The IPA uses the Send request to deliver the IPA message in resource /x/0/3.
7. The eIM confirms delivery of the Send message and the content of resource /x/0/3 is cleared in the IPA.

**B.2 MQTT**

In case of an MQTT implementation of the ESipa, it is expected that the eIM is a component that is integrated into a device management platform and the IPA is a component that is part of the device management client functionality in the IoT Device. For efficient message transfer MQTT v5.0 [23] or higher is RECOMMENDED.

NOTE: MQTT is running over TCP.

**B.2.1 Mapping of MQTT Roles**

Both the entity hosting the eIM and the IoT Device implementing the IPA will act as clients of the MQTT broker and both will be able to publish and subscribe to topics.

For the IPA to connect to the MQTT broker, the FQDN of the MQTT broker is used and configured in the `eimFqdn` within the `EimConfigurationData`.

**B.2.2 Secure connection**

The security of the MQTTs connection between involved entities (i.e. IPA and eIM) and the MQTT broker has to be ensured by using TLS transport security with certificate authentication. The security of the MQTT application level uses the same certificate for

authenticating the client to the MQTT broker. MQTT broker uses port 8883 as default MQTT secure port.

NOTE: MQTT v5.0 [23] uses TLS 1.2 as defined in RFC 5246 [8].

### B.2.3 Description

The following topics are proposed to handle the ESipa implementation in MQTT. The topics are focused on transferring eIM Packages and eIM Package Results between eIM and IPA and direct Profile download is assumed.

Name	Topic	Description
eIM message	eim-to-ipa/{eid}	Topic used by the eIM to publish an EsipaMessageFromEimToIpa to an IoT Device
IPA message	ipa-to-eim	Topic used by the IPA to publish an EsipaMessageFromIpaToEim to the eIM

**Table 33 eIM and IPA Message**

#### eIM Message

Publisher: eIM

Subscriber: IPA of the matching eID

Usage: This topic is used by the eIM to publish the requests to the IPA. The IPA will be subscribed and receive all the requests in order to process them.

Topic: eim- to-ipa/{eid}

Parameter	Type	Description
EsipaMessageFromEimToIpa	Binary	EsipaMessageFromEimToIpa following the ASN.1 structure as described in section 6.3.1.

**Table 34 EsipaMessageFromEimToIpa**

#### IPA message

Publisher: IPA

Subscriber: eIM

Usage: This topic is used by the IPA to publish the responses to the eIM requests. The eIM will be subscribed and receive all the responses from the processed requests.

Topic: ipa-to-eim

Parameter	Type	Description
EsipaMessageFromIpaToEim	Binary	EsipaMessageFromIpaToEim following the ASN.1 structure as described in section 6.3.1.

**Table 35 EsipaMessageFromIpaToEim**

## B.2.4 Procedures

### B.2.4.1 Connection

```

@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid
participant "<b>eIM" as EIM
participant "<b>MQTT broker" as BROKER
participant "<b>IPA" as IPA
participant "<b>eUICC" as E
note over EIM, BROKER : [1] [TLS session establishment]
EIM -> BROKER: [2] Subscribe to ipa-to-eim

note over IPA : [IoT Device internal trigger (e.g., power on, reconnection, etc.)]
alt IoT Device power on
IPA -> E: [3] ES10b.GetEid
activate E
E --> IPA: eid
deactivate E
end
note over IPA, BROKER : [4] [TLS session establishment]
IPA -> BROKER: [5] Subscribe to eim-to-ipa/{eid}
@enduml
    
```

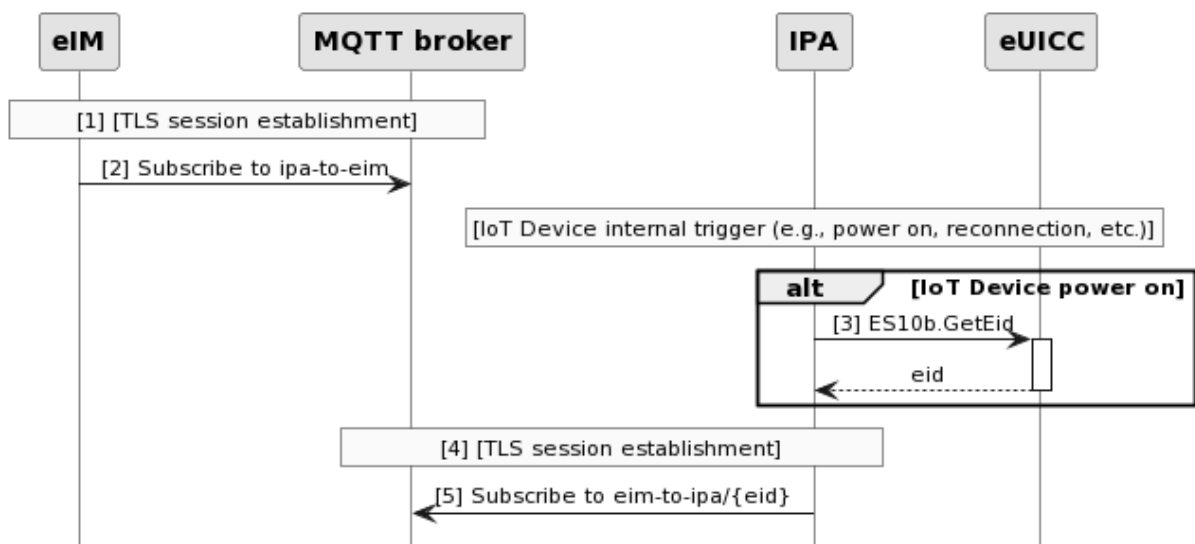


Figure 32 Connection

1. Establish TLS session between the eIM and MQTT broker.
2. eIM subscribes to the ipa-to-eim topic to be able to receive IPA responses.
3. IPA reads the eUICC EID using ES10b.GetEID.

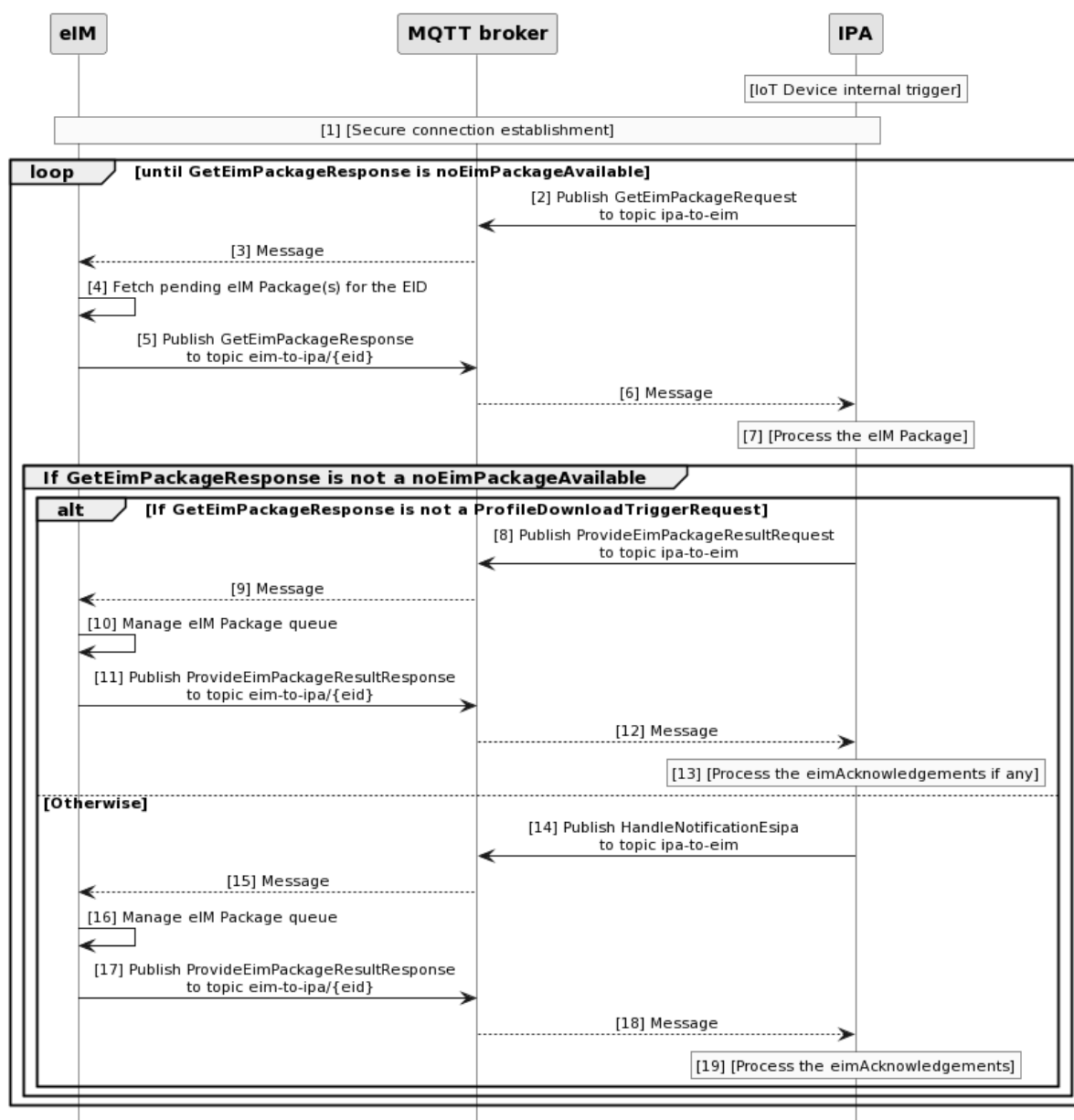
4. TLS session is established between the IPA and the MQTT broker.
5. IPA subscribes to the eim-to-ipa/{eid} to receive the eIM request.

#### B.2.4.2 Communication (Retrieval)

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>eIM" as EIM
participant "<b>MQTT broker" as BROKER
participant "<b>IPA" as IPA

rnote over IPA: [IoT Device internal trigger]
rnote over EIM, IPA : [1] [Secure connection establishment]
loop until GetEimPackageResponse is noEimPackageAvailable
IPA -> BROKER : [2] Publish GetEimPackageRequest \n to topic ipa-to-eim
BROKER --> EIM : [3] Message
EIM -> EIM : [4] Fetch pending eIM Package(s) for the EID
EIM -> BROKER : [5] Publish GetEimPackageResponse \n to topic eim-to-ipa/{eid}
BROKER --> IPA : [6] Message
rnote over IPA : [7] [Process the eIM Package]
group If GetEimPackageResponse is not a noEimPackageAvailable
alt If GetEimPackageResponse is not a ProfileDownloadTriggerRequest
IPA -> BROKER : [8] Publish ProvideEimPackageResultRequest \n to topic ipa-to-eim
BROKER --> EIM : [9] Message
EIM -> EIM : [10] Manage eIM Package queue
EIM -> BROKER : [11] Publish ProvideEimPackageResultResponse \n to topic eim-to-
ipa/{eid}
BROKER --> IPA : [12] Message
rnote over IPA : [13] [Process the eimAcknowledgements if any]
else Otherwise
IPA -> BROKER : [14] Publish HandleNotificationEsipa \n to topic ipa-to-eim
BROKER --> EIM : [15] Message
EIM -> EIM : [16] Manage eIM Package queue
EIM -> BROKER : [17] Publish ProvideEimPackageResultResponse \n to topic eim-to-
ipa/{eid}
BROKER --> IPA : [18] Message
rnote over IPA : [19] [Process the eimAcknowledgements]
end
end
end
end
@enduml
```



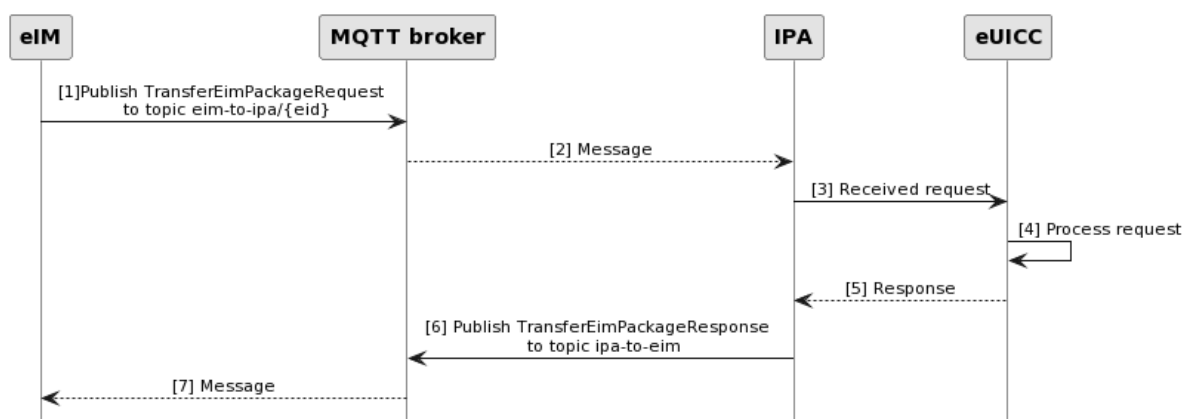
**Figure 33 Communication (Retrieval)**

1. Secure connection is established.
2. IPA publishes GetEimPackageRequest to topic ipa-to-eim.
3. Message is transferred to eIM.
4. eIM fetches the pending eIM Package(s) for the received EID.
5. eIM publishes the GetEimPackageResponse to topic eim-to-ipa/{eid}.
6. Message is transferred to the IPA.
7. The IPA processes the eIM Package by forwarding it to the eUICC. The eUICC processes the request and prepares the eIM Package Result and sends the result to the IPA.

8. If the GetEimPackageResponse is a ProfileDownloadTriggerRequest, the procedure continues in step 14, otherwise the IPA publishes the ProvideEimPackageResult in ipa-to-eim.
9. Message is transferred to the eIM.
10. eIM manages the eIM Package queue.
11. The eIM publishes ProvideEimPackageResultResponse to eim-to-ipa/{eid}.
12. Message is sent to the IPA.
13. The IPA processes the eimAcknowledgements (if any).
14. The IPA publishes HandleNotificationEsipa.
15. Message is transferred to the eIM.
16. The eIM manages the eIM Package queue.
17. the eIM publishes ProvideEimPackageResultResponse to topic eim-to-ipa/{eid}.
18. Message is transferred to the IPA.
19. The IPA processes the eimAcknowledgement.

### B.2.4.3 Communication (Injection)

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid
participant "<b>eIM" as EIM
participant "<b>MQTT broker" as BROKER
participant "<b>IPA" as IPA
participant "<b>eUICC" as E
EIM->BROKER: [1] Publish TransferEimPackageRequest \n to topic eim-to-ipa/{eid}
BROKER-->IPA: [2] Message
IPA->E: [3] Received request
E->E: [4] Process request
E-->IPA: [5] Response
IPA->BROKER: [6] Publish TransferEimPackageResponse \n to topic ipa-to-eim
BROKER-->EIM: [7] Message
@enduml
```



**Figure 34 Communication (injection)**

1. eIM publishes TransferEimPackageRequest to topic ipa-to-ipa/{eid}.
2. Message is transferred to the IPA.
3. The IPA forwards the received request to the eUICC.
4. The eUICC processes the request and prepares the eIM Package Result.
5. The eUICC sends the response to the IPA.
6. The IPA publishes the TransferEimPackageResponse in ipa-to-eim.
7. Message is transferred to the eIM.

## Annex C ASN.1 Definitions (Normative)

```

SGP32Definitions {joint-iso-itu-t(2) international-organizations(23) gsma(146)
rsp(1) asnlmodules(1) sgp32v1(31)}
DEFINITIONS
AUTOMATIC TAGS
EXTENSIBILITY IMPLIED ::=
BEGIN

IMPORTS Certificate, SubjectPublicKeyInfo FROM PKIX1Explicit88 {iso(1) identified-
organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-
pkix1-explicit(18)}
SubjectKeyIdentifier FROM PKIX1Implicit88 {iso(1) identified-organization(3) dod(6)
internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-pkix1-implicit(19)}
CancelSessionReason, ServerSigned1, RspCapability, Iccid, TransactionId,
ProfileInfoListRequest, RulesAuthorisationTable, EUICCInfo1, DeviceInfo,
VersionType, UICCCapability, PprIds, CertificationDataObject, Octet1, Octet16,
Octet32, PrepareDownloadResponseOk, PrepareDownloadResponseError,
AuthenticateResponseError, CtxParams1, ProfileInstallationResultData,
OtherSignedNotification, EuiccSignPIR, ErrorResult, NotificationMetadata,
CancelSessionResponseOk, SmdpSigned2, BoundProfilePackage, NotificationEvent,
VendorSpecificExtension, DpProprietaryData, OperatorId,
NotificationConfigurationInformation, ProfileClass, IconType, ProfileState,
OctetTo16 FROM RSPDefinitions {joint-iso-itu-t(2) international-organizations(23)
gsma(146) rsp(1) asnlmodules(1) sgp22v2(2)};
EuiccPackageRequest ::= [81] SEQUENCE { -- Tag 'BF51', #SupportedForPsmoV1.0.0#
    euiccPackageSigned EuiccPackageSigned,
    eimSignature [APPLICATION 55] OCTET STRING -- Tag '5F37'
}

EuiccPackageSigned ::= SEQUENCE {

```

```
eimId [0] UTF8String (SIZE(1..128)),
eidValue [APPLICATION 26] Octet16, -- Tag '5A'
counterValue [1] INTEGER,
eimTransactionId [2] TransactionId OPTIONAL,
euiccPackage EuiccPackage
}

EuiccPackage ::= CHOICE {
  psmoList SEQUENCE OF Psmo, -- #SupportedForPsmoV1.0.0#
  ecoList SEQUENCE OF Eco -- #SupportedForEcoV1.0.0#
}

EimConfigurationData ::= SEQUENCE {
  eimId [0] UTF8String (SIZE(1..128)), -- eimId of eIM
  eimFqdn [1] UTF8String OPTIONAL, -- FQDN of eIM or intermediate server, if used
  eimIdType [2] EimIdType OPTIONAL,
  counterValue [3] INTEGER OPTIONAL, -- initial counterValue for the eIM
  associationToken [4] INTEGER OPTIONAL,
  eimPublicKeyData [5] CHOICE {
    eimPublicKey SubjectPublicKeyInfo, -- public key of eIM, used for eUICC
    Package signature verification, where the encoding follows X.509 standard
    eimCertificate Certificate -- certificate of eIM, used for eUICC Package
    signature verification, where the encoding follows X.509 standard
  } OPTIONAL,
  trustedPublicKeyDataTls [6] CHOICE {
    trustedEimPkTls SubjectPublicKeyInfo, -- public key of eIM, used for TLS or
    DTLS, where the encoding follows X.509 standard
    trustedCertificateTls Certificate -- either the certificate of eIM, used for
    (D)TLS, or the certificate of the CA, where the encoding follows X.509 standard
  } OPTIONAL,
  eimSupportedProtocol [7] EimSupportedProtocol OPTIONAL,
  euiccCipKeyId [8] SubjectKeyIdentifier OPTIONAL, -- CI Public Key Identifier
  supported on the eUICC for signature creation
  indirectProfileDownload [9] NULL OPTIONAL, -- support of Indirect Profile
  download with indicated EimSupportedProtocol
  eSipaProprietaryProtocolInformation [10] VendorSpecificExtension OPTIONAL --
  Additional information for proprietary protocol(s)
}

EimIdType ::= INTEGER {
  eimIdTypeOid(1),
  eimIdTypeFqdn(2),
  eimIdTypeProprietary(3)
}

EimSupportedProtocol ::= BIT STRING {
  eimRetrieveHttps(0),
  eimRetrieveCoaps(1),
  eimInjectHttps(2),
  eimInjectCoaps(3),
  eimProprietary(4)
}

Eco ::= CHOICE {
  addEim [8] EimConfigurationData, -- for eIM configuration data see 2.11.1
  deleteEim [9] SEQUENCE {eimId [0] UTF8String},
  updateEim [10] EimConfigurationData,
  listEim [11] SEQUENCE {}
}

Psmo ::= CHOICE {
  enable [3] SEQUENCE {
    iccid [APPLICATION 26] Iccid,
    rollbackFlag NULL OPTIONAL
  },
  disable [4] SEQUENCE {iccid [APPLICATION 26] Iccid},
  delete [5] SEQUENCE {iccid [APPLICATION 26] Iccid},
  listProfileInfo [45] ProfileInfoListRequest, -- Tag 'BF2D'
  getRAT [6] SEQUENCE {},
  configureImmediateEnable [7] SEQUENCE {
    immediateEnableFlag [0] NULL OPTIONAL,
```

```

        defaultSmdpOid [1] OBJECT IDENTIFIER OPTIONAL,
        defaultSmdpAddress [2] UTF8String OPTIONAL
    },
    setFallbackAttribute [8] SEQUENCE { iccid [APPLICATION 26] Iccid},
    unsetFallbackAttribute [9] SEQUENCE {},
    setDefaultDpAddress [101] SetDefaultDpAddressRequest -- Tag 'BF65'
}
IpaEuiccDataRequest ::= [82] SEQUENCE { -- Tag BF52
    tagList [APPLICATION 28] OCTET STRING, -- Tag '5C'
    euiccCiPKIdentifierToBeUsed OCTET STRING OPTIONAL, -- CI Public Key Identifier
    (possibly truncated) supported on the eUICC for signature creation
    searchCriteriaNotification [1] CHOICE {
        seqNumber [0] INTEGER,
        profileManagementOperation [1] NotificationEvent
    } OPTIONAL,
    searchCriteriaEuiccPackageResult [2] CHOICE {
        seqNumber [0] INTEGER
    } OPTIONAL,
    eimTransactionId [3] TransactionId OPTIONAL -- Tag '83'
}
ProfileDownloadTriggerRequest ::= [84] SEQUENCE { -- Tag 'BF54'
    profileDownloadData [0] ProfileDownloadData OPTIONAL,
    eimTransactionId [2] TransactionId OPTIONAL
}
ProfileDownloadData ::= CHOICE {
    activationCode [0] UTF8String (SIZE(0..255)),
    contactDefaultSmdp [1] NULL,
    contactSmds [2] SEQUENCE {
        smdsAddress UTF8String OPTIONAL
    }
}
EimAcknowledgements ::= [83] SEQUENCE OF SequenceNumber -- Tag BF53
SequenceNumber ::= [0] INTEGER
EuiccPackageResult ::= [81] CHOICE { -- Tag 'BF51' #SupportedForPsmoV1.0.0#
    euiccPackageResultSigned EuiccPackageResultSigned,
    euiccPackageErrorSigned EuiccPackageErrorSigned,
    euiccPackageErrorUnsigned EuiccPackageErrorUnsigned
}
EuiccPackageResultSigned ::= SEQUENCE {
    euiccPackageResultDataSigned EuiccPackageResultDataSigned,
    euiccSignEPR [APPLICATION 55] OCTET STRING -- Tag '5F37'
}
EuiccPackageResultDataSigned ::= SEQUENCE { -- #SupportedForPsmoV1.0.0#
    eimId [0] UTF8String (SIZE(1..128)),
    counterValue [1] INTEGER,
    eimTransactionId[2] TransactionId OPTIONAL,
    seqNumber [3] INTEGER,
    euiccResult SEQUENCE OF EuiccResultData
}
EuiccResultData ::= CHOICE {
    enableResult [3] EnableProfileResult,
    disableResult [4] DisableProfileResult,
    deleteResult [5] DeleteProfileResult,
    listProfileInfoResult [45] ProfileInfoListResponse,
    getRATResult [6] RulesAuthorisationTable, -- see SGP.22
    configureImmediateEnableResult [7] ConfigureImmediateEnableResult,
    addEimResult [8] AddEimResult,
    deleteEimResult [9] DeleteEimResult,
    updateEimResult [10] UpdateEimResult,
    listEimResult [11] ListEimResult,
    rollbackResult [12] RollbackProfileResult,
    setFallbackAttributeResult [13] SetFallbackAttributeResult,
    unsetFallbackAttributeResult [14] UnsetFallbackAttributeResult,
    processingTerminated INTEGER {
        resultSizeOverflow(1),
        unknownOrDamagedCommand(2),
        interruption(3),
        undefinedError(127)
    }
}

```

```
    },
    setDefaultDpAddressResult [101] SetDefaultDpAddressResponse
}
EuiccPackageErrorSigned ::= SEQUENCE {
    euiccPackageErrorDataSigned EuiccPackageErrorDataSigned,
    euiccSignEPE [APPLICATION 55] OCTET STRING -- Tag '5F37'
}
EuiccPackageErrorDataSigned ::= SEQUENCE {
    eimId [0] UTF8String (SIZE(1..128)),
    counterValue [1] INTEGER,
    eimTransactionId [2] TransactionId OPTIONAL,
    euiccPackageErrorCode EuiccPackageErrorCode
}
EuiccPackageErrorCode ::= INTEGER {
    invalidEid(3),
    replayError(4),
    counterValueOutOfRange(6),
    sizeOverflow(15),
    ecallActive(104),
    undefinedError(127)
}
EuiccPackageUnsignedErrorCode ::= INTEGER {
    sizeOverflow(15),
    undefinedError(127)
}
EuiccPackageErrorUnsigned ::= SEQUENCE {
    eimId [0] UTF8String (SIZE(1..128)),
    eimTransactionId [2] TransactionId OPTIONAL,
    associationToken [4] INTEGER OPTIONAL,
    euiccPackageUnsignedErrorCode [15] EuiccPackageUnsignedErrorCode OPTIONAL
}
ConfigureImmediateEnableResult ::= INTEGER {
    ok(0),
    insufficientMemory(1),
    commandError(7),
    undefinedError(127)
}
EnableProfileResult ::= INTEGER {
    ok(0),
    iccidOrAidNotFound(1),
    profileNotInDisabledState(2),
    disallowedByPolicy(3),
    catBusy(5),
    rollbackNotAvailable(20),
    undefinedError(127)
}
DisableProfileResult ::= INTEGER {
    ok(0),
    iccidOrAidNotFound(1),
    profileNotInEnabledState(2),
    disallowedByPolicy(3),
    catBusy(5),
    undefinedError(127)
}
DeleteProfileResult ::= INTEGER {
    ok(0),
    iccidOrAidNotFound(1),
    profileNotInDisabledState(2),
    disallowedByPolicy(3),
    rollbackNotAvailable(20),
    returnFallbackProfile(21),
    undefinedError(127)
}
ProfileInfoListResponse ::= [45] CHOICE {
    profileInfoListOk SEQUENCE OF ProfileInfo, -- see SGP.22
    profileInfoListError ProfileInfoListError
}
```

```
}
ProfileInfoListError ::= INTEGER {
    incorrectInputValues(1),
    profileChangeOngoing(11),
    undefinedError(127)
}
RollbackProfileResult ::= INTEGER {
    ok(0),
    undefinedError(127)
}
SetFallbackAttributeResult ::= INTEGER {
    ok(0),
    iccidOrAidNotFound(1),
    fallbackNotAllowed(2),
    fallbackProfileEnabled(3),
    undefinedError(127)
}
UnsetFallbackAttributeResult ::= INTEGER {
    ok(0),
    noFallbackAttribute(2),
    fallbackProfileEnabled(3),
    commandError(7),
    undefinedError(127)
}
AddEimResult ::= CHOICE {
    associationToken [4] INTEGER,
    addEimResultCode INTEGER {
        ok(0),
        insufficientMemory(1),
        associatedEimAlreadyExists(2),
        ciPKUnknown(3),
        invalidAssociationToken(5),
        counterValueOutOfRange(6),
        commandError(7),
        undefinedError(127)
    }
}
DeleteEimResult ::= INTEGER {
    ok(0),
    eimNotFound(1),
    lastEimDeleted(2), -- no eIM Configuration Data available in eUICC,
    commandError(7),
    undefinedError(127)
}
UpdateEimResult ::= INTEGER {
    ok(0),
    eimNotFound(1),
    ciPKUnknown(3),
    counterValueOutOfRange(6),
    commandError(7),
    undefinedError(127)
}
ListEimResult ::= CHOICE {
    eimIdList SEQUENCE OF EimIdInfo,
    listEimError INTEGER {
        undefinedError(127)
    }
}
EimIdInfo ::= SEQUENCE {
    eimId [0] UTF8String (SIZE(1..128)),
    eimIdType [2] EimIdType OPTIONAL -- present in case of eimIdTypeOid and
    eimIdTypeFqdn
}
IpaEuiccDataErrorCode ::= INTEGER {
    incorrectTagList(1),
    euiccCiPKIdNotFound(5),
    ecallActive(104),
    undefinedError(127)
}
```

```
}

IpaEuiccDataResponseError ::= SEQUENCE {
    eimTransactionId [0] TransactionId OPTIONAL,
    ipaEuiccDataErrorCode IpaEuiccDataErrorCode
}

IpaEuiccDataResponse ::= [82] CHOICE { -- Tag 'BF52'
    ipaEuiccData IpaEuiccData,
    ipaEuiccDataResponseError IpaEuiccDataResponseError
}

PendingNotificationList ::= SEQUENCE OF PendingNotification
EuiccPackageResultList ::= SEQUENCE OF EuiccPackageResult

IpaEuiccData ::= SEQUENCE {
    notificationsList [0] PendingNotificationList OPTIONAL, -- Tag 'A0'
    defaultSmdpAddress [1] UTF8String OPTIONAL, -- Tag '81'
    euiccPackageResultList [2] EuiccPackageResultList OPTIONAL, -- Tag 'A2'
    euiccInfo1 [32] EUICCInfo1 OPTIONAL, -- Tag 'BF20'
    euiccInfo2 [34] EUICCInfo2 OPTIONAL, -- Tag 'BF22'
    rootSmdsAddress [3] UTF8String OPTIONAL, -- Tag '83'
    associationToken [4] INTEGER OPTIONAL, -- Tag '84'
    eumCertificate [5] Certificate OPTIONAL, -- Tag 'A5'
    euiccCertificate [6] Certificate OPTIONAL, -- Tag 'A6'
    eimTransactionId [7] TransactionId OPTIONAL, -- Tag '87'
    ipaCapabilities [8] IpaCapabilities OPTIONAL, -- Tag 'A8'
    deviceInfo [9] DeviceInfo OPTIONAL -- Tag 'A9'
}

ProfileDownloadTriggerResult ::= [84] SEQUENCE { -- tag 'BF54'
    eimTransactionId [2] TransactionId OPTIONAL,
    profileDownloadTriggerResultData CHOICE {
        profileInstallationResult [55] ProfileInstallationResult, -- see SGP.22 [4]
        profileDownloadError SEQUENCE {
            profileDownloadErrorReason [0] INTEGER {
                ecallActive (104),
                undefinedError(127)
            },
            errorResponse OCTET STRING OPTIONAL
        }
    }
}

ISDRProprietaryApplicationTemplateIoT ::= [PRIVATE 1] SEQUENCE { -- Tag 'E1'
    euiccConfiguration BIT STRING {
        ipaeSupported(0), -- IPA in the eUICC (IPaE) supported
        enabledProfile(1) -- eUICC contains an Enabled Profile
    }
}

IpaEActivationRequest ::= [66] SEQUENCE { -- Tag 'BF42'
    ipaeOption BIT STRING {
        activateIpaE(0) -- IPaE activation
    }
}

IpaEActivationResponse ::= [66] SEQUENCE { -- Tag 'BF42'
    ipaeActivationResult INTEGER {ok(0), notSupported(1)}
}

IpaCapabilities ::= SEQUENCE {
    ipaFeatures [0] BIT STRING {
        directRspServerCommunication (0),
        indirectRspServerCommunication (1),
        eimDownloadDataHandling (2),
        eimCtxParams1Generation (3),
        eimProfileMetadataVerification (4),
        minimizeEsipaBytes (5)
    },
    ipaSupportedProtocols [1] BIT STRING {
        ipaRetrieveHttps(0),
        ipaRetrieveCoaps(1),

```

```
        ipaInjectHttps(2),
        ipaInjectCoaps(3),
        ipaProprietary(4)
    } OPTIONAL,
    eSipaProprietaryProtocolInformation [2] VendorSpecificExtension OPTIONAL --
Additional information for proprietary protocol(s)
}
-- Definition of ProfileInfo
ProfileInfo ::= [PRIVATE 3] SEQUENCE { -- Tag 'E3'
    iccid Iccid OPTIONAL,
    isdpAid [APPLICATION 15] OctetTo16 OPTIONAL, -- AID of the ISD-P containing the
Profile, tag '4F'
    profileState [112] ProfileState OPTIONAL, -- Tag '9F70'
    profileNickname [16] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '90'
    serviceProviderName [17] UTF8String (SIZE(0..32)) OPTIONAL, -- Tag '91'
    profileName [18] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '92'
    iconType [19] IconType OPTIONAL, -- Tag '93'
    icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94', see condition in
ES10b:GetProfilesInfo
    profileClass [21] ProfileClass OPTIONAL, -- Tag '95'
    notificationConfigurationInfo [22] SEQUENCE OF
NotificationConfigurationInformation OPTIONAL, -- Tag 'B6'
    profileOwner [23] OperatorId OPTIONAL, -- Tag 'B7'
    dpProprietaryData [24] DpProprietaryData OPTIONAL, -- Tag 'B8'
    profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
    serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, -- Tag
'BF22'
    ecallIndication [123] BOOLEAN OPTIONAL, -- Tag '9F7B'
    fallbackAttribute [38] BOOLEAN DEFAULT FALSE, -- Tag '9F26', indicates a
Fallback Profile if set to TRUE
    fallbackAllowed [103] BOOLEAN OPTIONAL, -- Tag '9F67', indicates if Fallback is
authorized by the Profile Owner
    iotSpecificProfileInfo [100] SEQUENCE {
        -- Data objects and their tags
    } OPTIONAL -- Tag 'BF64' IoT specific Data object
}
UpdateMetadataRequest ::= [42] SEQUENCE { -- Tag 'BF2A'
    serviceProviderName [17] UTF8String (SIZE(0..32)) OPTIONAL, -- Tag '91'
    profileName [18] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '92'
    iconType [19] IconType OPTIONAL, -- Tag '93'
    icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94'
    profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
    serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, -- Tag
'BF22'
    fallbackAllowed [103] BOOLEAN OPTIONAL -- Tag '9F67'
}
StoreMetadataRequest ::= [37] SEQUENCE { -- Tag 'BF25'
    iccid Iccid,
    serviceProviderName [17] UTF8String (SIZE(0..32)), -- Tag '91'
    profileName [18] UTF8String (SIZE(0..64)), -- Tag '92' (corresponds to 'Short
Description' defined in SGP.21 [2])
    iconType [19] IconType OPTIONAL, -- Tag '93' (JPG or PNG)
    icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94' (Data of the icon.
Size 64 x 64 pixel. This field SHALL only be present if iconType is present)
    profileClass [21] ProfileClass DEFAULT operational, -- Tag '95'
    notificationConfigurationInfo [22] SEQUENCE OF
NotificationConfigurationInformation OPTIONAL,
    profileOwner [23] OperatorId OPTIONAL, -- Tag 'B7'
    profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
    serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, -- Tag
'BF22'
    serviceSpecificDataNotStoredInEuicc [35] VendorSpecificExtension OPTIONAL, --
Tag 'BF23'
    ecallIndication [123] BOOLEAN OPTIONAL, -- Tag '9F7B'
    fallbackAllowed [103] BOOLEAN OPTIONAL, -- Tag '9F67'
    iotSpecificMetadata [100] SEQUENCE {
        -- Data objects and their tags
    }
}
```

```

    } OPTIONAL -- Tag 'BF64' IoT specific Data object
}
AuthenticateClientRequest ::= [59] SEQUENCE { -- Tag 'BF3B'
    transactionId [0] TransactionId,
    authenticateServerResponse [56] AuthenticateServerResponse -- This is the
response from ES10b.AuthenticateServer
}
EUICCInfo2 ::= [34] SEQUENCE { -- Tag 'BF22'
    profileVersion [1] VersionType, -- Base eUICC Profile package version supported
svn [2] VersionType, -- GSMA SGP.22 version supported (SVN) referenced by SGP.32
    euiccFirmwareVer [3] VersionType, -- eUICC Firmware version
    extCardResource [4] OCTET STRING, -- Extended Card Resource Information
according to ETSI TS 102 226
    uiccCapability [5] UICCCapability,
    ts102241Version [6] VersionType OPTIONAL,
    globalplatformVersion [7] VersionType OPTIONAL,
    rspCapability [8] RspCapability,
    euiccCiPKIdListForVerification [9] SEQUENCE OF SubjectKeyIdentifier, -- List of
CI Public Key Identifiers supported on the eUICC for signature verification
    euiccCiPKIdListForSigning [10] SEQUENCE OF SubjectKeyIdentifier, -- List of CI
Public Key Identifier supported on the eUICC for signature creation
    euiccCategory [11] INTEGER {
        other(0),
        basicEuicc(1),
        mediumEuicc(2),
        contactlessEuicc(3)
    } OPTIONAL,
    forbiddenProfilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
    ppVersion VersionType, -- Protection Profile version
    sasAccreditationNumber UTF8String (SIZE(0..64)),
    certificationDataObject [12] CertificationDataObject OPTIONAL,
    treProperties [13] BIT STRING {
        isDiscrete(0),
        isIntegrated(1),
        usesRemoteMemory(2) -- refers to the usage of remote memory protected by the
Remote Memory Protection Function described in SGP.21 [4]
    } OPTIONAL,
    treProductReference [14] UTF8String OPTIONAL, -- Platform_Label as defined in
GlobalPlatform DLOA specification [57]
    additionalEuiccProfilePackageVersions [15] SEQUENCE OF VersionType OPTIONAL,
    ipaMode [16] IpaMode OPTIONAL, -- active IPA, mandatory within SGP.32
    euiccCiPKIdListForSigningV3 [17] SEQUENCE OF SubjectKeyIdentifier OPTIONAL, --
not used by this version of SGP.32.
    additionalEuiccInfo [18] OCTET STRING (SIZE(0..32)) OPTIONAL, -- not used by
this version of SGP.32
    highestSvn [19] VersionType OPTIONAL, -- not used by this version of SGP.32
    iotSpecificInfo [20] IoTSpecificInfo OPTIONAL, -- mandatory within SGP.32
    euiccMinimumSecurityLevel [21] OCTET STRING (SIZE (1)) OPTIONAL -- value for
eUICC Minimum Security Level Check
}

-- Definition of IoTSpecificInfo
IoTSpecificInfo ::= SEQUENCE {
    iotVersion [0] SEQUENCE OF VersionType, -- SGP.32 version(s) supported by the
eUICC, at least one must be present
    ecallSupported [1] NULL OPTIONAL, -- if present, eUICC supports the Emergency
Mechanism
    fallbackSupported [2] NULL OPTIONAL, -- if present, eUICC supports the Fallback
Mechanism
    fallbackAllowedUpdateSupported [3] NULL OPTIONAL -- if present, eUICC supports
the update of the Fallback Allowed metadata via ES6
}

-- Definition of IpaMode
IpaMode ::= INTEGER {
    ipad (0), -- IPAd is active
    ipae (1) -- IP Ae is active
}

```

```
AddInitialEimRequest ::= [87] SEQUENCE { -- Tag 'BF57'
  eimConfigurationDataList [0] SEQUENCE OF EimConfigurationData
}
AddInitialEimResponse ::= [87] CHOICE { -- Tag 'BF57'
  addInitialEimOk SEQUENCE OF CHOICE {
    associationToken [4] INTEGER,
    addOk NULL
  },
  addInitialEimError INTEGER {
    insufficientMemory(1),
    associatedEimAlreadyExists (2),
    ciPKUnknown(3),
    invalidAssociationToken(5),
    counterValueOutOfRange(6),
    commandError(7),
    undefinedError(127)
  }
}
EuiccMemoryResetRequest ::= [100] SEQUENCE { -- Tag 'BF64'
  resetOptions [2] BIT STRING {
    deleteOperationalProfiles(0),
    deleteFieldLoadedTestProfiles(1),
    resetDefaultSmdpAddress(2),
    deletePreLoadedTestProfiles(3),
    deleteProvisioningProfiles(4),
    resetEimConfigData(5),
    resetImmediateEnableConfig (6)
  }
}
EuiccMemoryResetResponse ::= [100] SEQUENCE { -- Tag 'BF64'
  resetResult INTEGER {ok(0), nothingToDelete(1), catBusy(5), ecallActive(104),
undefinedError(127)},
  resetEimResult INTEGER {ok(0), nothingToDelete(1), eimResetNotSupported(2),
undefinedError(127)} OPTIONAL,
  resetImmediateEnableConfigResult INTEGER {ok(0), resetIECNotSupported (1),
undefinedError(127)} OPTIONAL
}
GetCertsRequest ::= [86] SEQUENCE { -- Tag 'BF56'
  euiccCiPKId SubjectKeyIdentifier OPTIONAL -- CI Public Key Identifier supported
on the eUICC for signature creation
}
GetCertsResponse ::= [86] CHOICE { -- Tag 'BF56'
  certs SEQUENCE {
    eumCertificate [5] Certificate, -- Tag 'A5'
    euiccCertificate [6] Certificate -- Tag 'A6'
  },
  getCertsError INTEGER {invalidCiPKId(1), undfinedError(127)}
}
RetrieveNotificationsListRequest ::= [43] SEQUENCE { -- Tag 'BF2B'
  searchCriteria CHOICE {
    seqNumber [0] INTEGER,
    profileManagementOperation [1] NotificationEvent,
    euiccPackageResults [2] NULL
  } OPTIONAL
}
RetrieveNotificationsListResponse ::= [43] CHOICE { -- Tag 'BF2B'
  notificationList PendingNotificationList, -- Tag 'A0'
  notificationsListResultError INTEGER { undefinedError(127)}, -- Tag '81'
  euiccPackageResultList EuiccPackageResultList -- Tag 'A2'
}
ImmediateEnableRequest ::= [90] SEQUENCE { -- Tag 'BF5A'
  refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
ImmediateEnableResponse ::= [90] SEQUENCE { -- Tag 'BF5A'
  immediateEnableResult [0] INTEGER {
    ok(0),
    immediateEnableNotAvailable (1),
    noSessionContext(4),
```

```
        catBusy(5),
        undefinedError(127)
    }
}
ProfileRollbackRequest ::= [88] SEQUENCE { -- Tag 'BF58'
    refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
ProfileRollbackResponse ::= [88] SEQUENCE { -- Tag 'BF58'
    cmdResult INTEGER {
        ok(0),
        rollbackNotAllowed(1), -- Usage of rollback was not granted by the eIM
        catBusy(5),
        commandError(7),
        undefinedError(127)
    },
    eUICCPackageResult [81] EuiccPackageResult OPTIONAL
}
ConfigureImmediateProfileEnablingRequest ::= [89] SEQUENCE { -- Tag 'BF59'
    immediateEnableFlag [0] NULL OPTIONAL,
    defaultSmdpOid [1] OBJECT IDENTIFIER OPTIONAL,
    defaultSmdpAddress [2] UTF8String OPTIONAL
}
ConfigureImmediateProfileEnablingResponse ::= [89] SEQUENCE { -- Tag 'BF59'
    configImmediateEnableResult [0] INTEGER {
        ok(0),
        insufficientMemory(1),
        associatedEimAlreadyExists(2),
        undefinedError(127)
    }
}
GetEimConfigurationDataRequest ::= [85] SEQUENCE { -- Tag 'BF55'
    searchCriteria CHOICE {
        eimId [0] UTF8String (SIZE(1..128))
    } OPTIONAL
}
GetEimConfigurationDataResponse ::= [85] SEQUENCE { -- Tag 'BF55'
    eimConfigurationDataList [0] SEQUENCE OF EimConfigurationData
}
ExecuteFallbackMechanismRequest ::= [93] SEQUENCE { -- Tag 'BF5D'
    refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
ExecuteFallbackMechanismResponse ::= [93] SEQUENCE { -- Tag 'BF5D'
    executeFallbackMechanismResult [0] INTEGER {
        ok(0),
        profileNotInDisabledState(2),
        catBusy(5),
        fallbackNotAvailable(6), -- Fallback Attribute not set
        commandError(7),
        ecallActive(104),
        undefinedError(127)
    }
}
ReturnFromFallbackRequest ::= [94] SEQUENCE { -- Tag 'BF5E'
    refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
ReturnFromFallbackResponse ::= [94] SEQUENCE { -- Tag 'BF5E'
    returnFromFallbackResult [0] INTEGER {
        ok(0),
        catBusy(5),
        fallbackNotAvailable(6), -- Fallback Attribute not set
        commandError(7),
        undefinedError(127)
    }
}
EnableEmergencyProfileRequest ::= [91] SEQUENCE { -- Tag 'BF5B'
    refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
EnableEmergencyProfileResponse ::= [91] SEQUENCE { -- Tag 'BF5B'
```

```
enableEmergencyProfileResult [0] INTEGER {
    ok(0),
    profileNotInDisabledState(2),
    catBusy(5),
    ecallNotAvailable(8), -- Emergency Profile does not exist
    undefinedError(127)
}
}
DisableEmergencyProfileRequest ::= [92] SEQUENCE { -- Tag 'BF5C'
    refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
DisableEmergencyProfileResponse ::= [92] SEQUENCE { -- Tag 'BF5C'
    disableEmergencyProfileResult [0] INTEGER {
        ok(0),
        profileNotInEnabledState(2),
        catBusy(5),
        ecallNotAvailable(8), -- Emergency Profile does not exist or is not supported
        undefinedError(127)
    }
}
}
GetConnectivityParametersRequest ::= [95] SEQUENCE { -- Tag 'BF5F'
}
GetConnectivityParametersResponse ::= [95] CHOICE { -- Tag 'BF5F'
    connectivityParameters ConnectivityParameters,
    connectivityParametersError ConnectivityParametersError
}
}
ConnectivityParameters ::= SEQUENCE {
    httpParams [1] OCTET STRING OPTIONAL -- Also used for CoAP
}
}
ConnectivityParametersError ::= INTEGER {
    parametersNotAvailable(1),
    undefinedError(127)
}
}
SetDefaultDpAddressRequest ::= [101] SEQUENCE { -- Tag 'BF65'
    defaultDpAddress UTF8String -- Default SM-DP+ address as an FQDN
}
SetDefaultDpAddressResponse ::= [101] SEQUENCE { -- Tag 'BF65'
    setDefaultDpAddressResult INTEGER { ok (0), undefinedError (127) }
}
}
PrepareDownloadResponse ::= [33] CHOICE { -- Tag 'BF21'
    downloadResponseOk PrepareDownloadResponseOk,
    downloadResponseError PrepareDownloadResponseError,
    compactDownloadResponseOk CompactPrepareDownloadResponseOk
}
}
CompactPrepareDownloadResponseOk ::= SEQUENCE {
    compactEuiccSigned2 CompactEuiccSigned2, -- Compact version of EuiccSigned2
    euiccSignature2 [APPLICATION 55] OCTET STRING -- tag '5F37' signature on
    EuiccSigned2
}
}
CompactEuiccSigned2 ::= SEQUENCE {
    euiccOtpk [APPLICATION 73] OCTET STRING OPTIONAL, -- otPK.EUICC.ECKA, tag '5F49'
    euiccOtpk is always present except if bppEuiccOtpk was chosen by the eUICC
    hashCc Octet32 OPTIONAL -- Hash of confirmation code, if not received from eIM
}
}
EuiccSigned1 ::= SEQUENCE {
    transactionId [0] TransactionId,
    serverAddress [3] UTF8String,
    serverChallenge [4] Octet16, -- The RSP Server Challenge
    euiccInfo2 [34] EUICCInfo2,
    ctxParams1 CtxParams1
}
}
AuthenticateResponseOk ::= SEQUENCE {
    euiccSigned1 EuiccSigned1, -- Signed information
    euiccSignature1 [APPLICATION 55] OCTET STRING, --EUICC_Sign1, tag 5F37
    euiccCertificate Certificate, -- eUICC Certificate (CERT.EUICC.ECDSA) signed by
    the EUM
}
```

```

    eumCertificate Certificate -- EUM Certificate (CERT.EUM.ECDSA) signed by the
    requested CI
}
AuthenticateServerResponse ::= [56] CHOICE { -- Tag 'BF38'
    authenticateResponseOk AuthenticateResponseOk,
    authenticateResponseError AuthenticateResponseError,
    compactAuthenticateResponseOk CompactAuthenticateResponseOk
}
CompactAuthenticateResponseOk ::= SEQUENCE {
    signedData CHOICE {
        euiccSigned1 EuiccSigned1,
        compactEuiccSigned1 [0] CompactEuiccSigned1 -- Compact version of EuiccSigned1
    },
    euiccSignature1 [APPLICATION 55] OCTET STRING, -- tag 5F37 signature on
EuiccSigned1
    euiccCertificate [1] Certificate OPTIONAL, -- eUICC Certificate
(CERT.EUICC.ECDSA)
    eumCertificate [2] Certificate OPTIONAL -- EUM Certificate (CERT.EUM.ECDSA)
}
CompactEuiccSigned1 ::= SEQUENCE {
    extCardResource [4] OCTET STRING, -- Extended Card Resource Information
according to ETSI TS 102 226 extracted from euiccInfo2,
    ctxParams1 [2] CtxParams1 OPTIONAL -- ctxParams1 may be left out by IPA if eIM
ctxParams1 was received from the eIM
}
PendingNotification ::= CHOICE {
    profileInstallationResult [55] ProfileInstallationResult, -- tag 'BF37'
    otherSignedNotification OtherSignedNotification,
    compactProfileInstallationResult [0] CompactProfileInstallationResult,
    compactOtherSignedNotification [1] CompactOtherSignedNotification
}
ProfileInstallationResult ::= [55] SEQUENCE { -- Tag 'BF37'
    profileInstallationResultData [39] ProfileInstallationResultData,
    euiccSignPIR EuiccSignPIR
}
CompactProfileInstallationResult ::= SEQUENCE {
    compactProfileInstallationResultData [0] CompactProfileInstallationResultData,
    euiccSignPIR EuiccSignPIR
}
CompactProfileInstallationResultData ::= SEQUENCE {
    transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
seqNumber INTEGER,
    iccidPresent BOOLEAN DEFAULT TRUE,
    compactFinalResult [2] CHOICE {
        compactSuccessResult CompactSuccessResult,
        errorResult ErrorResult
    }
}
CompactSuccessResult ::= SEQUENCE {
    compactAid [APPLICATION 15] OCTET STRING (SIZE (2)), -- Byte 14 and 15 of ISD-P
AID
    simaResponse OCTET STRING OPTIONAL -- MUST be present if the simaResponse value
(EUICCResponse) is different from the 9-byte value '30 07 A0 05 30 03 80 01 00'
representing success
}
CompactOtherSignedNotification ::= SEQUENCE {
    eidValue [APPLICATION 26] Octet16 OPTIONAL, -- Tag '5A'
    tbsOtherNotification NotificationMetadata,
    euiccNotificationSignature [APPLICATION 55] OCTET STRING, -- eUICC signature of
tbsOtherNotification, Tag '5F37'
    euiccCiPKIdentifierToBeUsed OCTET STRING OPTIONAL -- CI Public Key Identifier
(possibly truncated) that was used by the eUICC to sign OtherSignedNotification
}
CancelSessionResponse ::= [65] CHOICE { -- Tag 'BF41'
    cancelSessionResponseOk CancelSessionResponseOk,
    cancelSessionResponseError INTEGER {invalidTransactionId(5),
undefinedError(127)},
    compactCancelSessionResponseOk CompactCancelSessionResponseOk
}

```

```

}
CompactCancelSessionResponseOk ::= SEQUENCE {
    compactEuiccCancelSessionSigned CompactEuiccCancelSessionSigned, -- Compact
    version of euiccCancelSessionSigned
    euiccCancelSessionSignature [APPLICATION 55] OCTET STRING -- tag 5F37 signature
    on euiccCancelSessionSigned
}
CompactEuiccCancelSessionSigned ::= SEQUENCE {
    reason CancelSessionReason OPTIONAL
}
EsipaMessageFromIpaToEim ::= CHOICE {
    initiateAuthenticationRequestEsipa [57] InitiateAuthenticationRequestEsipa, --
    Tag 'BF39'
    authenticateClientRequestEsipa [59] AuthenticateClientRequestEsipa, -- Tag
    'BF3B'
    getBoundProfilePackageRequestEsipa [58] GetBoundProfilePackageRequestEsipa, --
    Tag 'BF3A'
    cancelSessionRequestEsipa [65] CancelSessionRequestEsipa, -- Tag 'BF41'
    handleNotificationEsipa [61] HandleNotificationEsipa, -- Tag 'BF3D'
    transferEimPackageResponse [78] TransferEimPackageResponse, -- Tag 'BF4E'
    getEimPackageRequest [79] GetEimPackageRequest, -- Tag 'BF4F'
    provideEimPackageResult [80] ProvideEimPackageResult -- Tag 'BF50'
}
EsipaMessageFromEimToIpa ::= CHOICE {
    initiateAuthenticationResponseEsipa [57] InitiateAuthenticationResponseEsipa, --
    Tag 'BF39'
    authenticateClientResponseEsipa [59] AuthenticateClientResponseEsipa, -- Tag
    'BF3B'
    getBoundProfilePackageResponseEsipa [58] GetBoundProfilePackageResponseEsipa, --
    Tag 'BF3A'
    cancelSessionResponseEsipa [65] CancelSessionResponseEsipa, -- Tag 'BF41'
    transferEimPackageRequest [78] TransferEimPackageRequest, -- Tag 'BF4E'
    getEimPackageResponse [79] GetEimPackageResponse, -- Tag 'BF4F'
    provideEimPackageResultResponse [80] ProvideEimPackageResultResponse -- Tag
    'BF50'
}
InitiateAuthenticationRequestEsipa ::= [57] SEQUENCE { -- Tag 'BF39'
    euiccChallenge [1] Octet16, -- random eUICC challenge
    smdpAddress [3] UTF8String OPTIONAL,
    euiccInfo1 EUICCIInfo1 OPTIONAL,
    eimTransactionId [2] TransactionId OPTIONAL
}
InitiateAuthenticationResponseEsipa ::= [57] CHOICE { -- Tag 'BF39'
    initiateAuthenticationOkEsipa InitiateAuthenticationOkEsipa,
    initiateAuthenticationErrorEsipa INTEGER {
        invalidDpAddress(1),
        euiccVersionNotSupportedByDp(2),
        ciPKIdNotSupported(3),
        smdpAddressMismatch(50),
        smdpOidMismatch(51),
        invalidEimTransactionId(52),
        undefinedError(127)
    }
}
InitiateAuthenticationOkEsipa ::= SEQUENCE {
    transactionId [0] TransactionId OPTIONAL, -- The TransactionID generated by the
    SM-DP+
    serverSigned1 ServerSigned1, -- Signed information
    serverSignature1 [APPLICATION 55] OCTET STRING, -- Server Sign1, Tag '5F37'
    euiccCiPKIdentifierToBeUsed OCTET STRING, -- Key identifier (possibly truncated)
    of the CI Public Key to be used as required by ES10b.AuthenticateServer
    serverCertificate Certificate,
    matchingId UTF8String OPTIONAL,
    ctxParams1 [2] CtxParams1 OPTIONAL
}
AuthenticateClientRequestEsipa ::= [59] SEQUENCE { -- Tag 'BF3B'

```

```

    transactionId [0] TransactionId, -- The TransactionID generated by the SM-
DP+/SM-DS
    authenticateServerResponse [56] AuthenticateServerResponse -- This is the
response from ES10b.AuthenticateServer, possibly in compact format
}
AuthenticateClientResponseEsipa ::= [59] CHOICE { -- Tag 'BF3B'
    authenticateClientOkDPESipa AuthenticateClientOkDPESipa,
    authenticateClientOkDSEsipa AuthenticateClientOkDSEsipa,
    authenticateClientErrorEsipa INTEGER {
        eumCertificateInvalid(1),
        eumCertificateExpired(2),
        euiccCertificateInvalid(3),
        euiccCertificateExpired(4),
        euiccSignatureInvalid(5),
        matchingIdRefused(6),
        eidMismatch(7),
        noEligibleProfile(8),
        ciPKUnknown(9),
        invalidTransactionId(10),
        insufficientMemory(11), -- Note: values 12-17 are reserved
        downloadOrderExpired(18),
        pprNotAllowed(50),
        eventIdUnknown(56),
        undefinedError(127)
    }
}
AuthenticateClientOkDPESipa ::= SEQUENCE {
    transactionId [0] TransactionId OPTIONAL, -- The TransactionID generated by the
SM-DP+
    profileMetaData [37] StoreMetadataRequest OPTIONAL,
    smdpSigned2 SmdpSigned2, -- Signed information
    smdpSignature2 [APPLICATION 55] OCTET STRING, -- Tag '5F37'
    smdpCertificate Certificate, -- CERT.DPpb.ECDSA
    hashCc Octet32 OPTIONAL -- Hash of confirmation code
}
AuthenticateClientOkDSEsipa ::= SEQUENCE {
    transactionId [0] TransactionId, -- The TransactionID generated by the SM-DS
    profileDownloadTrigger [84] ProfileDownloadTriggerRequest OPTIONAL -- Tag 'BF54'
}
GetBoundProfilePackageRequestEsipa ::= [58] SEQUENCE { -- Tag 'BF3A'
    transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
    prepareDownloadResponse [33] PrepareDownloadResponse -- This is the response
from ES10b.PrepareDownload, possibly in compact format
}
GetBoundProfilePackageResponseEsipa ::= [58] CHOICE { -- Tag 'BF3A'
    getBoundProfilePackageOkEsipa GetBoundProfilePackageOkEsipa,
    getBoundProfilePackageErrorEsipa INTEGER {
        euiccSignatureInvalid(1),
        confirmationCodeMissing(2),
        confirmationCodeRefused(3),
        confirmationCodeRetriesExceeded(4),
        bppRebindingRefused(5),
        deprecated(6), -- this value is no longer used
        metadataMismatch(50),
        invalidTransactionId(95),
        undefinedError(127)
    }
}
GetBoundProfilePackageOkEsipa ::= SEQUENCE {
    transactionId [0] TransactionId OPTIONAL, -- The TransactionID generated by the
SM-DP+
    boundProfilePackage [54] BoundProfilePackage
}
HandleNotificationEsipa ::= [61] CHOICE { -- Tag 'BF3D'
    pendingNotification [0] PendingNotification, -- A Notification to be delivered
to a Notification Receiver, possibly in compact format
    provideEimPackageResult [80] ProvideEimPackageResult -- Tag 'BF50'
}

```

```

CancelSessionRequestEsipa ::= [65] SEQUENCE { -- Tag 'BF41'
    transactionId TransactionId, -- The TransactionID generated by the SM-DP+
    cancelSessionResponse CancelSessionResponse -- This is the response from ES10b.
    CancelSession function, possibly in compact format
}
CancelSessionResponseEsipa ::= [65] CHOICE { -- Tag 'BF41'
    cancelSessionOk CancelSessionOk,
    cancelSessionError INTEGER {
        invalidTransactionId(1),
        euiccSignatureInvalid(2),
        undefinedError(127)
    }
}
CancelSessionOk ::= SEQUENCE { -- This function has no output data
}
StateChangeCause ::= INTEGER {
    otherEim(0), -- for change(s) requested by another registered eIM
    fallback(1), -- for change(s) following fallback requested by IPA
    emergencyProfile(2), -- for change(s) following Emergency Profile swap requested
    by IPA
    local(3), -- for change(s) applied locally (RFU, not supported in this version)
    reset(4), -- for change(s) following an euicc reset
    immediateEnableProfile(5), -- for change(s) following immediate Profile enabling
    requested by IPA
    deviceChange(6), -- for change(s) detected by IPaE following a IoT Device change
    (removable eSIMs)
    undefined(127) -- unknown cause
}
GetEimPackageRequest ::= [79] SEQUENCE { -- Tag 'BF4F'
    eidValue [APPLICATION 26] Octet16, -- Tag '5A'
    notifyStateChange [0] NULL OPTIONAL, -- Notification to the eIM that it should
    update its information about the eUICC (e.g. list of Profiles, Profile states...)
    stateChangeCause [1] StateChangeCause OPTIONAL, -- Provide the cause of the
    notified state change(s)
    rPLMN [2] OCTET STRING (SIZE(3)) OPTIONAL -- MCC and MNC of the last registered
    PLMN, coded as defined in 3GPP TS 24.008 [22]
}
GetEimPackageResponse ::= [79] CHOICE { -- Tag 'BF4F'
    euiccPackageRequest [81] EuiccPackageRequest, -- Tag 'BF51'
    ipaEuiccDataRequest [82] IpaEuiccDataRequest, -- Tag 'BF52'
    profileDownloadTriggerRequest [84] ProfileDownloadTriggerRequest, -- Tag 'BF54'
    eimPackageError INTEGER {
        noEimPackageAvailable(1),
        eidNotFound(2),
        invalidEid(3),
        missingEid(4),
        undefinedError(127)
    }
}
EimPackageResultErrorCode ::= INTEGER {
    invalidPackageFormat(1),
    unknownPackage(2),
    undefinedError(127)
}
EimPackageResultResponseError ::= SEQUENCE {
    eimTransactionId [0] TransactionId OPTIONAL,
    eimPackageResultErrorCode EimPackageResultErrorCode
}
EimPackageResult ::= CHOICE {
    euiccPackageResult [81] EuiccPackageResult, -- Tag 'BF51'
    ePRAndNotifications SEQUENCE {
        euiccPackageResult [81] EuiccPackageResult, -- Tag 'BF51'
        notificationList [0] PendingNotificationList -- Tag 'A0'
    },
    ipaEuiccDataResponse [82] IpaEuiccDataResponse, -- Tag 'BF52'
}
    
```

```

profileDownloadTriggerResult [84] ProfileDownloadTriggerResult, -- Tag 'BF54'
eimPackageResultResponseError [0] EimPackageResultResponseError
}

ProvideEimPackageResult ::= [80] SEQUENCE { -- Tag 'BF50'
    eidValue [APPLICATION 26] Octet16 OPTIONAL, -- Tag '5A'
    eimPackageResult EimPackageResult
}

ProvideEimPackageResultResponse ::= [80] CHOICE { -- Tag 'BF50'
    eimAcknowledgements [83] EimAcknowledgements, -- Tag 'BF53'
    emptyResponse SEQUENCE {}, -- because EimAcknowledgements was OPTIONAL
    provideEimPackageResultError INTEGER {
        eidNotFound(2),
        invalidEid(3),
        missingEid(4),
        undefinedError(127)
    }
}

TransferEimPackageRequest ::= [78] CHOICE { -- Tag 'BF4E'
    euiccPackageRequest [81] EuiccPackageRequest, -- Tag 'BF51'
    ipaEuiccDataRequest [82] IpaEuiccDataRequest, -- Tag 'BF52'
    eimAcknowledgements [83] EimAcknowledgements, -- Tag 'BF53'
    profileDownloadTriggerRequest [84] ProfileDownloadTriggerRequest -- Tag 'BF54'
}

TransferEimPackageResponse ::= [78] CHOICE { -- Tag 'BF4E'
    euiccPackageResult [81] EuiccPackageResult, -- Tag 'BF51'
    ePRAndNotifications SEQUENCE {
        euiccPackageResult [81] EuiccPackageResult, -- Tag 'BF51'
        notificationList [0] PendingNotificationList -- Tag 'A0'
    },
    ipaEuiccDataResponse [82] IpaEuiccDataResponse, -- Tag 'BF52'
    eimPackageReceived NULL,
    eimPackageReceivedWithCid [96] EimPackageReceivedWithCid,
    eimPackageError INTEGER {
        invalidPackageFormat(1),
        unknownPackage(2),
        undefinedError(127)
    },
    eimPackageErrorWithCid [97] EimPackageErrorWithCid
}

EimPackageReceivedWithCid ::= SEQUENCE {
    correlationId CHOICE {
        eimTransactionId [0] TransactionId, -- if received in the EimPackage
        eidValue [APPLICATION 26] Octet16 -- otherwise
    } OPTIONAL
}

EimPackageErrorWithCid ::= SEQUENCE {
    correlationId CHOICE {
        eimTransactionId [0] TransactionId, -- if received in the EimPackage
        eidValue [APPLICATION 26] Octet16 -- otherwise
    } OPTIONAL,
    eimPackageError EimPackageResultErrorCode
}
END
    
```

## Annex D Tag allocation (Normative)

This annex lists the tags allocated to data objects that SHALL be used for the definition of the eUICC functions.

Tag	Data name	Introduced in
'BF20'	EUICCInfo1	SGP.22

Tag	Data name	Introduced in
'BF21'	PrepareDownloadRequest or PrepareDownloadResponse	SGP.22
'BF22'	EUICCInfo2	SGP.22
'BF25'	StoreMetadataRequest	SGP.22
'BF2B'	RetrieveNotificationsListRequest or RetrieveNotificationsListResponse	SGP.22
'BF2D'	ProfileInfoListRequest or ProfileInfoListResponse	SGP.22
'BF2E'	GetEuiccChallengeRequest or GetEuiccChallengeResponse	SGP.22
'BF2F'	NotificationMetadata	SGP.22
'BF30'	NotificationSentRequest or NotificationSentResponse	SGP.22
'BF36'	BoundProfilePackage	SGP.22
'BF37'	ProfileInstallationResult	SGP.22
'BF38'	AuthenticateServerRequest or AuthenticateServerResponse	SGP.22
'BF39'	InitiateAuthenticationRequest or InitiateAuthenticationResponse ( <i>InitiateAuthenticationRequestEsipa or InitiateAuthenticationResponseEsipa</i> )	SGP.22 (SGP.32)
'BF3A'	GetBoundProfilePackageRequest or GetBoundProfilePackageResponse ( <i>GetBoundProfilePackageRequestEsipa or GetBoundProfilePackageResponseEsipa</i> )	SGP.22 (SGP.32)
'BF3B'	AuthenticateClientRequest or AuthenticateClientResponse ( <i>AuthenticateClientRequestEsipa or AuthenticateClientResponseEsipa</i> )	SGP.22 (SGP.32)
'BF3C'	EuiccConfiguredAddressesRequest or EuiccConfiguredAddressesResponse	SGP.22
'BF3D'	HandleNotification ( <i>HandleNotificationEsipa</i> )	SGP.22 (SGP.32)
'BF3E'	GetEuiccDataRequest or GetEuiccDataResponse	SGP.22
'BF41'	CancelSessionRequest or CancelSessionResponse or CancelSessionRequestEs9 or CancelSessionResponseEs9 ( <i>CancelSessionRequestEsipa or CancelSessionResponseEsipa</i> )	SGP.22 (SGP.32)
'BF42'	IpaActivationRequest or IpaActivationResponse	SGP.32
'BF43'	GetRatRequest or GetRatResponse	SGP.22
'BF4E'	TransferEimPackageRequest or TransferEimPackageResponse	SGP.32
'BF4F'	GetEimPackageRequest or GetEimPackageResponse	SGP.32
'BF50'	ProvideEimPackageResult or ProvideEimPackageResponse	SGP.32
'BF51'	EuiccPackageRequest or EuiccPackageResult	SGP.32
'BF52'	IpaEuiccDataRequest or IpaEuiccDataResponse	SGP.32
'BF53'	EimAcknowledgements	SGP.32
'BF54'	ProfileDownloadTriggerRequest or ProfileDownloadTriggerResponse	SGP.32
'BF55'	GetEimConfigurationDataRequest or GetEimConfigurationDataResponse	SGP.32
'BF56'	GetCertsRequest or GetCertsResponse	SGP.32
'BF57'	AddInitialEimRequest or AddInitialEimResponse	SGP.32
'BF58'	ProfileRollbackRequest or ProfileRollbackResponse	SGP.32

Tag	Data name	Introduced in
'BF59'	ConfigureImmediateProfileEnablingRequest or ConfigureImmediateProfileEnablingResponse	SGP.32
'BF5A'	ImmediateEnableRequest or ImmediateEnableResponse	SGP.32
'BF5B'	EnableEmergencyProfileRequest or EnableEmergencyProfileResponse	SGP.32
'BF5C'	DisableEmergencyProfileRequest or DisableEmergencyProfileResponse	SGP.32
'BF5D'	ExecuteFallbackMechanismRequest or ExecuteFallbackMechanismResponse	SGP.32
'BF5E'	ReturnFromFallbackRequest or ReturnFromFallbackResponse	SGP.32
'BF5F'	GetConnectivityParametersRequest or GetConnectivityParametersResponse	SGP.32
'BF60' to 'BF63'	Reserved	
'BF64'	EuiccMemoryResetRequest or EuiccMemoryResetResponse	SGP.32
'BF65'	SetDefaultDpAddressRequest or SetDefaultDpAddressResponse	SGP.32
'9F26'	fallbackAttribute	SGP.32
'BF64'	iotSpecificMetadata	SGP.32
'9F67'	fallbackAllowed	SGP.32
'9F7B'	ecallIndication	SGP.32

**Table 36 Tag Allocation**

## Annex E Document Management (Informative)

### E.1 Document History

Version	Date	Change Request	Approval Authority	Editor / Company
V1.0	26 May 2023	<ul style="list-style-type: none"> <li>• CR001R01 Overview</li> <li>• CR002R01 Scope</li> <li>• CR003R01 Document Purpose</li> <li>• CR004R01 Intended Audience</li> <li>• CR005R01 Conventions</li> <li>• CR006R01 General Architecture</li> <li>• CR007R00 Roles</li> <li>• CR008R01 Interfaces</li> <li>• CR0012R01 eUICC Architecture diagram</li> <li>• CR0009R05 eUICC Architecture</li> <li>• CR0010R03 Security Overview</li> <li>• CR0014R00 ESIMWG7_52_AP2 - LPA within SGP.22 vs IPA within SGP.32</li> <li>• CR0017R02 Introduction of procedures</li> <li>• CR0018R01 eUICC Initialisation</li> <li>• CR0024R04 ESipa procedure (1/2): eIM Command Retrieval</li> <li>• CR0025R03 ESipa procedure (2/2): eIM Command Injection</li> <li>• CR0015R09 Direct Profile Download</li> <li>• CR0026R01 Add PSMO generic description</li> <li>• CR0030R01 Implementation of ESipa interface</li> <li>• CR0031R00 Communication between EIM and IPA</li> <li>• CR0034R02 Add definition of PSMO</li> </ul>	ISAG	Gloria Trujillo, GSMA (Editor)

V1.1	26 April 2024	<ul style="list-style-type: none"> <li>• CR0248R00 Correction of Typo in Section 3.2.3.1 "Direct Profile Download"</li> <li>• CR0249R00 List of status codes for ESipa.GetEimPackage</li> <li>• CR0251R01 Editorial modification to mapping of procedures to LwM2M</li> <li>• CR0252R00 Editorial change of Tag number for IpaCapabilities in IpaEuiccDat</li> <li>• CR0253R02 Correction to correlate package requests and results</li> <li>• CR0254R01 Editorial change to add Annex ASN.1 Definitions (Normative)</li> <li>• CR0255R01 Editorial change for ASN-1 imports</li> <li>• CR0256R01 Clarify ES10 CancelSession</li> <li>• CR0258R01 Correlate InitiateAuthentication with ProfileDownloadTriggerRequest</li> <li>• CR0260R00 Various minor editorials</li> <li>• CR0261R00 Fix provideEimPackageResult tag number in ESipa.HandleNotification</li> <li>• CR0262R01 Editorial changes to rename the functions &amp; procedures related to "Automatic Profile Enabling"</li> <li>• CR0263R03 Correction of Enable Profile PSMO command</li> <li>• CR0264R01 Correction of Delete Profile PSMO command</li> <li>• CR0265R00 Editorial change Missing ASN1START and ASN1STOP tags</li> <li>• CR0266R00 Editorial change Add Missing tag number in ESipa.HandleNotification</li> <li>• CR0268R01 Clarification on Update Eim</li> <li>• CR0269R01 Correction to eIM ID</li> <li>• CR0270R01 Correction to Generic eUICC Package download procedure</li> <li>• CR0274R01 Keep RetrieveNotificationsList function with empty argument compatible with SGP.22</li> <li>• CR0275R00 Editorial fixing Add Eim Transaction ID in ASN.1 binding of Esipa.InitiateAuthentication</li> <li>• CR0277R01 Add eIM Transaction ID in ASN.1 binding of ESipa.InitiateAuthentication</li> <li>• CR0273R01 Referencing EUICCInfo2 from SGP32</li> <li>• CR0279R00 Fix diagram to reflect changes agreed in CR269R01</li> <li>• CR0280R00 Add missing error field in InitiateAuthenticationErrorEsipa</li> <li>• CR0281R01 Fix ESipa.HandleNotification function description for IndirectProfileDownload</li> </ul>	ISAG	Gloria Trujillo, GSMA (Editor)
------	---------------	---	------	--------------------------------

		<ul style="list-style-type: none"> <li>• CR0283R00 Add missing parameters to GetEimConfiguration</li> <li>• CR0284R00 Error code for bad correlation of eimTransactionId after ProfileDownloadTriggerRequest</li> <li>• CR0287R00 Clarification about default SM-DP+ parameters for Immediate Profile Enabling procedures</li> <li>• CR0288R00 eUICCMemoryReset response</li> <li>• CR0290R01 Clarify behaviour of IP Ae in Immediate Profile Enabling</li> <li>• CR0240R04 Introduction of Profile Fallback Mechanism</li> <li>• CR0271R03 Correction of eUICC Memory Reset function</li> <li>• CR0272R03 Completion of eCall Mechanism</li> <li>• CR0285R03 Fix inconsistency in updateEim description</li> <li>• CR0289R01 eIM TLS/DTLS Certificate (CERT.EIM.TLS)</li> <li>• CR0276R03 Clarify the need of connectivity of parameters for constrained devices</li> <li>• CR0294R03 Add setDefaultDpAddress function</li> <li>• CR0296R01 Fix error code inconsistencies for some eCO commands</li> <li>• CR0297R00 Editorial change to fix HandleNotifications typo</li> <li>• CR0298R00 Clarifying empty response of ES10b.GetEimConfigurationData</li> <li>• CR0300R01 eIM TLS/DTLS Certificate format – update</li> <li>• CR0302R03 Add command data to emergency call function</li> <li>• CR0303R01 Editorial corrections to Fallback</li> <li>• CR0278R02 optimisation of EuiccPackageResult</li> <li>• CR0291R00 Field-Test eUICC Clarification</li> <li>• CR0295R00 Fix error code inconsistencies for some PSMO commands</li> <li>• CR0312R00 Editorial changes ASN_1 definition fixes</li> <li>• CR0314R01 Clarify triggering of Indirect Profile Download</li> <li>• CR0257R09 IP Ae configuration</li> <li>• CR0292R04 EuiccMemoryReset and Provisioning Profiles</li> <li>• CR0301R01 Correction to MQTT Broker address</li> <li>• CR0304R02 Rollback procedure update</li> </ul>		
--	--	---	--	--

		<ul style="list-style-type: none"> <li>• CR0305R01 Align enable and disable PSMO with SGP.22</li> <li>• CR0306R01 ES10b.getEimConfigurationData update</li> <li>• CR0310R01 Clarification of the use of the IoTSpecificInfo data object</li> <li>• CR0313R01 Support of SNI</li> <li>• CR0315R00 Update of references</li> <li>• CR0316R00 Correcting error in eUICC Memory Reset</li> <li>• CR0317R01 No space in eUICC with deleted Notifications</li> <li>• CR0320R01 Add Tag Allocation Annex</li> <li>• CR0321R01 Editorial changes for some SGP 32 tags</li> <li>• CR0323R01 Clarify TransactionID in PIR and HandleNotification addresses</li> <li>• CR0325R01 Clarifications to Emergency Profile handling</li> <li>• CR0326R03 Miscellaneous corrections to fallback mechanism</li> <li>• CR0327R01 Resetting fallback authorization upon ecall</li> <li>• CR0328R00 Editorial changes to fix ASN-1 compilation issues</li> <li>• CR0293R07 Support of JSON and ASN.1 bindings by Eim</li> <li>• CR0308R04 Support of JSON and ASN.1 bindings by SM-DP+ and SM-DS</li> <li>• CR0307R02 ESipa.ProvideEimPackageResult request update</li> <li>• CR0309R00 Clarification of Delete Profile procedure and Provisioning profiles</li> <li>• CR0318R02 Clarification on generating Notifications</li> <li>• CR0319R01 Replace sessionAborted reason code</li> <li>• CR0322R01 Adding EID to ESipa.ProvideEimPackageResult</li> <li>• CR0329R00 Add reference RFC 6066</li> <li>• CR0330R00 Editorial changes to ES9+ HandleNotification</li> <li>• CR0331R00 Clarification to Indirect Profile Download and Instalation procedure</li> <li>• CR0333R02 Extend use of default SM-DP+ to Indirect Profile Download</li> <li>• CR0334R04 Clarifications on Emergency Mechanism</li> <li>• CR0335R01 Clarify that fallback functions are optional to support</li> <li>• CR0338R01 Clarification on Enable and Disable PSMO procedure</li> <li>• CR0339R00 Editorial changes to fix ASN-1 compilation issues</li> </ul>		
--	--	--	--	--

		<ul style="list-style-type: none"> <li>• CR0340R01 Introducing a flag for Fallback Support into eUICCInfo2</li> <li>• CR0341R01 SNI section updates</li> <li>• CR0343R00 Correcting behavior in ES10b.AuthenticateServer</li> <li>• CR0344R00 GetEimPackage update</li> <li>• CR0345R00 TLS certificate retrieval for IAd in constrained IoT Devices</li> <li>• CR0346R00 Rollback notifications</li> <li>• CR0324R07 Revision of Annex B.2</li> <li>• CR0332R03 Clarify the conditions of using default SM-DP+</li> <li>• CR0348R01 List Profile default tag list</li> <li>• CR0350R01 Editorial changes to align CR0338 and CR0334</li> <li>• CR0351R01 Clarify Note 3 in ESipa.InitiateAuthentication function</li> <li>• CR0353R01 Clarify how the eIM verify Profile metadata</li> <li>• CR0352R02 Rename euiccCiPKIdToBeUsed to avoid incompatibility with with SGP.22</li> <li>• CR0354R00 Fix Setting the Fallback Attribute</li> <li>• CR0355R02 Fix ExecuteFallbackMechanism function</li> <li>• CR0356R00 Fix EuiccPackageResult errors in case of eCall</li> <li>• CR0357R00 Editorial correction section 2.6.5</li> <li>• CR0358R00 Clarification of REFRESH handling in Enable and Disable Emergency Profile</li> <li>• CR0359R01 Corrections to ASN.1</li> <li>• CR0360R00 Editorial changes to fix ASN-1 compilation issues</li> <li>• CR0361R01 Correction to terminal capability - option 1</li> <li>• CR0363R01 Correction to eIM ID</li> <li>• CR0364R01 Correction to notification retrieval if searchCriteria is NULL</li> <li>• CR0365R00 Clarification of Fallback procedure</li> <li>• CR0366R02 Use of REFRESH in Enable and Disable Emergency Profile</li> <li>• CR0367R01 Clarification on character set for JSON binding</li> <li>• CR0369R01 Editorial fix EuiccPackageResult errors in case of eCall</li> <li>• CR0368R01 Correction to ES10b.PrepareDownload function handling</li> <li>• CR0370R01 Clean-up ESipa.AuthenticateClient function description</li> </ul>		
--	--	---	--	--

		<ul style="list-style-type: none"> <li>• CR0371R00 Editorial Correction to Eim</li> <li>• CR0373R00 Editorial Correction to align capitalisation of eimId</li> <li>• CR0374R01 Editorial change to Annex B.2 MQTT on topic usage</li> <li>• CR0375R00 Editorial Correction to eimId formatting</li> <li>• CR0376R02 Change to correct ASN.1 for IpaEuiccDataRequest &amp; Response</li> <li>• CR0377R00 Yet another correction to TERMINAL CAPABILITY</li> <li>• CR0378R01 Reworking the tag numbers of ES10b.EuiccMemoryReset</li> <li>• CR0379R00 Editorial to correct use of eimPackageResponse</li> <li>• CR0381R01 Reworking the tag number of ES10b.SetDefaultDpAddress function</li> <li>• CR0382R01 Correction of ASN.1 TAGs</li> <li>• CR0383R01 Add ecallActive error code to euiccPackageResult</li> <li>• CR0384R00 Editorial correction to reason metadataMismatch</li> <li>• CR0385R00 Add Event &amp; EventID as per .22 in terms as used but undefined</li> <li>• CR0386R00 Add MatchingID as per .22 into Terms as used but undefined</li> <li>• CR0387R00 Clean-up of Definition and Abbreviation sections</li> <li>• CR0388R00 Add Event Retrieval definition</li> <li>• CR0390R01 Correction to Profile Download Option (b)</li> <li>• CR0392R00 Add Activation Code Token to Terms as used but undefined</li> <li>• CR0393R01 Remove NOTE from GetEimPackage section 5.14.5</li> <li>• CR0395R00 Correction to storemetadata request in relation to fallback</li> <li>• CR0396R00 Editorial correction for GetProfilesInfo</li> <li>• CR0397R00 Editorial changes for ES10b Enable and DisableEmergencyProfile functions</li> <li>• CR0399R00 Complete section 5.2.1 Common Data Types</li> <li>• CR0400R00 Editorial to make Profile Rollback Procedure a new section</li> <li>• CR0402R01 Add ecallActive for IPA EuiccDataRequest and ProfileDownloadTriggerRequest</li> <li>• CR0403R01 Add indirect Profile Download description to the eIM Configuration data</li> <li>• CR0404R00 Disable profile error conditions simplification</li> <li>• CR0405R01 Editorial changes for 'Profile MetaData' section</li> </ul>		
--	--	--	--	--

		<ul style="list-style-type: none"> <li>• CR0408R01 Editorial correction for ESipa.GetEimPackage function</li> <li>• CR0409R00 Alignment of error codes for Enable/Disable/Delete Profile PSMO commands with SGP 22</li> <li>• CR0410R00 Add refreshflag to ImmediateEnable</li> <li>• CR0411R01 Clarification about Notification sending</li> <li>• CR0412R01 Clarification of ipaeActivation by eUICC</li> <li>• CR0413R01 Clarification of the definition of Connectivity Parameters</li> <li>• CR0414R00 Delete IpaEuiccDataErrorCode from IpaEuiccDataResponse</li> <li>• CR0380R01 Editorial correction to change command to function</li> <li>• CR0389R00 Editorial Add definition for Event Deletion</li> <li>• CR0391R00 Add missing ES11' function calls &amp; associated refs</li> <li>• CR0401R00 Editorial to update StoreMetadata in text to correct function</li> <li>• CR0415R00 Renumber “ISD-R Selection and IP Ae Activation” section</li> <li>• CR0417R02 Split searchCriteria in IpaEuiccDataRequest</li> <li>• CR0418R01 ASN.1 RFU fields naming</li> <li>• CR0419R00 Correction to SetFallbackAttribute &amp; Emergency Profile – alternative</li> <li>• CR0420R00 Fixing rework of the tag numbers of ES10b.EuiccMemoryReset</li> <li>• CR0421R01 invalidPackageFormat error</li> <li>• CR0422R00 Cleaning error code for EnableProfile PSMO command</li> <li>• CR0424R02 Correction to tag number overlaps with SGP.22 v3.1</li> <li>• CR0423R00 Correction to ImmediateEnable function</li> </ul>		
<p>SGP.32 v1.2</p>	<p>27 June 2024</p>	<ul style="list-style-type: none"> <li>• First Draft of SGP.32 v1.2</li> <li>• CR12001R00 Editorial correction to DisableEmergencyFunction</li> <li>• CR12002R01 ProvideEimPackageResultResponse correction</li> <li>• CR12003R00 Delete EimPackageResultErrorCode from EimPackageResult</li> <li>• CR12004R00 Editorial correction to IpaEuiccDataRequest</li> <li>• CR12005R00 Missing User-Agent header definition</li> </ul>	<p>ISAG</p>	<p>Yolanda Sanz/GSMA (Editor)</p>

		<ul style="list-style-type: none"> <li>• CR12006R01 Editorial correction to direct profile download procedure</li> <li>• CR12007R00 Add reset of rollback authorization if refreshflag is set</li> <li>• Fix the implementation of CR CR12001R00 and CR12007R00:</li> <li>• CR12008R01 Add EidValue to CompactOtherSignedNotification</li> <li>• CR12009R00 Align direct and indirect profile download</li> <li>• CR12010R00 Correction to AddInitialEim function</li> <li>• CR12011R04 Correction to JSON bindings and ESipa function descriptions</li> <li>• CR12012R00 Adding a note clarifying max length of eIM ID</li> <li>• CR12013R00 Align JSON bindings with ASN.1</li> <li>• CR12014R02 Align JSON with ASN.1 in ProvideEimPackageResult and clarify need of EidValue</li> <li>• Fix editorial mistakes like adding/removing dots, spaces, quotation marks, etc</li> </ul>		
<p>SGP.32 v1.3</p>	<p>22 May 2026</p>	<ul style="list-style-type: none"> <li>• CR13001R00 Changes to Esipa x-admin-protocol value</li> <li>• CR13002R00 Correction of Setting Fallback Attribute procedure</li> <li>• CR13004R00 Clarify the presence of euiccCiPKId inGetEimConfigurationData</li> <li>• CR13003R02 Editorial update of Fallback procedure</li> <li>• CR13005R01 Diagram clarification of Fallback procedure</li> <li>• CR13006R01 Tag allocation list additions and corrections</li> <li>• CR13007R00 Generation of 'ctxParams1' by eIM</li> <li>• CR13008R01 Addition of missing ecallNotAvailable error code</li> <li>• CR13009R01 Changes to X-Admin-Protocol on Esipa</li> <li>• CR13010R00 Clarify handling of http POST response for ESipaMessageFromEimToIpa</li> <li>• CR13011R01 Addition of EIM OID ASN.1 definition</li> <li>• Undo CR13011R01 Addition of EIM OID ASN.1 definition as it was withdrawn</li> <li>• CR13012R01 Addition of euiccCiPKIdentifierToBeUsed to CompactOtherSignedNotification</li> <li>• CR13014R01 Clarification of profileChangeOngoing error case.</li> </ul>	<p>ISAG</p>	

		<ul style="list-style-type: none"> <li>• CR13015R04 eUICC Package Request maximums</li> <li>• CR13013R03 Addition of X-Admin-Protocol to CoAP</li> <li>• CR13017R03 Clarification on seqNumber incrementation on Profile rollback</li> <li>• CR13018R02 Connectivity Parameters clarifications</li> <li>• CR13020R02 Compact Authenticate Response ASN1 related note</li> <li>• CR13021R02 Cancel Session Procedure for Direct Profile Download</li> <li>• CR13022R01 Record currently enabled Profile information in ES10b.ExecuteFallbackMechanism</li> <li>• CR13023R02 Rollback Clarification when Fallback Profile was Enabled</li> <li>• CR13025R01 Clarifications on Fallback Mechanism</li> <li>• CR13026R01 Editorial correction on eUICCPackageRequest maximums size</li> <li>• CR13027R01 Clarification for the state of StoreMetadataRequest Boolean elements</li> <li>• CR13028R03. Resolving EN for connectivity parameters update by operator</li> <li>• CR13030R04 Vendor Specific Protocol Extension</li> <li>• CR13032R00 ASN.1 and JSON message definition HTTP POST method format correction</li> <li>• CR13033R00 Error fix in JSON binding of ESipa.AuthClient</li> <li>• CR13034R01 Clarification on association token definition.</li> <li>• CR13035R00 Addition of PROVIDE LOCAL INFORMATION (Location Information) to device requirements</li> <li>• CR13036R04 Download error response with cancel session cause</li> <li>• CR13038R01 ListProfileInfo_default_tag_list_clarification</li> <li>• CR13041R01 Correction on ES10b.ReturnFromFallback</li> <li>• CR13042R01 IP Ae_triggering</li> <li>• CR13043R00 Addition of PLI date and time</li> <li>• CR13040R01 Correction to notification handling for immediate enable</li> <li>• CR13044R01 Editorial change suggestions to accepted CRs in Draft 8</li> <li>• CR13047R06 Update Metadata over ES6 – update for IoT.</li> <li>• CR13048R03 Upgrade SGP.22 reference to v2.6.1</li> </ul>	<p>Hyewon Lee, Apple</p>	<p>Hyewon Lee, Apple</p>
--	--	--	--------------------------	--------------------------

		<ul style="list-style-type: none"> <li>• CR13049R00 ASN.1 format HTTP Path correction</li> <li>• CR13050R00 Editorial change in Function (ESipa) AuthenticateClient</li> <li>• CR13051R01 Handling of ASN.1 Default fields</li> <li>• CR13052R00 Clarification on IpaEuiccDataRequest regarding the Association Token</li> <li>• CR13054R00 Deletion of a Misleading Reference to Annex B.5</li> <li>• CR13055R00 Retrieval of eIM Configuration Data</li> <li>• CR13058R02 Remove duplicates with SGP.22</li> <li>• CR13059R03 Max_fragment_length TLS extension usage clarification</li> <li>• CR13060R02 Correction in direct and indirect profile download related to SGP.22 reference</li> <li>• CR13056R03 Indication of support of fallbackAllowed updateability</li> <li>• CR13061R02 Alignment with SGP.22 on long-term forward compatibility</li> <li>• CR13062R01 Upgrade SGP.22 reference to v2.7 and align the terminology</li> <li>• CR13066R01 IPA Correction on Cancel Session Procedure for Indirect Download</li> <li>• CR13067R03 Profile Installation result in Indirect Download</li> <li>• CR13069R00 Delete test profile</li> <li>• CR13039R03 Correction to eIM Package Injection (Alt to CR13031)</li> <li>• CR13064R02 Correction of eIM OID example</li> <li>• CR13068R05 Move eCallIndication verification</li> <li>• CR13070R01 Annex D Tag Allocation update</li> <li>• CR13071R00 Additional cleaning of references during eUICCMemoryReset</li> <li>• CR13073R02 IANA considerations for CoAP</li> <li>• CR13074R03 Relax restrictions for the deletion of the Profile to return form Fallback</li> <li>• CR13072R06 Alignment of PIR management for Indirect Profile Download</li> <li>• Editorial fix in ASN.1 module and Annex C</li> </ul>		
--	--	--	--	--

## Other Information

Type	Description
Document Owner	eSIM Group
Editor / Company	Hyewon Lee, Apple

It is our intention to provide a quality product for your use. If you find any errors or omissions, please contact us with your comments. You may notify us at [prd@gsma.com](mailto:prd@gsma.com)

Your comments or suggestions & questions are always welcome.