RSP Technical Specification
Version 2.1
27 February 2017

This is a Non-binding Permanent Reference Document of the GSMA

Security Classification: Non-confidential

Access to and distribution of this document is restricted to the persons permitted by the security classification. This document is confidential to the Association and 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

Disclaimer

The GSM Association ("Association") 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.

Antitrust Notice

The information contain herein is in full compliance with the GSM Association's antitrust compliance policy.

Table of Contents

1 Introduction

1.1 Overview

This document provides a technical description of the GSMA's 'Remote SIM Provisioning (RSP) Architecture for consumer Devices'.

1.2 Scope

This specification provides a technical description of:

- The eUICC Architecture;
- The interfaces used within the Remote SIM Provisioning Architecture; and
- The security functions used within the Remote SIM Provisioning Architecture.

1.3 Document Purpose

This document defines a technical solution for the remote provisioning and management of the eUICC in consumer Devices as defined in RSP Architecture [4]. The adoption of this technical solution will provide the basis for global interoperability between different Operator deployment scenarios, for example network equipment (e.g. Subscription Manager Data Preparation (SM-DP+)) and various eUICC platforms.

1.4 Intended Audience

Technical experts working for Operators, SIM solution providers, consumer Device vendors, standards organisations, network infrastructure vendors, Service Providers and other industry bodies, etc.

1.5 Definition of Terms

Term	Description
Activation Code	Information issued by an Operator/Service Provider to an End User. It is used by the End User to request the download and installation of a Profile.
Activation Code Token	A part of the Activation Code information provided by the Operator/Service Provider to reference a Subscription.
Alternative SM-DS	SM-DS used in cascade mode with a Root SM-DS to redirect Event Registration from a SM-DP+ to the Root SM-DS.
Authenticated Confirmation	A mechanism by which the End User confirms their action through a method involving the input of personalised information (e.g. PIN, fingerprint).
Bound Profile Package	A Protected Profile Package that has been cryptographically linked to a particular eUICC.
(Public Key) Certificate	A certificate as defined in RFC 5280 **Error! Reference source not found.Error! Reference source not found.**[17].
Certificate Authority	A Certificate Authority is an entity that issues digital certificates.
Certificate Issuer	An Entity that is Authorised to Issue digital certificates.
Companion Device	A Device that relies on the capabilities of a Primary Device for the purpose of Remote SIM Provisioning.

Confirmation Code	A code entered by an End User required by the SM-DP+ to confirm the download of a Profile.
Confirmation Code Required Flag	A parameter to indicate whether the Confirmation Code is required.
Device	User equipment used in conjunction with an eUICC to connect to a mobile network. E.g. a tablet, wearable, smartphone or handset.
Device Test Mode	A mode hidden from the End User that allows access to and use of Test Profiles.
Disabled (Profile)	The state of a Profile where all files and applications (e.g. NAA) present in the Profile are not selectable.
Enabled (Profile)	The state of a Profile when its files and/or applications (e.g., NAA) are selectable.
eUICC	A removable or non-removable UICC which enables the remote and/or local management of Profiles in a secure way. Note: The term originates from "embedded UICC".
eUICC Certificate	A certificate issued by the EUM for a specific eUICC. This Certificate can be verified using the EUM Certificate.
eUICC Memory Reset	An action that returns the eUICC to a state equivalent to a factory state.
eUICC Test Memory Reset	An action that deletes all post-issuance Test Profiles on an eUICC.
EUM Certificate	A certificate issued by a GSMA CI to a GSMA accredited EUM which can be used to verify eUICC Certificates.
Event	A Profile download which is set by an SM-DP+ on behalf of an Operator, to be processed by a specific eUICC.
EventID	Unique identifier of an Event for a specific EID generated by the SM-DP+ / SM-DS.
Event Record	The set of information stored on the SM-DS for a specific Event, via the Event Registration procedure. This information consists of either: • the Event-ID, EID, and SM-DP+ address or • the Event-ID, EID, and SM-DS address.
Event Registration	A process notifying the SM-DS on the availability of information on either a specific SM-DP+ or a specific SM-DS for a specific eUICC.
GSMA Certificate Issuer	A Certificate Authority accredited by GSMA.
Integrated Circuit Card ID	Unique number to identify a Profile in an eUICC as defined by ITU-T E.118 [21].
International Mobile Subscriber Identity	Unique identifier owned and issued by Mobile operators as defined in 3GPP TS 23.003 [35] section 2.2.
Issuer Identifier Number	The first 8 digits of the EID identifying the EUM issuing the eUICC.
Issuer Security Domain	A security domain on the UICC as defined by GlobalPlatform Card Specification [8].

Local Profile Assistant	A functional element in the Device or in the eUICC that provides the Local Profile Download (LPD), Local Discovery Services (LDS) and Local User Interface (LUI) features. When the LPA is located in the Device, they are called LPAd, LPDd, LUId, LDSd. When the LPA is located in the eUICC, they are called LPAe, LPDe, LUIe, LDSe. Where LPA, LPD, LDS or LUI are used, they apply to the element independent of its location in the Device or in the eUICC.
Local Profile Management	Local Profile Management are operations that are locally initiated on the End User (ESeu) interface.
Local Profile Management Operation	Local Profile Management Operations include enable Profile, disable Profile, delete Profile, query Profile Metadata, eUICC Memory Reset, eUICC Test Memory Reset, set/edit Nickname, add Profile and edit default SM-DP+ address.
MatchingID	Reference data for an RSP Server which could be an Activation Code Token or the EventID.
Mobile Network Operator	An entity providing access capability and communication services to its End User through a mobile network infrastructure.
Mobile Network Operator Security Domain (MNO-SD)	Part of the Profile, owned by the Operator, providing the Secured Channel to the Operator's Over The Air (OTA) Platform. It is used to manage the content of a Profile once the Profile is enabled.
Network Access Application	Application residing in a Profile providing authorisation to access a network.
NFC Device	A Device compliant with GSMA TS.26 [40].
Notification	A report about a Profile Installation or Local Profile Management Operation processed by the eUICC.
Operational Profile	A combination of Operator data and applications to be provisioned on an eUICC for the purposes of providing services by the Operator. The Profile SHALL be in support of a Subscription with the relevant Operator and allow connectivity to a mobile network. Applications MAY be included to provide non-telecommunication services.
Operator	A Mobile Network Operator or Mobile Virtual Network Operator; a company providing wireless cellular network services.
OTA Keys	The credentials included in the Profile, used in conjunction with OTA Platforms.
OTA Platform	An Operator platform for remote management of UICCs and the content of Enabled Operator Profiles on eUICCs.
Other Notification	Any Notification other than a Profile Installation Result.
PIX	Proprietary application Identifier extension, the value of which is part of the Application Identifier (AID).
Primary Device	A Device that can be used to provide some capabilities to a Companion Device for the purpose of Remote SIM Provisioning.
Profile	A combination of data and applications to be provisioned on an eUICC for the purpose of providing services.

Profile Component	A Profile Component is an element of the Profile, when installed in the eUICC, and MAY be one of the following: • An element of the file system like an MF, EF or DF; • An Application, including NAA and Security Domain; • Profile Metadata, including Profile Policy Rules; • An MNO-SD.
Profile Installation Result	A Notification that contains the result of a Profile installation.
Profile Management	A combination of local and remote management operations (e.g.: enable Profile, disable Profile, delete Profile, and query Profile Metadata).
Profile Management Operation	An operation related to the content and state update of a Profile in a dedicated ISD-P on the eUICC.
Profile Metadata	Information pertaining to a Profile used for the purpose of Local Profile Management.
Profile Nickname	Alternative name of the Profile set by the End User.
Profile Owner	The entity that controls the operations that can be performed upon its Profile. With the exception of Test Profiles, this is always the Operator.
Profile Package	A personalised Profile using an interoperable description format that is transmitted to an eUICC to load and install a Profile.
Profile Policy Authorisation Rule	A set of data that governs the ability of a Profile Owner to make use of a Profile Policy Rule in a Profile.
Profile Policy Enabler	The functional element within the Profile management system that interprets and enforces Profile Policy Rules.
Profile Policy Management	A policy control system that allows the Service Provider to implement, manage and enforce its subscription terms and conditions associated with the installed Profile.
Profile Policy Rule	Defines a qualification for or enforcement of an action to be performed on a Profile when a certain condition occurs.
Profile Type	Operator specific defined type of Profile. This is equivalent to the "Profile Description ID" as described in Annex B of SGP.21 [4]
Protected Profile Package	A Profile Package which has been cryptographically protected for storage but not linked to a particular eUICC.
Provisioning Profile	A combination of Operator data and applications to be provisioned on an eUICC for the purposes of providing connectivity to a mobile network solely for the purpose of the provisioning of Profiles on the eUICC. NOTE: Use of Provisioning Profiles for other system services in version 3 of this specification may require modifications of this definition.
Remote SIM Provisioning	The downloading, installing, enabling, disabling, and deleting of a Profile on an eUICC.
Roles	Roles are representing a logical grouping of functions.
Root SM-DS	A globally identified central access point for finding Events from one or more SM-DP+(s).

Rules Authorisation Table	A set of Profile Policy Authorisation Rules that, together, determines the ability of a Profile Owner to make use of a set of Profile Policy Rules in a Profile.
RSP Server	Either an SM-DS or SM-DP+.
Service Provider	The organization through which the End User obtains PLMN telecommunication services. This is usually the network operator or possibly a separate body.
Simple Confirmation	A mechanism by which the End User confirms their action, e.g. by selecting Yes/No, OK/Cancel.
SM-DP+ Certificate	A Certificate issued by a GSMA CI to a GSMA accredited SM-DP+.
SM-DS Certificate	A Certificate used by a GSMA CI to a GSMA accredited SM-DS.
SM-DP+ OID	Identifier of the SM-DP+ that is globally unique and is included as part of the SM-DP+ Certificate.
SM-DS OID	Identifier of the SM-DS that is globally unique and is included as part of the SM-DS Certificate.
Subscription	Describes the commercial relationship between the End User and the Service Provider.
Subscription Manager Data Preparation+ (SM-DP+)	This role prepares Profile Packages, secures them with a Profile protection key, stores Profile protection keys in a secure manner and the Protected Profile Packages in a Profile Package repository, and allocates the Protected Profile Packages to specified EIDs. The SM-DP+ binds Protected Profile Packages to the respective EID and securely downloads these Bound Profile Packages to the LPA of the respective eUICC.
Subscription Manager Discovery Server (SM-DS)	This is responsible for providing addresses of one or more SM-DP+(s) to a LDS.
Test Profile	A combination of data and applications to be provisioned on an eUICC to provide connectivity to test equipment for the purpose of testing the Device and the eUICC. A test profile is not intended to store any Operator Credentials.
User Intent	Describes the direct, real time acquisition and validation of the manual End User instruction on the LUI to trigger locally a Profile download or Profile Management Operation. As defined in SGP.21 [4].

1.6 Abbreviations

Abbreviation	Description
AID	Application Identifier
ASN.1	Abstract Syntax Notation One
BPP	Bound Profile Package
CA	Certificate Authority
CASD	Controlling Authority Security Domain

CAT	Card Application Toolkit
CERT.CI.ECDSA	Certificate of the CI for its Public ECDSA Key
CERT.DPauth.ECDSA	Certificate of the SM-DP+ for its Public ECDSA key used for SM-DP+ authentication
CERT.DPpb.ECDSA	Certificate of the SM-DP+ for its Public ECDSA key used for Profile Package Binding
CERT.DSauth.ECDSA	Certificate of the SM-DS for its Public ECDSA key used for SM-DS authentication
CERT.EUICC.ECDSA	Certificate of the eUICC for its Public ECDSA key
CERT.EUM.ECDSA	Certificate of the EUM for its Public ECDSA key
CERT.DP.TLS	Certificate of the SM-DP+ for securing TLS
CERT.DS.TLS	Certificate of the SM-DS for securing TLS
CI	Certificate Issuer
CMAC	Cipher-based MAC
CRL	Certificate Revocation List
CRT	Control Reference Template
DH	Diffie-Hellman
ECASD	eUICC Controlling Authority Security Domain
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve cryptography Digital Signature Algorithm
ECKA	Elliptic Curve cryptography Key Agreement algorithm
EID	eUICC-ID as defined in SGP.02 [2]
ETSI	European Telecommunications Standards Institute
EUM	eUICC Manufacturer
FFS	For Further Study
FQDN	Fully Qualified Domain Name
GID1	Group Identifier 1, as defined in 3GPP TS 31.102 [54]
GID2	Group Identifier 2, as defined in 3GPP TS 31.102 [54]
GP	GlobalPlatform
GSMA	GSM Association
HLR	Home Location Register
ICCID	Integrated Circuit Card ID
ICV	Initial Chaining Vector
IIN	Issuer Identifier Number
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
ISD	Issuer Security Domain
ISD-P	Issuer Security Domain Profile
ISD-R	Issuer Security Domain Root
ISO	International Standards Organisation

ITU	International Telecommunications Union
LDS	Local Discovery Service
LDSd	Local Discovery Service when LPA is in the Device
LDSe	Local Discovery Service when LPA is in the eUICC
LPA	Local Profile Assistant
LPAd	Local Profile Assistant when LPA is in the Device
LPAe	Local Profile Assistant when LPA is in the eUICC
LPD	Local Profile Download
LPDd	Local Profile Download when LPA is in the Device
LPDe	Local Profile Download when LPA is in the eUICC
LTE	Long Term Evolution
LUI	Local User Interface
LUId	Local User Interface when LPA is in the Device
LUIe	Local User Interface when LPA is in the eUICC
M4M	Mifare4Mobile™
MAC	Message Authentication Code
MEP	Message Exchange Pattern
MNO	Mobile Network Operator
MOC	Mandatory, Optional or Conditional
NAA	Network Access Application
OTA	Over The Air
otPK.DP.ECKA	One-time Public Key of the SM-DP+ for ECKA
otPK.EUICC.ECKA	One-time Public Key of the eUICC for ECKA
otSK.DP.ECKA	One-time Private Key of the SM-DP+ for ECKA
otSK.EUICC.ECKA	One-time Private Key of the eUICC for ECKA
PE	Profile Element
PIX	Proprietary application Identifier eXtension
PKI	Public Key Infrastructure
PK.CI.ECDSA	Public Key of the CI, part of the CERT.CI.ECDSA
PK.DPauth.ECDSA	Public Key of the SM-DP+ part of the CERT.DPauth.ECDSA
PK.DPpb.ECDSA	Public Key of the SM-DP+ part of the CERT.DPpb.ECDSA
PK.DSauth.ECDSA	Public Key of the SM-DS part of the CERT.DSauth.ECDSA
PK.EUICC.ECDSA	Public Key of the eUICC, part of the CERT.EUICC.ECDSA
PK.EUM.ECDSA	Public Key of the EUM, part of the CERT.EUM.ECDSA
POS	Point Of Sale
PPAR	Profile Policy Authorisation Rule
PPE	Profile Policy Enabler
PPK-ENC	Profile Protection Key for message encryption/decryption
PPK-MAC	Profile Protection Key for message MAC generation/verification

PPP	Protected Profile Package
PPR	Profile Policy Rule
RAT	Rules Authorisation Table
RFU	Reserved for Future Use
RSA	Rivest / Shamir / Adleman asymmetric algorithm
RSP	Remote SIM Provisioning
SAS	Security Accreditation Scheme
SBPP	Segmented Bound Profile Package
SCP	Secure Channel Protocol
SCWS	Smartcard Web Server
SD	Security Domain
S-ENC	Session key for message encryption/decryption
S-MAC	Session Key for message MAC generation/verification
ShS	Shared Secret
SK.CI.ECDSA	Private key of the CI for signing certificates
SK.DPauth.ECDSA	Private Key of the of SM-DP+ for creating signatures for SM-DP+ authentication
SK.DPpb.ECDSA	Private key of the SM-DP+ used to provide signatures for Profile binding
SK.DSauth.ECDSA	Private Key of the of SM-DS for creating signatures for SM-DS authentication
SK.EUICC.ECDSA	Private key of the eUICC for creating signatures
SK.EUM.ECDSA	Private key of the EUM for creating signatures
SK.DP.TLS	Private key of the SM-DP+ for securing TLS connection
SK.DS.TLS	Private key of the SM-DS for securing TLS connection
SM-DP+	Subscription Manager Data Preparation (Enhanced compared to the SM-DP in SGP.02 [2])
SVN	SGP.22 Specification Version Number (referred to as 'eSVN' in SGP.21 [4]).
TAC	Type Allocation Code
TAR	Toolkit Application Reference
TLS	Transport Layer Security
TLV	Tag-Length-Value
UPP	Unprotected Profile Package
URI	Uniform Resource Identifier
URL	Uniform Resource locator
USIM	Universal Subscriber Identity Module
W3C	World Wide Web Consortium

1.7 References

Ref	Document Number	Title
[1]	TF_Req	GSMA "Embedded SIM Task Force Requirements and Use Cases" Version 1.0
[2]	SGP.02	GSMA "Remote Provisioning of Embedded UICC Technical specification" V3.1
[3]	void	Void
[4]	SGP.21	RSP Architecture V2.0
[5]	SIMalliance	SIMalliance eUICC Profile Package: Interoperable Format Technical Specification V2.0
[6]	ETSI TS 102 221	Smart Cards; UICC-Terminal interface
[7]	OMA-TS-Smartcard_Web_Server-V1_2_1-20130913-A	Open Mobile Alliance: Smartcard-Web-Server, Version 1.2.1 – 13 Sep 2013
[8]	GPC_SPE_034	GlobalPlatform Card Specification v.2.3
[9]	GPC_SPE_007	GlobalPlatform Card Specification v.2.3 Amendment A: Confidential Card Content Management v1.1
[10]	GPC_SPE_025	GlobalPlatform Card Specification v.2.3 Amendment C: Contactless Services v1.2
[11]	GPC_SPE_014	GlobalPlatform Card Specification v.2.2 Amendment D: Secure Channel Protocol '03' v1.1.1
[12]	GPC_SPE_042	GlobalPlatform Card Specification v.2.2 Amendment E: Security Upgrade for Card Content Management v1.0
[13]	GPC_SPE_093	GlobalPlatform Card Specification v.2.2 Amendment F: Secure Channel Protocol '11'
[14]	ISO/IEC 7816-4:2013	Identification cards – Integrated circuit cards - Part 4: Organization, security and commands for interchange
[15]	ISO/IEC 18004:2015	Information technology -- Automatic identification and data capture techniques -- QR Code bar code symbology specification
[16]	RFC 5246	The TLS Protocol – Version 1.2
[17]	RFC 5280	Internet X.509 PKI Certificate and CRL Profile
[18]	RFC 5639	Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation
[19]	RFC 793	Transmission Control Protocol, DARPA Internet Program, Protocol specification, Sept 1981
[20]	ANSSI ECC FRP256V1	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
[21]	ITU E.118	The international telecommunication charge card
[22]	GSMA Security Principles Related to Handset Theft	GSMA Doc Reference: Security Principles Related to Handset Theft 3.0.0 EICTA CCIG Doc Reference: EICTA Doc: 04cc100

[23]	GSMA SAS-SM	GSMA SAS Standard for Subscription Manager Roles Version 2.0 - 13 May 2015
[24]	ITU-T X.520	ITU-T X.520 Information technology – Open Systems Interconnection – The Directory: Selected attribute types
[25]	RFC 5758	RFC 5758 Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA
[26]	RFC 5759	RFC 5759 Suite B Certificate and Certificate Revocation List (CRL) Profile
[27]	RFC 5480	RFC 5480 Elliptic Curve Cryptography Subject Public Key Information
[28]	RFC 4519	Lightweight Directory Access Protocol (LDAP)
[29]	NIST SP 800-56A	NIST Special Publication SP 800-56A: Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revision 2), May 2013
[30]	ITU E.212	The international identification plan for public networks and Subscriptions
[31]	ETSI TS 102 223	Smart Cards; Card Application Toolkit (CAT)
[32]	3GPP TS 24.008	Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; Mobile radio interface Layer 3 specification; Core network protocols; Stage 3
[33]	ETSI TS 101 220	Smart Cards; ETSI numbering system for telecommunication application providers
[34]	RFC 768	User Datagram Protocol, Aug 1980.
[35]	3GPP TS 23.003	Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Numbering, addressing and identification
[36]	3GPP2 S.R0048-A	3GPP2 - 3G Mobile Equipment Identifier (MEID)
[37]	ISO/IEC 7812-1:2015	Identification cards -- Identification of issuers -- Part 1: Numbering system
[38]	ETSI TS 102 225	Secured packet structure for UICC based applications; Release 12
[39]	ETSI TS 102 226	Remote APDU structure for UICC based applications; Release 9
[40]	TS.26	GSMA NFC Handset Requirements V9.0
[41]	BSI TR-03111	BSI Technical Guideline; Elliptic Curve Cryptography
[42]	Draft-ietf-tls-tls13-12	The Transport Layer Security (TLS) Protocol Version 1.3, Draft 12, draft-ietf-tls-tls13-12
[43]	RFC 2986	PKCS #10: Certification Request Syntax Specification
[44]	RFC 6960	X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP

[45]	SGP.14	GSMA eUICC PKI Certificate Policy V1.0
[46]	RFC 5289	TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)
[47]	RFC 4279	Pre-Shared Key Cipher suites for Transport Layer Security (TLS)
[48]	RFC 2616	Hypertext Transfer Protocol -- HTTP/1.1
[49]	ITU-T X.680 (11/2008)	Abstract Syntax Notation One (ASN.1): Specification of basic notation including Corrigendum 1 and 2
[50]	ITU-T X.690 (11/2008)	ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER) including Corrigendum 1 and 2
[51]	3GPP TS 35.231	Specification of the TUAK Algorithm Set; Document 1: Algorithm Specification
[52]	3GPP TS 35.205	Specification of the MILENAGE Algorithm Set; Document 1: General
[53]	ETSI TS 102 241	Smart cards; UICC Application Programming Interface (UICC API) for Java Card™
[54]	3GPP TS 31.102	Characteristics of the Universal Subscriber Identity Module (USIM) application
[55]	SGP.03	GSMA NFC UICC Requirements Specification V6.1
[56]	GPD_SPE_013	GlobalPlatform Device Technology – Secure Element Access Control - Version 1.1
[57]	GPC_SPE_095	GlobalPlatform Card - Digital Letter of Approval - Version 1.0
[58]	M4M	MIFARE4Mobile Architecture – V 2.1.1
[59]	ISO/IEC 10646:2014	Information technology — Universal Coded Character Set (UCS)
[60]	RFC 6066	Transport Layer Security (TLS) Extensions: Extension Definitions
[61]	RFC 2119	Key words for use in RFCs to Indicate Requirement Levels, S. Bradner http://www.ietf.org/rfc/rfc2119.txt
[62]	3GPP TS 34.108	Common test environments for User Equipment (UE); Conformance testing
[63]	3GPP TS 29.002	Mobile Application Part (MAP) specification

1.8 Conventions

The key words "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document SHALL be interpreted as described in RFC 2119 [61].

2 General Architecture

This section contains a technical description and architecture of the Remote SIM Provisioning System for consumer 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 Diagram

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

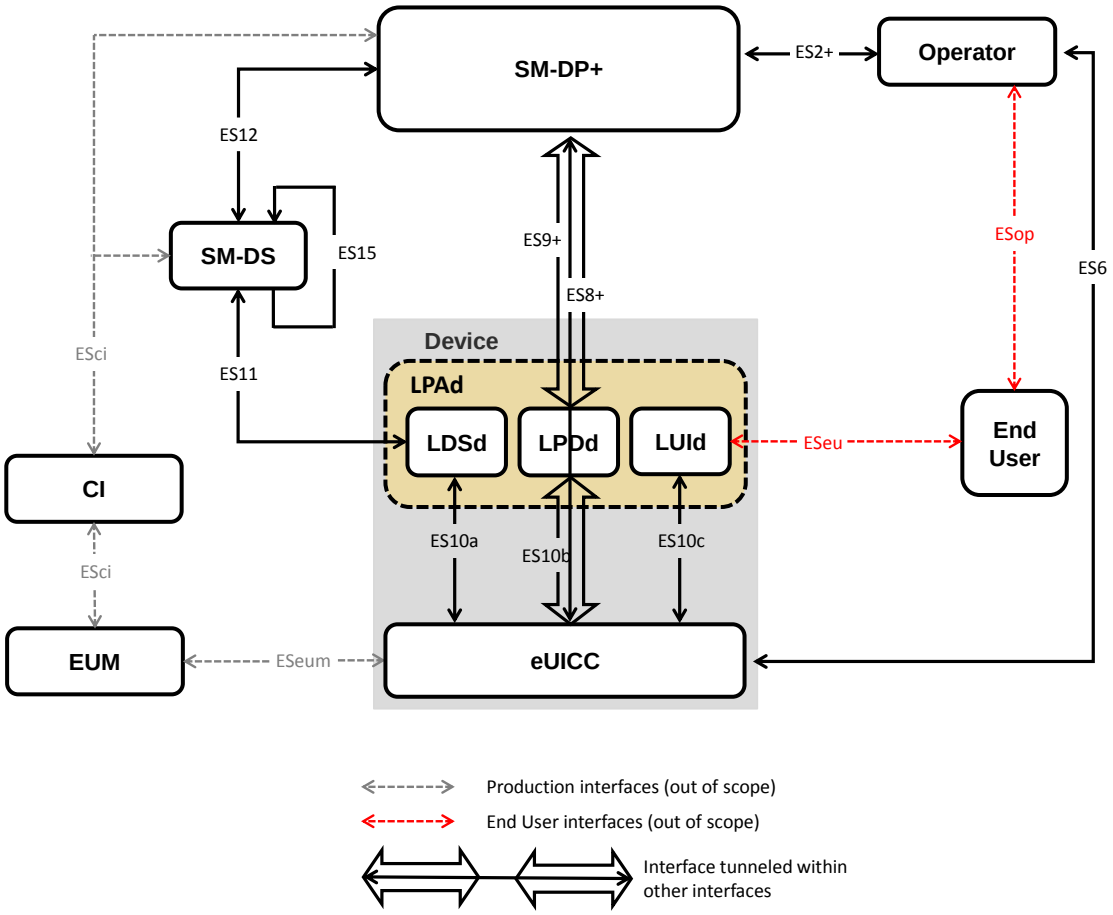


Figure 1: Remote SIM Provisioning System, LPA in the Device

A Device compliant with this specification SHALL implement at least one of the following:

- the LPAd, or
- the requirements for one of the options for the LPAe (section 5.11).

A Device that supports a non-removable eUICC without an LPAe SHALL provide an LPAd.

An eUICC compliant with this specification SHALL implement the LPA Services and optionally the LPAe.

A Device supporting both the LPAd and the LPAe SHALL implement an appropriate mechanism that sets the LPA to be used.

The above figure provides the complete description of the consumer Remote SIM Provisioning and Management system, when LPA is in the Device (LPAd).

The Remote SIM Provisioning and Management system also allows to have the LPA in the eUICC (LPAe). This architecture is shown in the following figure.

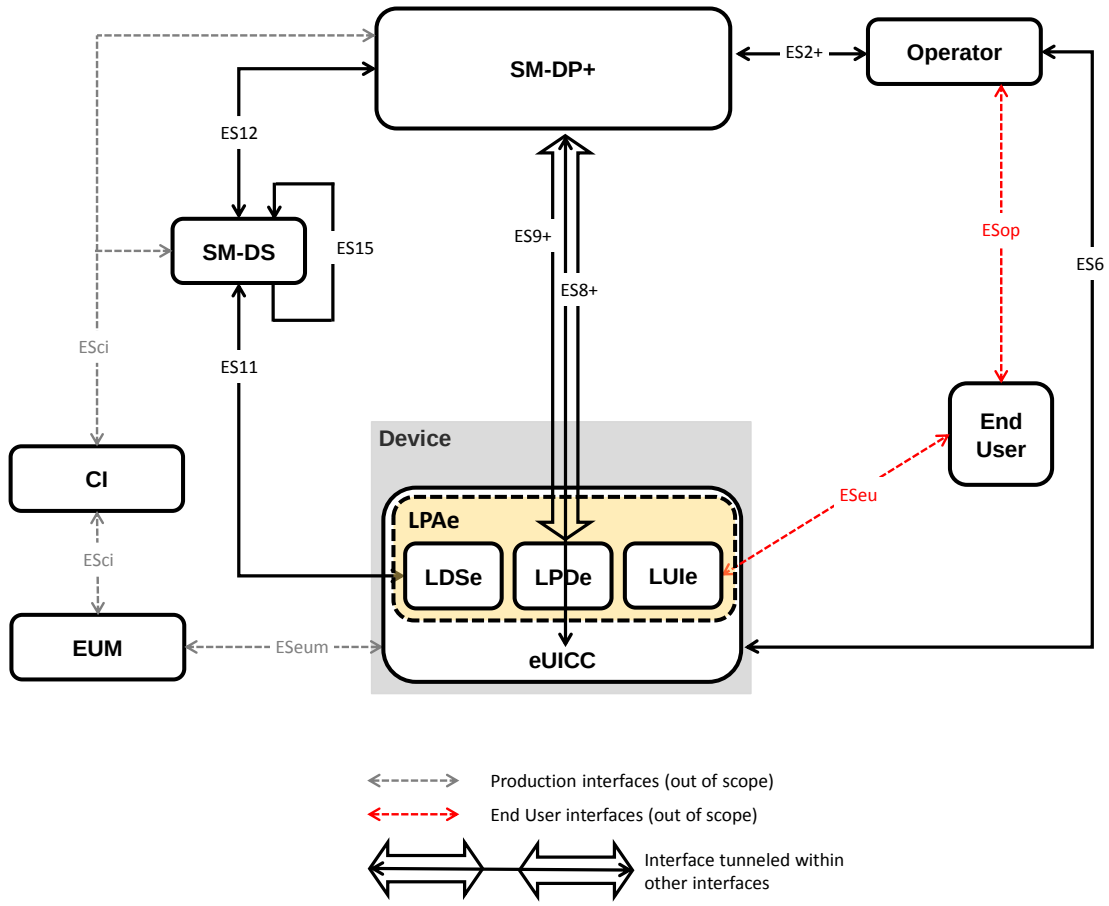


Figure 2: Remote SIM Provisioning System, LPA in the eUICC

2.2 Roles

Roles are defined within SGP.21 [4] Architecture Specification 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.
ES6	Operator	eUICC	Used by the Operator for the management of Operator services via OTA services.
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.

Interface	Between		Description
ES9+	SM-DP+	LPD	Used to provide a secure transport between the SM-DP+ and the LPA (LPD) for the delivery of the Bound Profile Package.
ES10a	LDSd	eUICC	Used between the LDSd and the LPA Services to handle a Profile discovery.
ES10b	LPDd	eUICC	Used between the LPDd and the LPA services to transfer a Bound Profile Package to the eUICC. This interface plays no role in the decryption of Profile Packages.
ES10c	LUId	eUICC	Used between the LUId and the LPA services for Local Profile Management by the End User.
ES11	LDS	SM-DS	Used by the LDS to retrieve Event Records for the respective eUICC.
ES12	SM-DP+	SM-DS	Used by the SM-DP+ to issue or remove Event Registrations on the SM-DS.
ES15	SM-DS	SM-DS	Used in the case of deployments of cascaded SM-DSs to connect those SM-DSs.
ESop	Operator	End User	These interfaces will not be specified within the technical specification. They are specific to business relationships between the entities involved.
ESeu	End User	LUI	
ESeum	eUICC	EUM	
ESci	CI	SM-DP+ SM-DS EUM	This interface is used by the SM-DP+, SM-DS and EUM to request a Certificate and retrieve Certificate revocation status. Any other relying party MAY retrieve Certificate revocation status. This interface, even if not specified within the technical specification, SHALL rely on RFC 2986 [43] for a Certificate request, and RFC 5280 [17] for CRL retrieval.

Table 1:Interfaces

2.4 eUICC Architecture

2.4.1 eUICC Overview

This section describes the internal high-level architecture of the eUICC. It should be noted that the eUICC architecture is very similar to that used in the GSMA Remote SIM Provisioning of Embedded UICC Technical specification [2]. Operator Profiles are stored inside security domains within the eUICC and are implemented using GlobalPlatform standards. These ensure that it is impossible for any Profile to access the applications or data of any other Profile stored on the eUICC. The same mechanism is currently in use within SIM cards to ensure payment applications are kept secure.

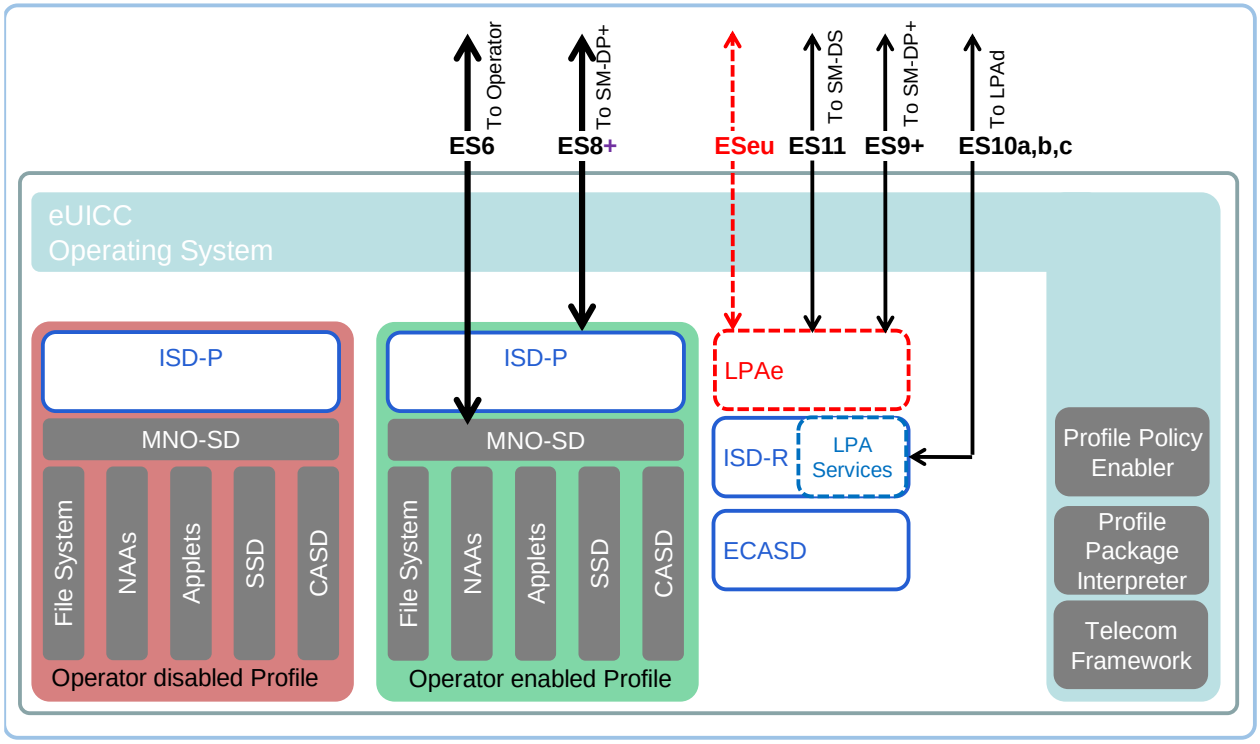


Figure 3: Schematic Representation of the eUICC

2.4.2 ECASD

The Embedded UICC Controlling Authority Security Domain (ECASD) is responsible for secure storage of credentials required to support the required security domains on the eUICC.

There SHALL be only one ECASD on an eUICC. The ECASD SHALL be installed and personalized by the EUM (eUICC Manufacturer) during the eUICC manufacturing. After eUICC manufacturing, the ECASD SHALL be in life-cycle state PERSONALIZED as defined in GlobalPlatform Card Specification [8] section 5.3.

The AID of the ECASD SHALL follow SGP.02 [2].

The ECASD SHALL contains

- The eUICC's Private Key(s) (SK.EUICC.ECDSA) for creating ECDSA signatures
- The eUICC's Certificate(s) for eUICC authentication (CERT.EUICC.ECDSA) containing the eUICC's public key (PK.EUICC.ECDSA)
- The GSMA Certificate Issuer's (CI) Public Key(s) (PK.CI.ECDSA) for verifying off-card entities certificates (e.g. SM-DP+) and Certificate Revocation List (CRL). ECASD MAY contain several public keys belonging to the same GSMA CI or different GSMA CIs. Each PK.CI.ECDSA SHALL be stored with information coming from the CERT.CI.ECDSA the key is included in, at least:
    - Certificate serial number: required to manage GSMA CI revocation by CRL
    - GSMA Certificate Issuer identifier: GSMA CI OID

  - Subject Key Identifier: required to verify the Certification chain of the off-card entity

- The Certificate(s) of the EUM (CERT.EUM.ECDSA)

The ECASD SHOULD also contain:

- eUICC Manufacturer's (EUMs) keyset for key/certificate renewal:
  - Renew eUICC's Private Key(s) and Certificate(s)
  - Renew EUM Certificate(s)
  - Renew CI public key(s)

The means by which the EUM SHOULD perform key/certificate renewal is out of scope of this specification but, if provided, it SHALL be a GlobalPlatform [8] mechanism with a minimum security level corresponding to the AES algorithm using a minimum key length of 128 bits. EUM MAY also do GSMA CI Certificate revocation on eUICC (e.g. by deleting the related public key), in addition to using a CRL loaded by the LPA (sections 4.6 and 5.7.12).

The ECASD SHALL provide the following services to the ISD-R:

- eUICC signature creation on material provided by an ISD-R
- Verification of the off-card entities Certificates (e.g. SM-DP+), provided by an ISD-R, with the CI public key (PK.CI.ECDSA)

Personalisation of the ECASD SHALL be done in a certified 'GSMA SAS-UP environment'.

NOTE: Per NIST publication SP800-57 part.1 [29], ECC256 (128-bit security strength) is sufficient for current implementation beyond year 2031.

2.4.3 ISD-R

The ISD-R is responsible for the creation of new ISD-Ps and lifecycle management of all ISD-Ps.

There SHALL be only one ISD-R on an eUICC.

The ISD-R SHALL be installed and personalized by the EUM during eUICC manufacturing. The ISD-R SHALL be associated with itself. The ISD-R privileges SHALL be granted according to Annex A.

The ISD-R cannot be deleted or disabled.

2.4.4 ISD-P

The ISD-P is the on-card representative of the SM-DP+ and is a secure container (Security Domain) for the hosting of a Profile. The ISD-P is used for the Profile download and installation in collaboration with the Profile Package Interpreter for the decoding/interpretation of the received Profile Package.

An ISD-P hosts a unique Profile.

No component outside the ISD-P SHALL have visibility or access to any Profile Component with the exception of the ISD-R, which SHALL have access to Profile Metadata.

A Profile Component SHALL not have any visibility of, or access to, components outside its ISD-P. An ISD-P SHALL not have any visibility of, or access to, any other ISD-P.

Deletion of a Profile SHALL remove the containing ISD-P and all Profile Components of the Profile.

2.4.5 Profile

A Profile consists of Profile Components:

- One MNO-SD
- Supplementary Security Domains (SSD) and a CASD
- Applets
- Applications, e.g. NFC applications
- NAAs
- Other elements of the File System
- Profile Metadata, including Profile Policy Rules

The MNO-SD is the on-card representative of the Operator. It contains the Operator's Over-The-Air (OTA) keys and provides a secure OTA channel.

All security domains of a Profile SHALL be located in the hierarchy of the MNO-SD or an SD extradited to itself.

The behaviour of an eUICC with an Enabled Profile SHALL be equivalent to a UICC. This applies especially for the NAAs and applets contained in the Profile.

When a Profile is Disabled, the eUICC SHALL ensure that:

- Remote management of any Profile Component is not possible via the ES6 interface.
- The file system within the Profile cannot be selected by the Device or any application on the eUICC.
- The applications (including NAAs and Security Domains) within the Profile cannot be selected, triggered or individually deleted.
- For an eUICC compliant with M4M [58], no M4M Virtual Card inside that Profile is visible nor accessible through any interface.

2.4.5.1 Operational Profile

An Operational Profile SHALL have its Profile Class set to 'operational' in its Profile Metadata to indicate to the LPA and the eUICC that it SHALL be handled in the manner that is appropriate for an Operational Profile.

2.4.5.2 Provisioning Profile

A Provisioning Profile SHALL have its Profile Class set to 'provisioning' in its Profile Metadata to indicate to the LPA and the eUICC that it SHALL be handled in the manner that is appropriate for a Provisioning Profile. In every other respect, a Provisioning Profile SHALL have the same format structure as any other Profile.

Provisioning Profile Impact on LUI

Provisioning Profiles and their associated Profile Metadata SHALL not be visible to the End User in the LUI. As a result, Provisioning Profiles SHALL not be selectable by the End User nor deletable through any End User action, including eUICC Memory Reset.

Provisioning Profile and Operational Profile Policies

Provisioning Profiles SHALL still be usable, even if the currently enabled Operational Profile is subject to Profile Policy Rule 'ppr1'. In the case where a Provisioning Profile needs to be enabled, the LPA SHALL directly enable the Provisioning Profile, without first explicitly disabling the currently enabled Operational Profile; the eUICC SHALL allow this operation and implicitly disable the currently enabled Operational Profile without enforcing the Profile Policy Rule.

2.4.5.3 Test Profile

An eUICC MAY support Test Profiles.
A Test Profile SHALL have its Profile Class set to 'test' in its Profile Metadata to indicate to the LPA and the eUICC that it SHALL be handled in the manner that is appropriate for a Test Profile. To ensure that a Test Profile is not used as an Operational Profile, the value of its key(s) for network authentication SHALL comply with one of the following:

1. All bits set to zero except the lowest 32 bits.
2. '00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F' (default K value of Test USIM as defined in Section 8.2 of 3GPP TS 34.108 [62]).
3. Any arbitrary value, if the network authentication algorithm is the Test Algorithm as defined in Section 8.1.2 of 3GPP TS 34.108 [62] or the IMSI value complies with the Test USIM IMSI defined in Section 8.3.2.2 of 3GPP TS 34.108 [62].

When a Test Profile is downloaded, the eUICC SHALL accept keys compliant with (1) and SHOULD accept keys compliant with (2) or (3).

Preloaded Test Profiles SHALL comply with one of these conditions.

In every other respect, a Test Profile SHALL have the same format structure as any other Profile.

NOTE: A live commercial network would never use a key with so little entropy.

Test Profiles and their associated Profile Metadata SHALL be visible in the LUI when the Device is in Device Test Mode.

NOTE: While the LPAd can receive an implementation specific notification when the Device enters/leaves Test Mode, such a mechanism is not available for the LPAe. To avoid extra messages, the LPAe MAY implement a protected option in its user interface for activating Test Mode.

2.4.6 Telecom Framework

The Telecom Framework is an Operating System service that provides standardised network authentication algorithms to the NAAs hosted in the ISD-Ps. Furthermore, it provides the capabilities to configure the algorithm with the necessary parameters in the Enabled Profile.

2.4.7 Profile Package Interpreter

The Profile Package Interpreter is an eUICC Operating System service that translates the Profile Package data as defined in SIMalliance eUICC Profile Package Specification [5] into an installed Profile using the specific internal format of the target eUICC.

2.4.8 LPAe

The LPAe is a functional element that provides the LPDe, LDSe and LUIe features. These features are similar to the features of an LPAd.

LPAe is optional.

The technical implementation of LPAe is up to the EUM. For example, the LPAe MAY be a feature of the ISD-R.

2.4.9 LPA Services

This role provides the necessary access to the services and data required by LPA functions. These services are:

- Transfer Bound Profile Package from the LPAd to the ISD-P
- Provide list of installed Profiles
- Retrieve EID
- Provide Local Profile Management Operations

LPA Services are mandatory even if the LPAe is provided in the eUICC.

2.4.10 Hardware Characteristics of the eUICC

The following requirements apply:

- The eUICC SHALL be a discrete tamper resistant hardware component.
- The eUICC MAY either be removable or non-removable. A removable eUICC SHALL be packaged in a Form Factor specified in ETSI TS 102 221 [6].

2.4.11 Platform Characteristics of the eUICC

The following requirements apply:

- The eUICC SHALL support SHA-1.
- The eUICC SHALL support TUAK [51].
- The eUICC SHALL support Milenage [52].
- All cryptographic functions SHALL be implemented in a tamper-resistant way and SHALL resist side-channel attacks.

2.4.11.1 Java Card packages

An eUICC supporting Java Card™ SHALL support the Java Packages listed below. The implementation of each Package SHALL as a minimum be according to the given Package version and Specification version.

Package	Description	Package Version	Spec Version
java.* (java.lang, java.io)	**java.io** defines a subset of the java.io package in the standard Java programming language.	1.0	Java Card™ 3.0.4 Classic
	java.lang Provides classes that are fundamental to the design of the Java Card technology subset of the Java programming language.	1.0	
javacard.* (javacard.framework, javacard.security)	**javacard.framework** Provides a framework of classes and interfaces for building, communicating with and working with Java Card technology based applets.	1.5	Java Card™ 3.0.4 Classic
	javacard.security Provides classes and interfaces that contain publicly available functionality for implementing a security and cryptography framework on the Java Card platform	1.5	
javacardx.* (javacardx.crypto)	**javacardx.crypto** Extension package that contains functionality, which may be subject to export controls, for implementing a security and cryptography framework on the Java Card platform.	1.5	Java Card™ 3.0.4 Classic
uicc.* (uicc.access, uicc.system, uicc.toolkit)	**uicc.access** This package provides the means to the applets for accessing the UICC file system defined in the TS 102 221 [6] specification.	1.2	TS 102 241 [53] Rel-9
	uicc.system This package provides the means to the applets for accessing system wide services of the UICC platform.	1.0	
	uicc.toolkit This package provides the means for the toolkit applets to register to the events of the common application toolkit (CAT) framework, to handle TLV information and to send proactive commands according to the ETSI TS 102 223 [31] specification.	1.6	
uicc.usim.* (uicc.usim.access, uicc.usim.toolkit)	**uicc.usim.access** This package provides the means for applets to get access to the files defined in the USIM,SIM and ISIM specification.	1.0	3GPP TS 31.130 Rel-9
	uicc.usim.toolkit This package provides the means for the toolkit applets to register to the events defined in the USAT and STK	1.8	

	specification to handle TLV information and to send proactive command according to the ETSI 3GPP TS 31.111 and 3GPP TS 51.014 specification.		
org.globalplatform	Provides a framework of classes and interfaces related to core services defined for smart cards based on GlobalPlatform specifications	1.6	GP 2.3 [8]

Table 2: Java Card Packages

The following additional Java Card packages SHALL be supported by an eUICC supporting NFC:

Package	Description	Package Version	Spec Version
javacardx.* (javacardx.external, javacardx.framework.tlv, javacardx.framework.util.intx)	**javacardx.external** Extension package that provides mechanisms to access memory subsystems which are not directly addressable by the Java Card runtime environment (Java Card RE) on the Java Card platform.	1.0	Java Card™ 3.0.4 Classic
	javacardx.framework.tlv Extension package that contains functionality, for managing storage for BER TLV formatted data, based on the ASN.1 BER encoding rules of ISO/IEC 8825-1:2002, as well as parsing and editing BER TLV formatted data in I/O buffers.	1.0	
	javacardx.framework.util.intx Extension package that contains common utility functions for using `int` components.	1.0	
uicc.hci.* (uicc.hci.framework, uicc.hci.services.cardemulation, uicc.hci.services.connectivity, uicc.hci.services.readermode)	**uicc.hci.framework** This package defines the basic interfaces for the HCI protocol.	1.1	TS 102 705 Rel-9
	uicc.hci.services.cardemulation This package defines the interfaces for the card emulation mode of the HCI protocol.	1.0	
	uicc.hci.services.connectivity This package defines the interfaces for the functionality of the connectivity gate defined in the HCI protocol.	1.0	
	uicc.hci.services.readermode This package defines the interfaces for	1.0	

	the reader emulation mode of the HCI protocol.		
org.globalplatform.contactless	Provides a framework of classes and interfaces related to contactless services defined for smart cards based on GlobalPlatform specifications	1.3	Amd C 1.2 [10]

Table 3: Java Card Packages

2.4.12 Profile Policy Enabler

The Profile Policy Enabler is described in detail in section 2.9.3.

2.5 Profile Protection and Delivery

This section describes how an Operator's Profile is protected within a Profile Package prior to being downloaded to the eUICC. This also applies when the Profile Package is protected during transmission between Roles within the system.

2.5.1 Profile Package Types Overview

From generation to download, the Profile Package will take different formats. This specification uses the following terms:

- Unprotected Profile Package (UPP): Raw SIMalliance TLV sequence.
- Protected Profile Package (PPP): Segmented and protected in SCP03t TLVs.
- Bound Profile Package (BPP): Prepended with session key agreement info, key replacement package, ISD-P creation and configuration info.
- Segmented Bound Profile Package (SBPP): BPP segmented into STORE DATA APDU script for loading into eUICC. This step in performed by the LPD when LPD is in the Device.

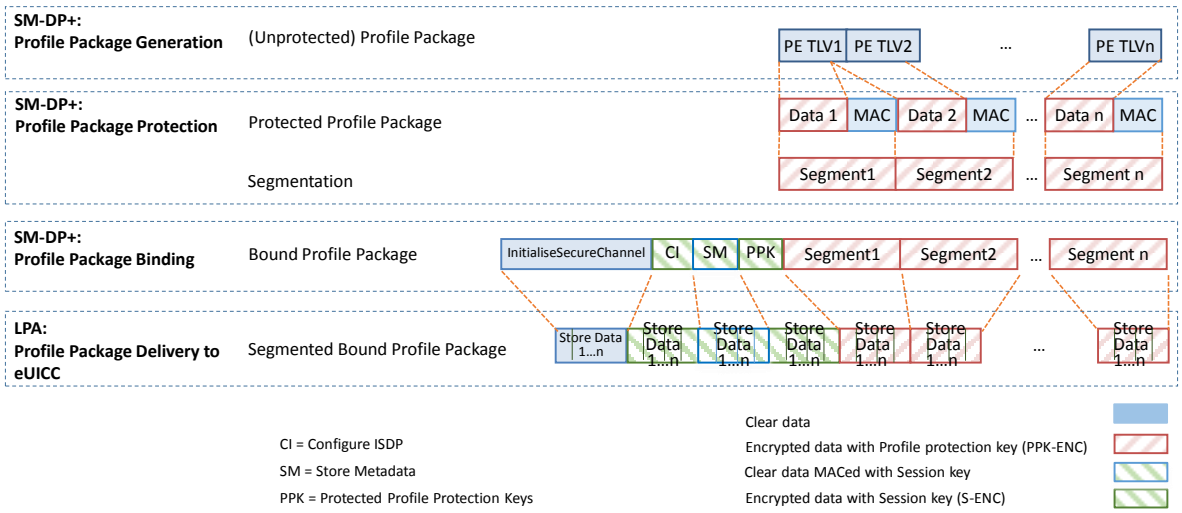


Figure 4: Profile Package stage Description

The above diagram describes the case where Profile Package is protected with keys different from the session keys established during the key agreement with the eUICC

(S-ENC, S-MAC). It MAY also be possible to have a Profile Package protected with the session keys; in that case the 'Profile Protection keys' block SHALL not be present.

2.5.2 Unprotected Profile Package

The Unprotected Profile Package (UPP) is generated by the SM-DP+, within the Profile Package Generation function. The Profile Package Generation takes as input the profile specification established with the Operator and input data provided by the Operator. The processes of profile specification and input data acquisition are out of scope of this specification.

The Unprotected Profile Package consists of a sequence of Profile Element (PE) TLVs according to the SIMalliance specification [5].

2.5.3 Protected Profile Package

The Protected Profile Package (PPP) is generated by the SM-DP+, within the Profile Package Protection function.

The PPP SHALL be protected with SCP03t. Command TLV encryption and MACing follows SGP.02 [2] section 4.1.3.3. During this step the internal UPP structure is not considered, and rather seen as a unique block of data. That block of data is split into segments of a maximum size of 1020 bytes (including the tag, length field and MAC). The eUICC SHALL support receiving data segments of at least up to this size.

NOTE: From the 1020 bytes of each data segment, only 1008 bytes are usable for payload (deducted the 1 byte tag, 3 bytes length field and 8 bytes MAC). Considering the necessary padding during encryption (16 bytes length block encryption and necessary '80' byte padding), then each data segment can only contain 1007 bytes of the PPP data block.

Profile protection SHALL performed using either:

- Session keys (S-ENC, S-MAC) resulting from the key agreement with eUICC (Matching ID section 4.1.1).
- Or random keys per Profile (denoted PPK-ENC and PPK-MAC in this document and referred to as S-ENC and S-MAC respectively in SGP.02 [2]), generated by the SM-DP+.

If random key mode is selected by the SM-DP+, the initial MAC chaining value to be used for the first segment of the PPP is provided together with the random key (section 5.5.4) and the encryption counter for ICV calculation is reset to its initial state (i.e. the value on 16 bytes is '00…01'). Otherwise the MAC chaining method defined in SGP.02 [2] SHALL be applied (i.e. the MAC chaining value of the previous SHALL be used).

S-ENC, S-MAC, PPK-ENC and PPK-MAC SHALL be 128 bits length.

Each data segment of the PPP is identified by the tag '86' as defined in SGP.02 [2].

It is the SM-DP+ choice (in line with the agreement with the Operator) as to whether to use this random keyset (PPK-ENC and PPK-MAC). This mode allows performing Profile

Package Protection in advance without having any eUICC knowledge. It may help to provide a better SM-DP+ scalability. The eUICC SHALL be able to support both modes.

In case the random key mode is used, the PPP is not bound to any particular eUICC or ISD-P AID value at this stage.

Session keys and, if used, the random keys SHALL only be used in the Profile download process. They SHALL be deleted on the eUICC latest at the end of the process.

The process of Profile creation is out of scope of this specification; however, the Operator MAY request the SM-DP+ to create multiple Profiles in advance. In this case the SM-DP+ SHALL create the Profiles in bulk, protect them using the random key mode, and store the resulting PPPs for later use.

2.5.4 Bound Profile Package

The Bound Profile Package (BPP) is generated by the SM-DP+, within the Profile Package Binding function. The purpose of this operation is to link a Protected Profile Package to a particular eUICC. This is done within a key agreement between the eUICC and the SM-DP+. See download and installation procedure (section 3.1.3).

The BPP comprises a sequence of TLV commands (in this order):

- TLV command for Key agreement in clear.
- Set of SCP03t payload TLVs (tag '87') containing TLV commands for ConfigureISDP
- Set of SCP03t payload TLVs (tag '88') containing TLV command for StoreMetadata
- Set of optional SCP03t payload TLVs (tag '87') containing TLV command for 'Profile Protection keys'
- Followed by the SCP03t payload TLVs (tag '86') of the PPP

The encryption counter for ICV calculation is incremented each time a TLV with tag '86', '87' or '88' is received.

The data structure of the Bound Profile Package is as follows:

```
BoundProfilePackage ::= [54] SEQUENCE { -- Tag 'BF36'
   initialiseSecureChannelRequest [35] InitialiseSecureChannelRequest, -- Tag
'BF23'
   firstSequenceOf87 [0] SEQUENCE OF [7] OCTET STRING, -- sequence of '87' TLVs
   sequenceOf88 [1] SEQUENCE OF [8] OCTET STRING, -- sequence of '88' TLVs
   secondSequenceOf87 [2] SEQUENCE OF [7] OCTET STRING OPTIONAL, -- sequence of
'87' TLVs
   sequenceOf86 [3] SEQUENCE OF [6] OCTET STRING -- sequence of '86' TLVs
}
```

The following table describes the various sequences of '86', '87' and '88' TLV

Tag	Length	Value Description	MOC
'A0'	Var.	firstSequenceOf87	M

		'87'	Var.	SCP03t segment containing ConfigureISDP, protected with session keys resulting from the key agreement (S-ENC, S-CMAC) (section 2.6.4) Content: TLV for "ES8+.ConfigureISDP" function (section 5.5.2)	M
'A1'	Var	sequenceOf88			M
		'88'	Var.	SCP03t segment containing StoreMetadata, MAC protected with session keys resulting from the key agreement (S-CMAC) (See section 2.6.4) (i.e. not encrypted). Content: TLV for "ES8+.StoreMetadata" function (section 5.5.3)	M
		'88'	Var.	SCP03t segment containing the remainder of StoreMetadata if one '88' TLV is not able to contain the whole data structure.	C
'A2'	Var.	secondSequenceOf87 SHALL be absent if no content			C
		'87'	Var.	SCP03t segment containing the Profile Protection Keys, protected with session keys resulting from the key agreement (S-ENC, S-CMAC) (section 2.6.4). Content: TLV for "ES8+.ReplaceSessionKeys" function (section 5.5.4)	O
'A3'	Var.	sequenceOf86			M
		'86'	Var.	SCP03t payload, segment b1 protected with Profile Protection keys (PPK-ENC, PPK-MAC) or with session keys resulting from the key agreement (S-ENC, S-CMAC) (section 2.6.4).	M
		'86'	Var.	Subsequent SCP03t payload, segment b2…bn	O

Table 4: Profile Installation sequences of TLV

2.5.4.1 Description of 'InitialiseSecureChannel' Block

This block comprises the TLVs for opening a remote personalisation session with eUICC, including key agreement.

These TLVs, part of the function "ES8+.InitialiseSecureChannel", are listed and described in section 5.5.1. These TLVs SHALL not be encrypted. Integrity and authenticity are ensured by the signatures.

The execution of this function by the eUICC will result in the generation of the SCP03t session keys, denoted S-ENC, S-MAC and initial MAC chaining value, that will be used by the SM-DP+ to protect subsequent TLVs.

2.5.4.2 Description of 'ConfigureISDP' Block

This block comprises one TLV for ISD-P creation and configuration.

The TLV, part of the function "ES8+.ConfigureISDP", is listed and described in section 5.5.2. This TLV SHALL be encrypted and MACed with the SCP03t session keys.

2.5.4.3 Description of 'StoreMetadata' Block

This block comprises one or two TLV(s) containing the Profile Metadata.

The TLV(s), part of the function "ES8+.StoreMetadata", are listed and described in section 5.5.3. These TLV(s) SHALL be MACed only with the SCP03t session keys.

2.5.4.4 Description of 'Profile Protection keys' Block

The 'Profile Protection keys' block contains the function "ES8+.ReplaceSessionKeys" to replace the session S-ENC and S-MAC keys resulting from key agreement, by the keys used for protecting the Protected Profile Package, PPK-ENC and PPK_MAC.

This function is protected by SCP03t with the S-ENC and S-MAC keys resulting from the key agreement.

This block is optional depending on the mode selected by the SM-DP+ to protect the Profile Package (section 2.5.3).

2.5.5 Segmented Bound Profile Package

The Segmented Bound Profile Package (SBPP) is generated by the LPAd, to transfer the Bound Profile Package to the eUICC using the local interface ES10b.

The segmentation SHALL be done according to the structure of the Bound Profile Package:

- Tag and length fields of the BoundProfilePackage TLV plus the initialiseSecureChannelRequest TLV
- Tag and length fields of the first sequenceOf87 TLV plus the first '87' TLV
- Tag and length fields of the sequenceOf88 TLV
- Each of the '88' TLVs
- Tag and length fields of the sequenceOf87 TLV plus the first '87' TLV
- Tag and length fields of the sequenceOf86 TLV
- Each of the '86' TLVs

Each segment of this list that is up to 255 bytes is transported in one APDU. Larger TLVs are sent in blocks of 255 bytes for the first blocks and a last block that MAY be shorter.

2.5.6 Profile Installation Result

The Profile Installation Result contains the following data:

- Notification Metadata: The Notification Metadata includes:
    - Sequence Number
    - Profile Management Operation
    - Recipient Address
    - ICCID (Not provided if the Notification reports an error that has happened before ICCID was known by the eUICC, otherwise it SHALL be present)
- Transaction ID: The Transaction Identifier given to the eUICC during the Profile "Download and Installation" procedure (section 3.1.3).
- Final Result: provides the final Profile installation status.

- SM-DP+ OID: The SM-DP+ OID as contained in CERT.DPpb.ECDSA used during the profile download and installation procedure (section 3.1.3).
- eUICC signature: A signature created by the eUICC ensuring the authenticity and the integrity of the Profile Installation Result.

The Profile Installation Result SHALL be created by the eUICC after the execution of the last TLVs of the BPP, or right after the first BPP's TLV executed with error. The notificationAddress in the profileInstallationResultData SHALL be set to the serverAddress provided in "ES10b.AuthenticateServer".

The Profile Installation Result is returned at the end of processing the BPP.

Until the Profile Download and Installation process is completed, no Result is available for the LPA.

The Profile Installation Result SHALL be kept by the eUICC (which can hold one or several Profile Installation Results) until explicitly deleted by the LPA, after successfully delivered to the SM-DP+. Before being deleted the Profile Installation Result(s) MAY be retrieved at any time by the LPA.

When the eUICC needs to store a new Profile Installation Result, if there is not enough room the eUICC SHALL delete one or more of the previously stored Profile Installation Results in order of their Sequence Number, beginning with the lowest.

The Profile Installation Result SHALL be encoded in the ASN.1 data object as shown below. It SHALL include an eUICC signature data object computed as defined in section 2.6.7.2, using the eUICC private key SK.EUICC.ECDSA across the data object ProfileInstallationResultData (tag 'BF 27').

```
-- Definition of Profile Installation Result
ProfileInstallationResult ::= [55] SEQUENCE { -- Tag 'BF37'
   profileInstallationResultData [39] ProfileInstallationResultData,
   euiccSignPIR EuiccSignPIR
}

ProfileInstallationResultData ::= [39] SEQUENCE { -- Tag 'BF27'
   transactionId[0] TransactionId, -- The TransactionID generated by the SM-DP+
   notificationMetadata[47] NotificationMetadata,
   smdpOid OBJECT IDENTIFIER, -- SM-DP+ OID (same value as in CERT.DPpb.ECDSA)
   finalResult [2] CHOICE {
      successResult SuccessResult,
      errorResult ErrorResult
   }
}

EuiccSignPIR [APPLICATION 55] OCTET STRING -- Tag '5F37', eUICC's signature

SuccessResult ::= SEQUENCE {
   aid [APPLICATION 15] OCTET STRING (SIZE (5..16)), -- AID of ISD-P
   simaResponse OCTET STRING -- contains (multiple) 'EUICCResponse' as defined in
[5]
}

ErrorResult ::= SEQUENCE {
   bppCommandId BppCommandId,
   errorReason ErrorReason,
```

```
   simaResponse OCTET STRING OPTIONAL -- contains (multiple) 'EUICCResponse' as
defined in [5]
}

BppCommandId ::= INTEGER {initialiseSecureChannel(0), configureISDP(1),
storeMetadata(2), storeMetadata2(3), replaceSessionKeys(4), loadProfileElements(5)}

ErrorReason ::= INTEGER {
   incorrectInputValues(1),
   invalidSignature(2),
   invalidTransactionId(3),
   unsupportedCrtValues(4),
   unsupportedRemoteOperationType(5),
   unsupportedProfileClass(6),
   scp03tStructureError(7),
   scp03tSecurityError(8),
   installFailedDueToIccidAlreadyExistsOnEuicc(9),
   installFailedDueToInsufficientMemoryForProfile(10),
   installFailedDueToInterruption(11),
   installFailedDueToPEProcessingError (12),
   installFailedDueToDataMismatch(13),
   testProfileInstallFailedDueToInvalidNaaKey(14),
   pprNotAllowed(15),
   installFailedDueToUnknownError(127)
}
```

2.5.6.1 Profile Installation Result errors

`ErrorReason` data object contained in `ErrorResult` data object depends on the function that generated an error during processing of the BoundProfilePackage. The following table details authorised combinations:

`ErrorReason` in `ErrorResult`	ES8+ function				
	Initialise Secure Channel	Configure ISDP	Store Metadata	Replace Session Keys	Load Profile Elements
`incorrectInputValues(1)`	✓	✓	✓	✓	✓
`invalidSignature(2)`	✓				
`invalidTransactionId(3)`	✓				
`unsupportedCrtValues(4)`	✓				
`unsupportedRemoteOperationType(5)`	✓				
`unsupportedProfileClass(6)`			✓		
`scp03tStructureError(7)`		✓	✓	✓	✓
`scp03tSecurityError(8)`		✓	✓	✓	✓
`installFailedDueToIccidAlreadyExistsOnEuicc(9)`			✓		
`installFailedDueToInsufficientMemoryForProfile(10)`		✓	✓		✓
`installFailedDueToInterruption(11)`	✓	✓	✓	✓	✓
`installFailedDueToPEProcessingError(12)`					✓
`installFailedDueToDataMismatch(13)`					✓
`testProfileInstallFailedDueToInvalidNaaKey(14)`					✓
`pprNotAllowed(15)`			✓		
`installFailedDueToUnknownError(127)`	✓	✓	✓	✓	✓

Table 4a: Authorised combinations of ES8+ Errors

If an error is generated during the processing of a ProfileElement of the SIMalliance profile package a corresponding `ErrorReason` SHALL be set in the Profile Installation Result according to the following table:

SIMalliance error *status* in *PEStatus*	`ErrorReason` in `ErrorResult`
`pe-not-supported(1)`	`installFailedDueToPEProcessingError(12)`
`memory-failure(2)`	`installFailedDueToPEProcessingError(12)`
`bad-values(3)`	`installFailedDueToPEProcessingError(12)`
`not-enough-memory(4)`	`installFailedDueToInsufficientMemoryForProfile(10)`
`invalid-request-format(5)`	`installFailedDueToPEProcessingError(12)`
`invalid-parameter(6)`	`installFailedDueToPEProcessingError(12)`
`runtime-not-supported(7)`	`installFailedDueToPEProcessingError(12)`
`lib-not-supported(8)`	`installFailedDueToPEProcessingError(12)`
`template-not-supported(9)`	`installFailedDueToPEProcessingError(12)`
`feature-not-supported(10)`	`installFailedDueToPEProcessingError(12)`
`unsupported-profile-version(31)`	`installFailedDueToPEProcessingError(12)`
`Other SIMalliance status codes except (0)`	`installFailedDueToPEProcessingError(12)`
`Any SIMalliance status code`	`installFailedDueToDataMismatch(13), testProfileInstallFailedDueToInvalidNaaKey(14)`

Table 4b: SIMalliance error mapping to ErrorReason

2.6 Security Overview

This section provides an overview of the overall ecosystem security features.

2.6.1 Certification of the Entities

According to SGP.21 [4]:

- The eUICC SHALL be certified according to the GSMA eUICC Protection Profile for RSP

NOTE: This document does not exist at the time of writing, reference to it will be added once the document will be available.

- The EUM, the SM-DP+ and the SM-DS SHALL be GSMA SAS certified

The Device and the LPA SHALL comply with the security features defined in this specification.

2.6.2 Remote Secure Communication

The RSP ecosystem relies on remote secure communication to achieve function execution requests and data exchanges. Any of the remote secure communication defined for RSP SHALL follow the hereunder rules.

Mutual authentication:

- The Server (the entity providing the function, e.g. SM-DP+) SHALL be authenticated first by the Client (the entity requesting the function). Authentication SHALL include the verification of a valid Server Certificate signed by a GSMA CI (section 4.5.2).
- The Client SHALL be authenticated by the Server in a second step. In case the Client is the eUICC, authentication SHALL include the verification of a valid eUICC and EUM Certificate signed by a GSMA CI (section 4.5.2).

Data privacy:

- The eUICC, as a Client, SHALL not reveal any private data to an unauthenticated Server.
- The eUICC, as a Client, SHALL not generate any signed material before having authenticated itself to the Server.

Communication protection:

- Once mutually authenticated, the remote entities SHALL negotiate a common cryptographic suite for further communication.
- The communication SHALL be origin authenticated, as well as integrity and confidentiality protected.
- When applicable, the secure communication SHALL be established using Forward Secrecy.

Authorisation:

- On the basis of authentication, the Server SHALL always check that the requesting Client is authorised before delivering the requested function execution.

2.6.3 Public Key Infrastructure

General security of the RSP ecosystem is based on a Public Key Infrastructure (PKI).

Any Certificate defined in this specification has a validation chain whose root is a GSMA CI Certificate (section 4.5.2).

Certificates MAY be revoked; Revocation status are managed and made available by GSMA CI (section 4.6).

Certificates are used for authentication of the belonging entity via signature created with associated private key. This signature SHALL follow ECDSA as defined in section 2.6.7.2.

2.6.4 Protocol for Profile Protection and eUICC Binding

The Profile is protected by security mechanisms which are based on SCP11a as specified by the GlobalPlatform Card Specification Amendment F [13].

This section describes the differences between SCP11a and the Protocol for Profile Protection. The SM-DP+ plays the role of the Off Card Entity (OCE) specified in GlobalPlatform Card Specification Amendment F [13].

- The mutual authentication defined for SCP11a is modified: Whereas in SCP11a, authentication is achieved by a shared secret calculated from static key pairs being

fed into the generation of the session keys, in the Protocol for Profile Protection signatures of each side are used to authenticate to the other side. ECKA certificates are not used for mutual authentication.

NOTE: Using ECDSA signature keys for the authentication during the key establishment allows the same keys also to be used to sign other content, e.g. notifications. Only one certificate per entity is required in this case.

- Ephemeral keys are renamed to one-time keys in this specification, as they MAY live longer and are stored in non-volatile memory. With respect to Forward Secrecy, they serve the same purpose.
- The ISD-R SHALL not persistently store any SM-DP+ public key (GlobalPlatform Card Specification Amendment F [13] sections 4.1 and 4.2).
- Establishment of the session keys SHALL use only the shared secret generated from the one-time key pairs.
- The first TLV(s) following the data for key establishment are protected with the session keys generated in the key agreement. MACing and encryption is done as specified for SCP03t in SGP.02 [2] (NOTE below).
- The data contains the ISD-P configuration data. When this is processed by the eUICC, the ISD-P is created.
- Optionally, the session keys can be replaced by the Profile protection keys. The Profile protection keys are themselves secured by the session keys, Subsequent data is exchanged as TLVs as specified for SCP03t in SGP.02 [2] (NOTE below), protected by the Profile protection keys.

NOTE: This specification only reuses parts of SCP03t as specified in SGP.02 [2]. No INITIALIZE UPDATE or EXTERNAL AUTHENTICATE command TLVs are required. Instead, this specification introduces new functions for key agreement and ISD-P creation in tag '87' and uses the MACed and encrypted data TLVs (tag '86') from SGP.02 [2].

2.6.5 Key Length and Hashing Functions

Except if stated otherwise, this specification follows the recommendations defined in SGP.02 [2] regarding key length.

Algorithm	Minimum Key Length
Symmetric (AES)	128 bits, block size of 128 bits
Elliptic curve	256 bits
Hashing for Digital signatures and hash-only applications	SHA-256
Hashing for HMAC, Key Derivation Functions and Random Number Generation	SHA-256

TLS is used in RSP to provide a first level transport layer.

2.6.6 TLS Requirements

RSP mandates use of TLS v1.2 as defined in RFC 5246 [16] as the minimal version for TLS connection.

To fulfil the security requirements of the previous section, the client SHALL offer sha256/ecdsa in the "supported_signature_algorithms" of TLS 1.2 [16] and the server SHALL select this hash/signature pair.

TLS v1.3 [42] is not yet available at this date (July 2016). But some stable directions are already given, and any static RSA and DH key exchange has been forbidden.

In order to anticipate recommendation of TLS v1.3 [42], RSP requires that RSP Servers (e.g. SM-DP+) SHALL support at least these cipher suites:

```
(1) TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256

(2) TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
```

Cipher suites (1) and (2) are defined in RFC 5289 [46].

Clients SHALL support at least one of the cipher suites defined above. For the purpose of RSP, one of these cipher suites SHALL be used.

Clients SHALL support at least one set of elliptic curve parameters defined in 2.6.7.1.

An RSP Server SHALL support all three sets of elliptic curve parameters defined in 2.6.7.1.

2.6.7 Elliptic Curves Algorithms

2.6.7.1 Domain Parameters

In order to facilitate interoperability, this specification is limited to the three following curves (similar as SGP.02 [2]):

- NIST P-256, defined in Digital Signature Standard [29] (recommended by NIST)
- brainpoolP256r1, defined in RFC 5639 [18] (recommended by BSI)
- FRP256V1, defined in ANSSI ECC [20] (recommended by ANSSI)

An eUICC SHALL have at least two set of elliptic curve parameters preloaded by the EUM during eUICC manufacturing.

An RSP Server SHALL support all three sets of elliptic curve parameters.

The capabilities of each party are exchanged during the common mutual authentication procedure. The RSP Server SHALL select the most suitable GSMA CI certificates supported for signature verification and signature generation, respectively. This selection SHALL be based on the euiccCiPKIdListForVerification and the euiccCIPKIdListForSigning, which list the CI public keys the eUICC supports in descending priority. If an RSP server does not have any other priorities defined, it SHALL follow the priorities given by the eUICC. An SM-DP+ MAY follow the priorities defined by a Profile Owner. The curve parameters are identified by the selected certificates.

In the event that no common GSMA CI is supported by the RSP Server and the eUICC, the procedure SHALL be stopped.

2.6.7.2 ECDSA

A signature based on ECDSA SHALL be computed as defined in GlobalPlatform Card Specification Amendment E [12] with one of the domain parameters defined above in section 2.6.7.1 and key length and HASH function recommended above in section 2.6.5.

When applied to an ASN.1 data object, the signature SHALL be computed for the data object after encoding (i.e. in its DER representation).

2.6.7.3 ECKA

An Elliptic Curve Key Agreement Algorithm (ECKA) is used in RSP for the establishment of any session keys between the eUICC and the SM-DP+. The key agreement and key derivation process is detailed in Annex G.

2.7 Certificate Revocation

The following Certificates MAY be revoked at any time:

- GSMA CI Certificate (CERT.CI.ECDSA)
- EUM Certificate (CERT.EUM.ECDSA)
- SM-DP+ Certificates (CERT.DPauth.ECDSA, CERT.DPpb.ECDSA)
- SM-DP+ TLS Certificate (CERT.DP.TLS)
- SM-DS Certificate (CERT.DSauth.ECDSA)
- SM-DS TLS Certificate (CERT.DS.TLS)

Because of their potential number, EUICC Certificates (CERT.EUICC.ECDSA) are not revoked individually. Also, it is unlikely that an individual EUICC would be compromised. It is instead more probable that an eUICC model or an entire eUICC production batch would be declared as compromised. This approach is reflected by revoking the EUM Certificate attached to the production of the particular eUICC model or batch.

As a consequence, it is up to the EUM to consider using distinct Certificates for distinct eUICC models or production batches. This is out of the scope of this specification.

Each GSMA CI SHALL manage the revocation status for the Certificates it has issued. A revoked Certificate SHALL not be automatically renewed. Renewal SHALL be upon the GSMA Certification authority agreement.

Revocation status is made available by each GSMA CI by the mean of a Certificate Revocation List (CRL) that SHALL be made available to any RSP entity (section 4.6).

GSMA CI MAY provide additional means to make available Certificate revocation status (e.g. OCSP as defined in RFC 6960 [44]).

The Certificate revocation management is optional for the eUICC. If the capability is not supported in the eUICC, the LPA SHALL not pass the CRL to the eUICC.

2.8 ASN.1

The description of some data objects in this specification is based on ASN.1 specified in ITU-T X.680 [48] and encoded in TLV structures using DER (Distinguished Encoding Rule) encoding as specified in ITU-T X.690 [49]. This provides a flexible description of those data objects.

The Remote SIM Provisioning format is defined in a single, self-contained, ASN.1 definition module called `RSPDefinitions`, with an ISO Object Identifier in the GSMA namespace.

```
RSPDefinitions {joint-iso-itu-t(2) international-organizations(23) gsma(146) rsp(1)
spec-version(1) version-two(2)}
DEFINITIONS
AUTOMATIC TAGS
EXTENSIBILITY IMPLIED ::=
BEGIN
```

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. In some cases the eUICC is even mandated to report undefined tags, see e.g. sections 3.1.5 and 5.7.6.

2.8.1 Common ASN.1 data types

NOTE: Other common data types may be added here in future versions.

2.8.1.1 Data type: PprIds

The data type `PprIds` codes the identifiers for Profile Policy Rules defined in this document.

```
PprIds ::= BIT STRING {-- Definition of Profile Policy Rules identifiers
   pprUpdateControl(0), -- defines how to update PPRs via ES6
   ppr1(1), -- Indicator for PPR1 'Disabling of this Profile is not allowed'
   ppr2(2) -- Indicator for PPR2 'Deletion of this Profile is not allowed'
}
```

For pprX: a bit set to '1' indicates that the corresponding PPR is set.

Further versions of this specification MAY introduce new Profile Policy Rule identifiers

2.8.1.2 Data type: OperatorId

The data type `OperatorId` contains the identification of an Operator. This type is used to identify a Profile Owner.

```
OperatorId ::= SEQUENCE {
   mccMnc OCTET STRING (SIZE(3)), -- MCC&MNC coded as 3GPP TS 24.008
   gid1 OCTET STRING OPTIONAL, -- referring to content of EF GID1 (file identifier
'6F3E') in 3GPP TS 31.102 [54]
   gid2 OCTET STRING OPTIONAL  -- referring to content of EF GID2 (file identifier
'6F3F') in 3GPP TS 31.102 [54]
}
```

Coding of mccMnc: contains MCC (3 digits) and MNC (2 or 3 digits) on 3 bytes coded as in 3GPP TS 24.008 [32]. For instance, an Operator identified by 246 for the MCC and 81 for the MNC will be coded as bytes 1 to 3: '42' 'F6' '18'.

Coding of `gid1` and `gid2`: both are optional. Content SHALL be coded as defined in 3GPP TS 31.102 [54].

In case the Profile contains multiple USIM applications, the `OperatorId` SHALL reflect the values of one of the USIM applications.

NOTE: Additional mechanism for identifying MVNO/Service Providers is for further study.

2.8.2 ASN.1 data type UTF8String

The size limits for UTF-8 strings apply to the number of UTF-8 characters, each coded on 1 to 4 bytes, see ISO/IEC 10646 [59]. Thus the length of the TLV object counted in bytes can be up to 4 times the number of characters.

The eUICC is not mandated to analyse the character structure of UTF-8 strings provided in a command. The LPA SHOULD take care that the eUICC may provide a string with a number of characters exceeding the ASN.1 size limit if such a string was previously stored.

2.9 Profile Policy Management

Profile Policy Management provides mechanisms by which Profile Owners can enforce the conditions of use under which services are provided.

Profile Policy Management comprises three main elements:

- Profile Policy Rules (PPR)
- Rules Authorisation Table (RAT)
- Profile Policy Enabler (PPE)

More details are provided in the next sub sections.

2.9.1 Profile Policy Rules

The Profile Policy Rules (PPRs) are defined by the Profile Owners and set by the SM-DP+ in the Profile Metadata. They are also accessible by the LPA for verification or display to the End User.

A Profile MAY have zero or more Profile Policy Rules.

A Test Profile SHOULD NOT contain any Profile Policy Rules.

When installed on the eUICC, Profile Policy Rules SHALL be contained in the associated Profile.

The following Profile Policy Rules are defined in this version of the specification:

- (PPR1) 'Disabling of this Profile is not allowed'
- (PPR2) 'Deletion of this Profile is not allowed'

The coding of PPRs is given in section 2.8.1.1.

2.9.2 Rules Authorisation Table (RAT)

The Rules Authorisation Table (RAT) contains the description of the acceptable set of PPRs that can be set in a Profile. The RAT is defined at eUICC platform level and is used by the Profile Policy Enabler (PPE) and the LPA to determine whether or not a Profile that contains PPRs is authorised and can be installed on the eUICC.

The RAT is initialised at eUICC manufacturing time or during the initial Device setup provided that there is no installed Operational Profile. The OEM or EUM is responsible for setting the content of the RAT.

The RAT SHALL not be affected by the ES10b.eUICCMemoryReset function (section 5.7.19).

2.9.2.1 Profile Policy Authorisation Rules (PPAR)

The RAT contains a list of Profile Policy Authorisation Rules (PPAR).

A PPAR is composed of the following information:

Data	Description
Profile Policy Rule Identifier	Identifies the Profile Policy Rules to which this PPAR applies. This field SHALL contain one or several PPR(s) being set as defined in 2.8.1.1.
Allowed Operators	List of Profile Owners, as defined in section 2.8.1.2, allowed to use this PPR. Wildcards can be used to indicate that all, or a set of, Profile Owners are allowed. See below.
End User Consent Required	Indicates if the related PPR needs the End User Consent for the Profile to be installed (true/false).

Table 5: PPAR description

The RAT MAY contain zero or more PPAR(s) related to a particular PPR. The order of the PPARs in the RAT is significant (see below).

'Allowed Operators' field

The 'Allowed Operators' field contains the identifier(s) of the Profile Owner(s) (explicitly listed or matching a wild card) allowed to use the related PPR. It SHALL be compared against the `profileOwner` field of the Metadata of the Profile.

Any of the digits of the `mccMnc` data object can be wildcard-ed by setting the appropriate nibble to 'E'.

If present in the PPAR, a non-empty `gid1` or `gid2` value SHALL exactly match the corresponding value in the `profileOwner` field.

The `gid1` or `gid2` data objects can be wildcard-ed by setting an empty value (length zero).

An omitted `gid1` or `gid2` value in the PPAR SHALL only match a `profileOwner` field where the corresponding `gid1` or `gid2` value is absent.

NOTE: a PPR MAY be allowed for all Profile Owners by setting the 'Allowed Operators' field with a unique OperatorId having the `mccMnc` field value set to 'EEEEEE' and `gid1` and `gid2` data objects set with an empty value (length zero).

A PPR MAY be 'forbidden' for all Profile Owners by not defining any related PPAR.

Case where multiple PPARs are defined for a PPR:

A PPR is allowed for a Profile Owner whose identifier appears in the 'Allowed Operators' field (explicitly listed or matching a wild card) in one of the related PPARs.

'End User Consent required' field
When set to 'true', it indicates that for all Profile Owners allowed by the 'Allowed Operators' field the LPA SHALL get the End User Consent for the related PPR to install the Profile.

When set to 'false', it indicates that this End User Consent is not mandatory.

Case where multiple PPARs are defined for a PPR:

When a Profile Owner is allowed in several PPARs (explicitly listed or matching a wild card), the 'End User Consent required' field value of the first of these PPARs SHALL be used.

Example of RAT configuration (for illustration only and not intended to represent a real case):

PPRid	Allowed Operators	End User Consent Required
PPR1	OP-A	false
PPR2	OP-B	false
PPR1, PPR2	*	true

The '*' in the 'Allowed Operators' field denotes a PPR that is allowed for any Profile Owner; and if there is no PPAR for a particular PPR, then that PPR is forbidden.

With this configuration, Operator OP-A:

- can use PPR1 without the End User consent
- can use PPR2 with the End User consent

With this configuration, Operator OP-B:

- can use PPR1 with the End User consent

- can use PPR2 without the End User consent

With this configuration, any other Profile Owner:

- can use PPR1 and PPR2 with the End User consent

2.9.2.2 Notable RAT configurations

'All PPRs allowed for all Profile Owners, End User Consent required'

PPRid	Allowed Operators	End User Consent Required
PPR1, PPR2	*	true

'All PPRs forbidden for all Profile Owners'

PPRid	Allowed Operators	End User Consent Required
<no entry>		

2.9.2.3 RAT configuration for products compliant with SGP.22 v2.1

While this specification describes the generic RAT mechanism, for products compliant with SGP.22 v2.1 the RAT SHALL be configured according to SGP.21 Annex H and in compliance with local regulatory requirements.

2.9.2.4 LPA verification

During the Profile Download and Installation procedure (see section 3.1.3), the LPA SHALL verify that the PPRs defined in the Profile to install can be set by the Profile Owner, and if an End User Consent is required.

The figure below describes the process to determine if all PPRs of a Profile can be set by the Profile Owner, and if an End User consent is required, according to RAT configuration.

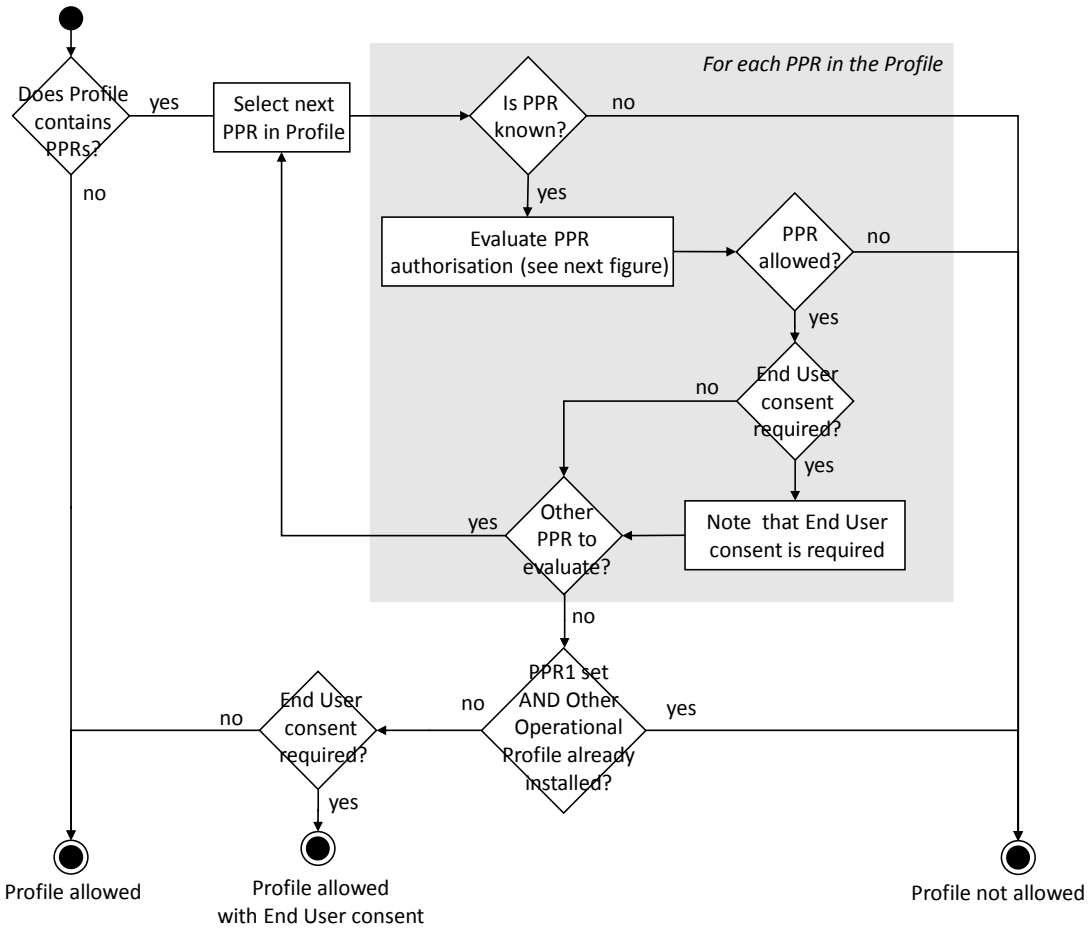


Figure 5: Profile's PPRs verification by LPA

The figure below describes the process to determine if a particular PPR can be set by the Profile Owner, and if an End User consent is required, according to its related PPAR(s) configuration.

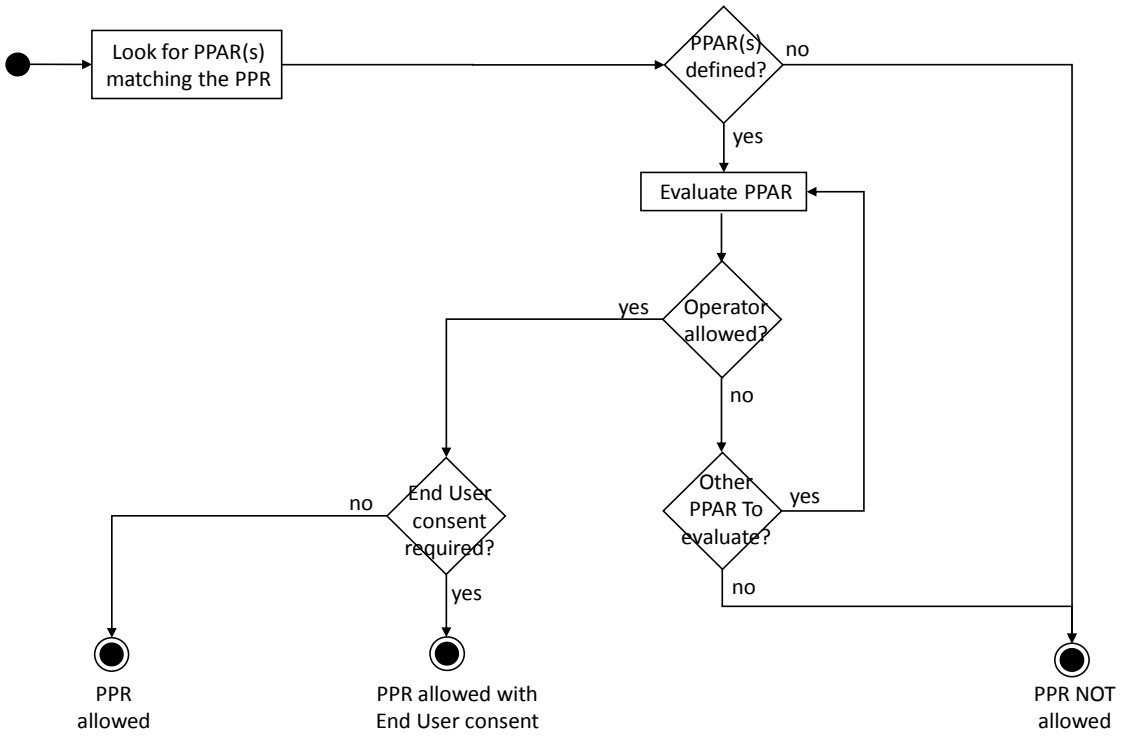


Figure 6: Particular PPR verification by the LPA

2.9.3 Profile Policy Enabler

The Profile Policy Enabler on the eUICC has two functions:

- Verification that a Profile containing PPRs is authorised by the RAT.
- Enforcement of the PPRs of a Profile.

2.9.3.1 PPRs Verification: Profile installation time

At Profile installation time the Profile Policy Enabler SHALL verify each of the PPRs as described below, to determine if it allows the Profile installation to continue. If the verification results in the Profile not being allowed, then the Profile installation SHALL be rejected and a Profile Installation Result SHALL be generated and returned to the LPA.

The PPE cannot enforce that the End User consent, if any is required, is captured by the LPA. As a consequence the 'End User Consent required' field SHALL not be considered during the PPRs verification by the PPE.

The figure below describes the process that the PPE SHALL run to determine if a Profile containing PPRs can be installed on the eUICC, according to the RAT configuration.

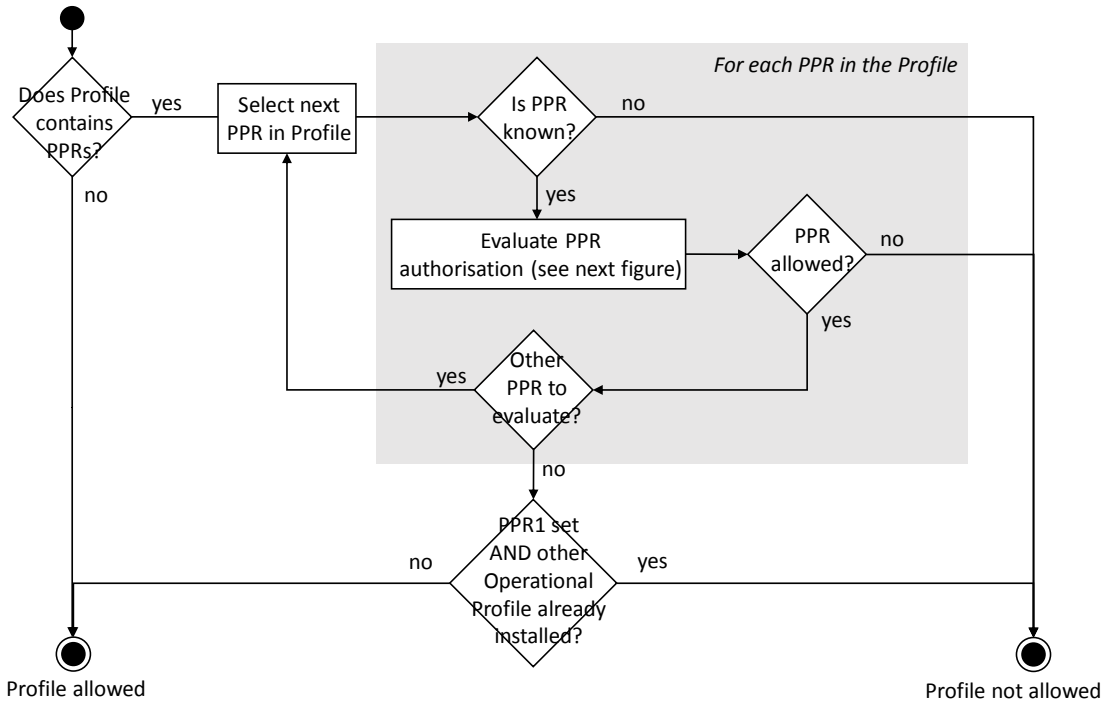


Figure 7: Profile's PPRs verification by PPE

The figure below describes the process to determine if a PPR is allowed according to its related PPAR(s) configuration.

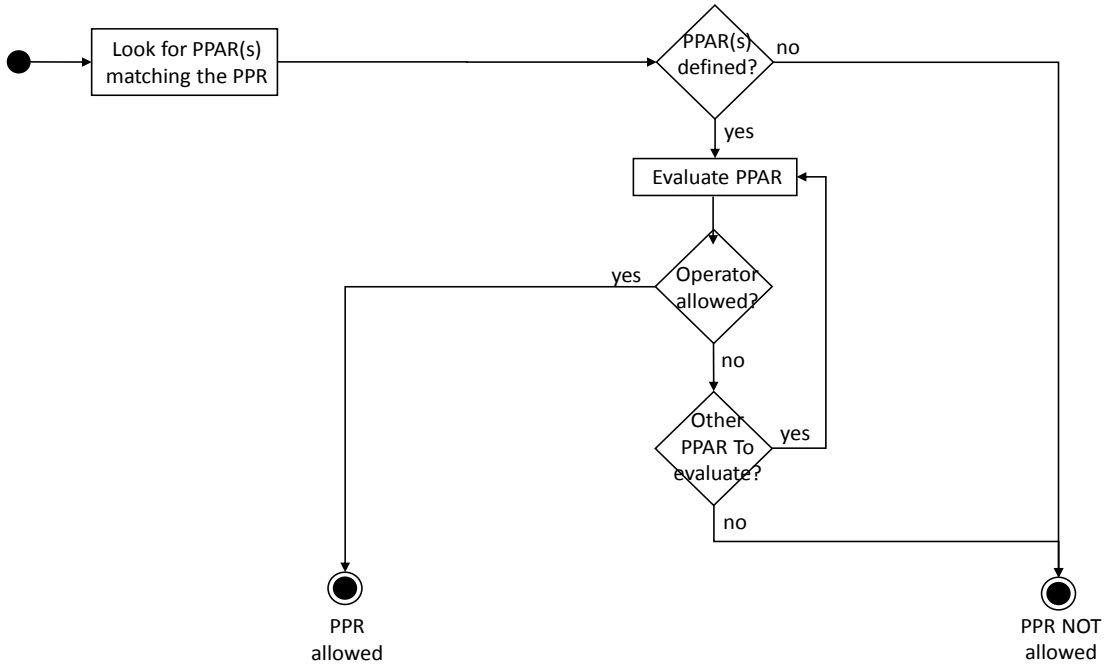


Figure 8: Particular PPR verification by the PPE

2.9.3.2 PPR Verification: PPR update after Profile is installed

A PPR in a Profile installed in the eUICC can be unset (using the ES6+.UpdateMetadata Function by the Profile Owner). The setting of a PPR in the eUICC is for further study.

2.9.3.3 PPR Enforcement

The Profile Policy Enabler SHALL enforce the PPRs of a Profile when a Local Profile Management Operation is requested upon this Profile. Each of the defined enforcement cases are described in the concerned procedures (see section 3.2 and 3.3).

2.9.3.3.1 Void

Table 6: Void

2.9.3.3.2 Enforcement involving Test Profile

When a Test Profile is requested to be enabled whereas the currently Enabled Profile has a PPR1 set, PPE SHALL not enforce this PPR1 to allow the Test Profile to be enabled.

2.9.3.3.3 Void

3 Procedures

This section specifies the procedures associated with Remote SIM Provisioning and Management of the eUICC for consumer Devices.

Some call flows illustrate the case where the LPA is in the Device (LPAd). Such call flows with an LPAe would be identical except that all ES10a, ES10b and ES10c calls become internal to the eUICC and out of scope of this specification.

3.1 Remote Provisioning

3.1.1 Profile Download Initiation

Normal Case:

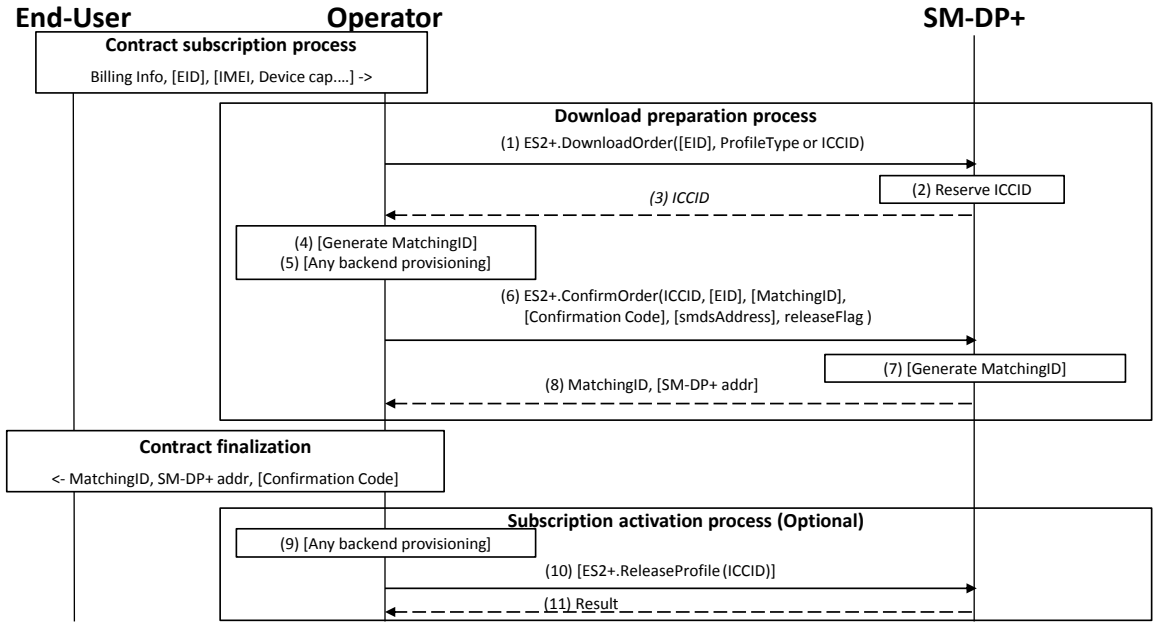


Figure 9: Profile Download Initiation

Start Conditions:

The End User has selected the Operator with which he wants to sign a contract with.

The End User MAY initiate the process:

- From any other Device (e.g. PC)
- Through a Customer Agent of the Operator
- Or any other convenient means provided by the Operator

Procedure:

The download initiation procedure consists of the following sub-processes:

A. Contract subscription process
B. Download preparation process
C. Contract finalization process
D. Subscription activation process (Optional)

NOTE: This section describes the case where these sub-process are performed in the described order. In this case, it is most likely that the download and installation procedure will happen right after this procedure. There also are cases where these sub-processes MAY be performed in different order like B -> A -> C [-> D] or B -> C -> A [-> D] (e.g. for prepaid Subscription). In these cases the download order requested from the SM-DP+ MAY remain pending for a significant amount of time.

NOTE: The following table summarizes the input data to be provided in "ES2+.DownloadOrder" and "ES2+.ConfirmOrder" functions.

ES2+ Function	Input Parameters	Profile Download Use Cases		
		Default SM-DP+	Activation Code	SM-DS
DownloadOrder (Section 5.3.1)	eid	O	O	O
	iccid	O	O	O
	profileType	C (1)	C (1)	C (1)
ConfirmOrder (Section 5.3.2)	iccid	M	M	M
	eid	C (2)	O	C (2)
	matchingId	M (3)	O (4)	O (5)
	confirmationCode	O	O	O
	smdsAddress	Not Present	Not Present	M
	releaseFlag	M	M	M
NOTE 1: Required if iccid is not present for DownloadOrder NOTE 2: Required if it is not present for DownloadOrder NOTE 3: With empty value NOTE 4: If not present, SM-DP+ generates matchingId for ActivationCodeToken NOTE 5: If not present, SM-DP+ generates matchingId. It is used as EventID				

Table 6a: "ES2+.DownloadOrder" and "ES2+.ConfirmOrder" input data

3.1.1.1 Contract Subscription Process (Informative)

The contract selection process, while being out of scope of this specification, is given as it SHALL happen prior to the Profile download and installation procedure (section 3.1.3). This process description describes the information exchanged and data that are used as input data for the Profile download and installation procedure.

This process can be performed at an Operator's Point of Sale (POS), using the Operator's web portal from a Device which is not the one onto which the Profile will be downloaded (e.g. a PC) or from a web browser on the Primary Device, or even using a companion application on the Primary Device. Any other mean defined by the Operator can also be possible as far as it provides a convenient End User experience and it provides the expected output data required for the execution of the Profile download and installation procedure.

During the execution of the process of contract Subscription, the Operator acquires the necessary information. As part of this data, the EID and IMEI of the target Device MAY be provided, and related Device capabilities MAY be acquired (e.g. based on the TAC information comprised in the IMEI). Acquisition and verification of these capabilities are out of scope of this specification. Additional information such as contract details, user details, payment details and similar are also out of scope of this specification.

If the EID and the IMEI are provided, the Operator can verify if the target Device (both eUICC and Device can be relevant for this verification) is supported, and determine the Profile Type for the target Device and the offer given to the End User. If no information about the target Device is provided, this preliminary verification cannot be performed and it will be performed during the execution of the Profile download and installation procedure (section 3.1.3). For additional info see Annex F on Profile eligibility check.

If EID and IMEI are provided and the Operator cannot provide an appropriate Profile, the process fails and stops at this point.

3.1.1.2 Download Preparation Process

1. The Operator calls the "ES2+.DownloadOrder" (section 5.3.1) function of the SM-DP+ with the relevant input data.

   'EID' is optional. If the SM-DS or the Default SM-DP+ is to be used for the Profile download, then the EID SHALL be present. One of the value 'ProfileType' or 'ICCID' SHALL be provided. If ICCID is given, the SM-DP+ SHALL verify that this ICCID is available. If 'ProfileType' is given, the SM-DP+ SHALL pick one of the related ICCID in its inventory.
   The SM-DP+ MAY optionally verify additional compatibility between the eUICC (if EID is provided) and the requested Profile Type. This verification is out of scope of this specification.

2. The SM-DP+ reserves the ICCID for this request. At this stage the SM-DP+ MAY simply pick the related Protected Profile Package from its inventory or generate and protect the Profile corresponding to this ICCID.
3. The SM-DP+ returns the acknowledged ICCID (SHALL be the same value as the received one, if any).

4. Optionally, the Operator MAY generate a MatchingID (section 4.1.1). If the Default SM-DP+ is to be used for the Profile download, then the Operator SHALL send an empty string in the MatchingID value field.
   At this stage the Operator knows the ICCID selected for this contract Subscription. It MAY perform any relevant operation on its back-end (e.g. provisioning of HLR). If an error occurs during this step, the process fails and stops at this point.
5. to 8. The Operator SHALL confirm the download order by calling the "ES2+.ConfirmOrder" (section 5.3.2) function of the SM-DP+ with the ICCID and its relevant input data.

   - If EID is available, the EID SHALL be included in the input data. If the EID was provided with previous "ES2+.DownloadOrder", the same EID SHALL be provided.
   - If generated in Step 4, the MatchingID SHALL be included in the input data and then the SM-DP+ SHALL return the acknowledged value that is the same as the received one. Otherwise, the SM-DP+ SHALL generate a MatchingID and return the generated value to the Operator. The ICCID SHALL be associated to the generated MatchingID.
   - If it is required for the End User to enter the Confirmation Code to download the Profile, the Confirmation Code SHALL be included in the input data of the "ES2+.ConfirmOrder" (section 5.3.2) function.
   - The Operator MAY send an SM-DS address, which could be the address of either the Alternative SM-DS or the Root SM-DS, to the SM-DP+ as defined in section 3.6.1. If the SM-DS address is given, the SM-DP+ SHALL perform Event Registration to the specified SM-DS. If the Default SM-DP+ is to be used, then the SM-DS address SHALL not be present.
   - If all necessary operations on Operator’s back-end provisioning has been completed by this point, releaseFlag SHALL be set to 'true' in the input data. Otherwise, releaseFlag SHALL be set to 'false' and additional "ES2+.ReleaseProfile" function SHALL be called later in Subscription activation process.

   The SM-DP+ MAY return an SM-DP+ address value. In this case the Operator SHALL use this value to generate the Activation Code; otherwise the default SM-DP+ address SHALL be used.

NOTE: If no EID is given at this stage, the Operator MAY be involved later during the download and installation procedure to determine the right 'ProfileType'/'ICCID' in case the provided 'ProfileType'/'ICCID' is not compatible with the eUICC identified by the EID once it is acquired by SM-DP+ during the download and installation procedure. See Annex F on Profile eligibility check.

3.1.1.3 Contract Finalization (Informative)

The Operator provides the End User with relevant information necessary for the Profile download.

If the Activation Code is to be used for the Profile download, the MatchingID and SM-DP+ address are provided via the Activation Code as described in section 4.1. If the optional

Confirmation Code is to be used, it is provided to the End User separately from the Activation Code.

If the SM-DS or the Default SM-DP+ is to be used for the Profile download, the Operator informs the End User of the condition that triggers the Profile download procedure, e.g., the very first boot-up and/or IP connection of the device.

3.1.1.4 Subscription Activation Process (Optional)

It is most likely that the Operator backend provisioning can be performed during the download preparation process. But if it cannot be performed, the Subscription activation process can be performed as a separate process to decouple the download preparation processes and Contract finalization process.

9. If all necessary operations on its back-end (e.g. provisioning of HLR) were not performed in Step 5, they SHALL be performed in this stage.
10. The Operator calls the ES2+.ReleaseProfile function of the SM-DP+ with ICCID to release the Profile to allow the download and installation procedure to be started by the End User. If the download and installation procedure is initiated by the End User before this function call, the download and installation procedure SHALL not be allowed and SHALL return a specific error code.
11. The SM-DP+ SHALL return the result.

3.1.2 Common Mutual Authentication Procedure

This section describes the common mutual authentication call flow that is used in various others places in this document.

In this section the following notations are used:

- SM-XX denotes either an SM-DP+ or an SM-DS.
- CERT.XXauth.ECDSA denotes either CERT.DPauth.ECDSA or CERT.DSauth.ECDSA.
- SK.XXauth.ECDSA denotes either SK.DPauth.ECDSA or SK.DSauth.ECDSA.
- CERT.XX.TLS denotes either CERT.DP.TLS or CERT.DS.TLS.
- SK.XX.TLS denotes either SK.DP.TLS or SK.DS.TLS.
- ESXX denotes either ES9+ when communicating with an SM-DP+ or an ES11 when communicating with an SM-DS.

This procedure implies the use of CERT.XXauth.ECDSA. Following this common mutual authentication procedure, if any other certificates of the SM-XX are used, e.g. the CERT.DPpb.ECDSA, these certificates SHALL have a trust chain leading to the same GSMA CI certificate as CERT.XXauth.ECDSA.

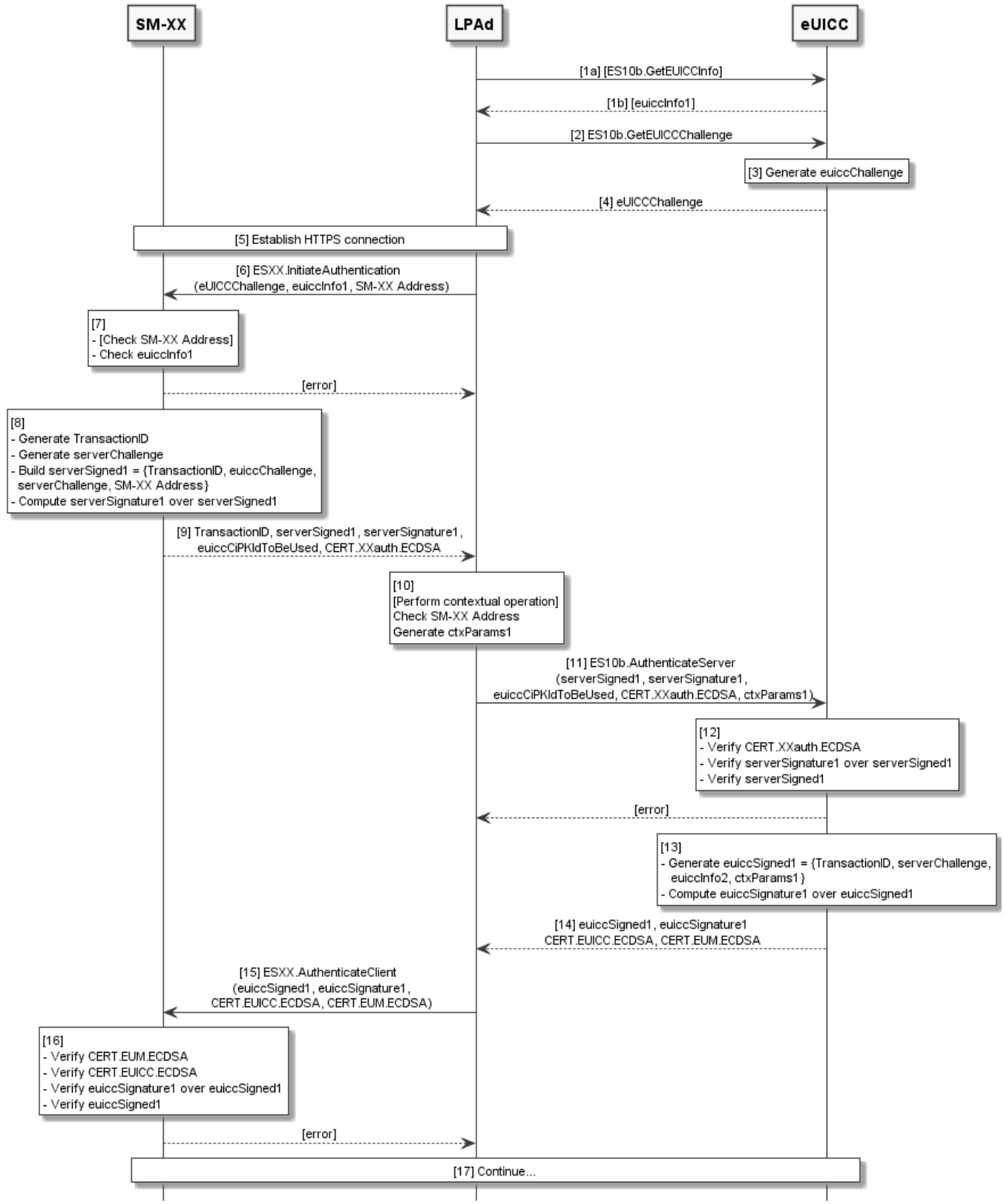


Figure 10: Common Mutual Authentication Procedure

Start conditions:

The SM-XX is provisioned with its certificate (CERT.XXauth.ECDSA), its private key (SK.XXauth.ECDSA), the CI Certificates (CERT.CI.ECDSA), its TLS Certificate (CERT.XX.TLS) and its TLS Private Key (SK.XX.TLS).

The eUICC is provisioned with its certificate (CERT.EUICC.ECDSA), its private key (SK.EUICC.ECDSA), the EUM Certificate (CERT.EUM.ECDSA) and the CI Public Key (PK.CI.ECDSA).

Procedure:

1. (a) Optionally, the LPA MAY request eUICC Information euiccInfo1 from eUICC by calling the "ES10b.GetEUICCInfo" function. This is required if the LPAd hasn't already retrieved this information.
1. (b) The eUICC returns the euiccInfo1 to the LPAd.
2. The LPAd requests an eUICC Challenge from the eUICC by calling the "ES10b. GetEUICCChallenge" function (section 5.7.7).
3. The eUICC SHALL generate an eUICC Challenge which SHALL be signed later by the SM-XX for SM-XX authentication by the eUICC.
4. The eUICC returns the eUICC Challenge to the LPAd.
5. The LPAd establishes a new HTTPS connection with the SM-XX in server authentication mode. The TLS session establishment SHALL perform a new key exchange (it SHALL not reuse keys from a previous session). During this step, the LPAd SHALL verify that CERT.XX.TLS is valid as described in section 4.5.3. If CERT.XX.TLS is invalid the procedure SHALL be stopped.

NOTE: The TLS handshake as defined in RFC 5246 [16] doesn't allow the LPAd to indicate in the "Client_Hello" message the list of GSMA CI root public keys it supports for signature verification. Therefore, in a Multiple GSMA CI environment, the SM-XX cannot provide with certainty a CERT.XX.TLS that the LPAd will be able to verify, and the TLS handshake may fail. In that case the LPAd MAY retry the TLS handshake, and the SM-XX MAY select a different CERT.XX.TLS. Alternatively, as defined in RFC 6066 [60] the LPAd and SM-XX MAY use the trusted_ca_keys extension in the "Client Hello" with IdentifierType key_sha1_hash to communicate a list of the CI root public keys that the LPAd supports.

6. The LPAd SHALL call the "ESXX.InitiateAuthentication" function (sections 5.6.1 and 5.8.1) with its input data including the euiccChallenge, euiccInfo1 and SM-XX Address. euiccInfo1 contains eUICCVerSupport (svn), euiccCiPKIdListForVerification and euiccCiPKIdListForSigning. SM-XX is the Address used by the LPAd to access the SM-XX. The way the SM-XX Address is acquired depends on the procedure where this common call flow is used.
7. Depending on the server (i.e. SM-DS or SM-DP+), the SM-XX MAY check if the SM-XX Address sent by the LPAd is valid. If the SM-XX Address is not valid, the SM-XX SHALL return an error status and the procedure SHALL be stopped.

   The SM-XX SHALL check the list of CI Public Keys that are associated to the eUICC credentials (euiccCiPKIdListForSigning). If the SM-XX does not accept any of these CI Public Keys it SHALL return an error status and the procedure SHALL be stopped.

The SM-XX SHALL check the received CI Public Key Identifier list (euiccCiPKIdListForVerification) contained in the euiccInfo1. If it cannot provide a CERT.XXauth.ECDSA signed by a CI Public Key supported by the eUICC, it SHALL return an error status and the procedure SHALL be stopped.

8. The SM-XX SHALL perform the following:

   - Generate a TransactionID which is used to uniquely identify the RSP session and to correlate the multiple ESXX request messages that belong to the same RSP session.
   - Generate an SM-XX Challenge (serverChallenge) which SHALL be signed later by the eUICC for the eUICC authentication.
   - Select one CI Public Key among those provided within euiccCiPKIdListForSigning that is supported by the RSP Server for signature verification and indicate it in euiccCiPKIdToBeUsed.
   - Generate a serverSigned1 data structure containing the TransactionID, euiccChallenge, serverChallenge, and SM-XX Address.
   - Compute the serverSignature1 over serverSigned1 using the SK.XXauth.ECDSA.

9. The SM-XX SHALL return the TransactionID, serverSigned1, serverSignature1, euiccCiPKIdToBeUsed and CERT.XXauth.ECDSA to the LPAd.
10. The LPAd SHALL:

    - (Optional) Perform contextual operation depending on the procedure where this call flow is used.
    - Verify that the SM-XX Address returned by the SM-XX matches the SM-XX Address that the LPAd has provided in step (6). If not, the LPA SHALL inform the End User and the procedure SHALL be stopped.
    - Generate a data structure, ctxParams1, to be given to the eUICC to be included in signed data.

11. The LPAd SHALL call "ES10b.AuthenticateServer" function with input data including the serverSigned1, serverSignature1, euiccCiPKIdToBeUsed, CERT.XXauth.ECDSA and ctxParams1 generated during the previous step.
12. The eUICC SHALL verify the CERT.XXauth.ECDSA using the relevant PK.CI.ECDSA. If the eUICC doesn't have the PK.CI.ECDSA to be used for CERT.XXauth.ECDSA verification, the eUICC SHALL return the relevant error status and the procedure SHALL be stopped.

    The eUICC SHALL also check the following:

    - Verify the serverSignature1 performed over serverSigned1, in particular also making sure the serverAddress contained in serverSigned1 has not been changed.
    - Verify that euiccChallenge contained in serverSigned1 matches the one generated by the eUICC during step (3).
    - Verify that euiccCiPKIdToBeUsed is supported and related credentials are available for signing.

If any verification fails, the eUICC SHALL return a relevant error status and the procedure SHALL be stopped.
If all the verifications succeed, the SM-XX is authenticated by the eUICC.

13. The eUICC SHALL:

    - Generate the euiccSigned1 data structure containing the TransactionID, serverChallenge, eUICCInfo2 and ctxParams1.
    - Compute the euiccSignature1 over euiccSigned1 using SK.EUICC.ECDSA. When generating the euiccSignature1, the eUICC SHALL use credentials related to the euiccCiPKIdToBeUsed parameter received from the SM-XX.

14. The eUICC SHALL return the euiccSigned1, euiccSignature1, CERT.EUICC.ECDSA, and CERT.EUM.ECDSA to the LPA.
15. The LPAd SHALL call the "ESXX.AuthenticateClient" function with input data including the euiccSigned1, euiccSignature1, CERT.EUICC.ECDSA, and the CERT.EUM.ECDSA.
16. On reception of the "ESXX.AuthenticateClient" function call, the SM-XX SHALL:

    - Correlate it with the "ESXX.InitiateAuthentication" function processed in step (7), by verifying the two TransactionIDs match.
    - Verify that the CERT.EUICC.ECDSA and CERT.EUM.ECDSA are valid as described in section 4.5.2.2.
    - Verify the euiccSignature1 performed over euiccSigned1 using the PK.EUICC.ECDSA contained in the CERT.EUICC.ECDSA.
    - Verify that serverChallenge contained in euiccSigned1 matches the one generated by the SM-XX during step (7).Verify that the SVN included in the euiccInfo2 is the same as the SVN which was delivered in the step (6).

    If any verification fails, the SM-XX SHALL return a relevant error status and the procedure SHALL be stopped.

17. This common call flow SHALL be followed by additional steps depending on the procedure within which it is used.

3.1.3 Profile Download and Installation

This section describes the Profile download and installation procedure.

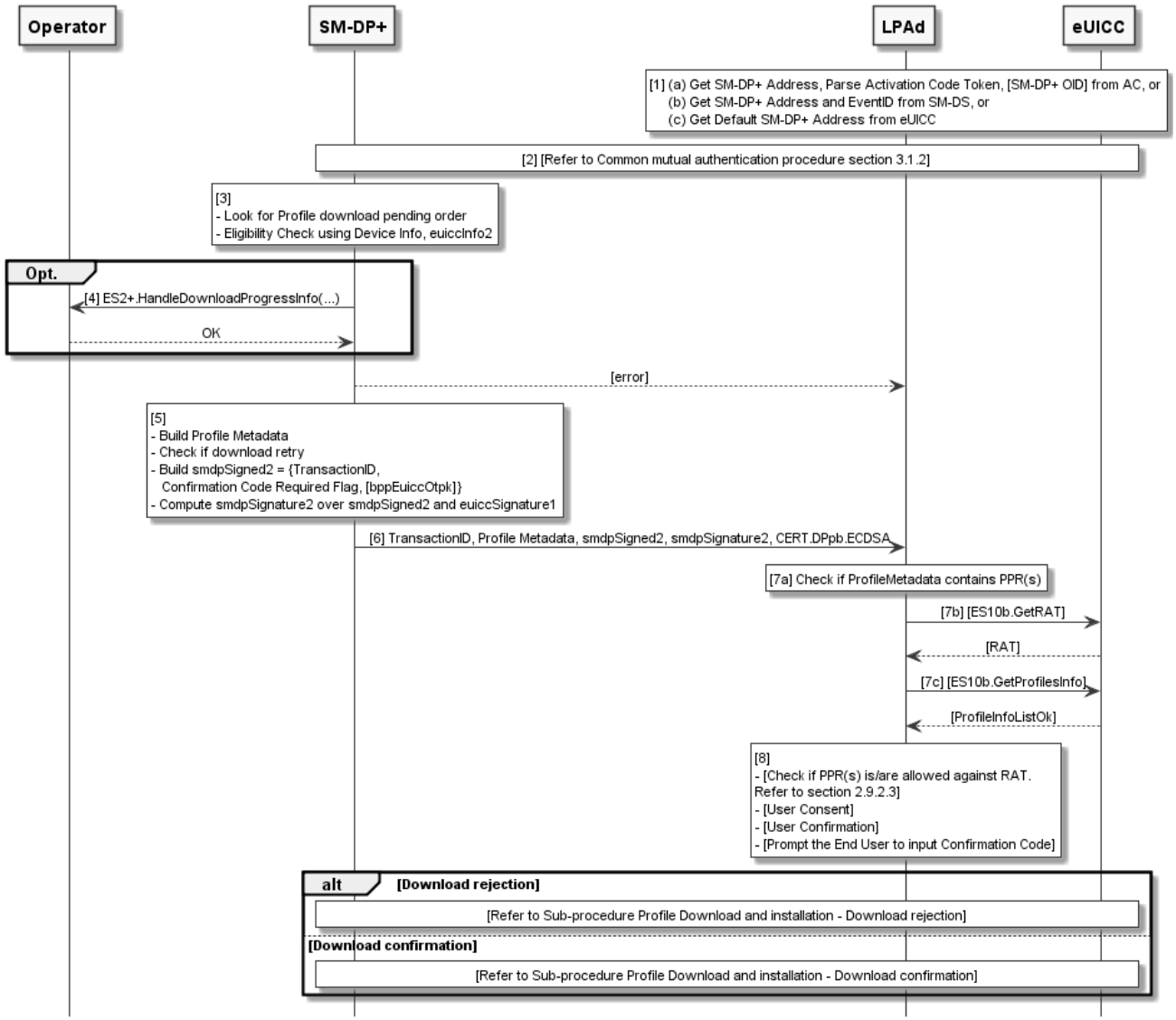


Figure 11: Profile Download and Installation

Start Conditions:

In addition to the start conditions required by the common mutual authentication procedure defined in section 3.1.2, this procedure requires the following start conditions depending on options in step 1:

- If this procedure uses an Activation Code (option a):
    - The End User has an Activation Code that is coded as described in the section 4.1.
    - The End User has entered the Activation Code to the LPAd. Depending on the Device capabilities, the LPAd SHALL support entry of the Activation Code by manual typing and QR code scanning.
- If this procedure uses an SM-DS (option b):

  - The LPAd has retrieved an SM-DP+ Address and EventID from the SM-DS.

- If this procedure uses a Default SM-DP+ (option c):
  - The LPAd has retrieved the Default SM-DP+ Address from the eUICC.

Further, for each Profile the SM-DP+ SHALL maintain a count of the number of attempts to download that Profile and a count of the number of attempts to enter the Confirmation Code during download of that Profile. The SM-DP+ SHALL limit the number of download attempts and the number of Confirmation Code attempts, respectively.

A Provisioning Profile MAY be enabled by the LPAd upon End User request for RSP operations as defined in SGP.21 [4], which SHALL include End User consent if an Operational Profile is to be disabled and if establishment of the connectivity using the currently Enabled Profile is not successful.

Finally, if there is already an installed Profile with PPR1 set, the following has occurred: The End User has been advised of this condition and has given consent for download. The LPA MAY alternatively request this consent at any later point during the download procedure.

Procedure:

1. (Optionally, i.e. for option (a)) The LPAd 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 be stopped with an error message provided by the LPAd to the End User.
2. The common mutual authentication procedure defined in section 3.1.2 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), the LPAd 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 LPAd has acquired it from the Activation Code at step (1). If the comparison fails, the LPAd SHALL inform the End User and the procedure SHALL be stopped.
   During the common mutual authentication procedure at step (10), the LPAd 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, the MatchingID value SHALL be set to Activation Code Token.
   - If an SM-DS is used, the MatchingID value SHALL be set to EventID.
   - If a Default SM-DP+ is used, the MatchingID value SHALL be set to an empty string.

3. After having successfully authenticated the eUICC at the end of the step (2) above, the SM-DP+ SHALL:

- Verify that there is a related pending Profile download order for provided the 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 (section 3.1.6).

If any of these verifications fail, the SM-DP+ SHALL return a relevant error status and the procedure SHALL be stopped.
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 for more information on Eligibility checks.

4. (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.

5. 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).
   - Return an error status to the LPAd and the procedure SHALL be stopped.

   Otherwise, the SM-DP+ SHALL:

   - Determine whether the Profile is already bound to the EID from a previous unsuccessful download attempt. If so, the SMDP+ 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.
   - 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.

6. The SM-DP+ returns the TransactionID, ProfileMetadata, smdpSigned2, smdpSignature2 and CERT.DPpb.ECDSA to the LPAd.
7. a) On reception of the SM-DP+ response, the LPAd SHALL check if the ProfileMetadata contains PPR(s).

7. b) If the ProfileMetadata contains PPR(s) and the LPAd does not already have the Rules Authorisation Table, then the LPAd SHALL request the Rules Authorisation Table from the eUICC by calling the "ES10b.GetRAT" function.
7. c) If the ProfileMetadata contains PPR1 and the LPAd does not already have the list of installed Profiles, then the LPAd SHALL request the information from the eUICC by calling the "ES10b.GetProfilesInfo" function. If the ProfileMetadata contains PPR1 and an Operational Profile is installed, the LPAd SHALL perform the Sub-procedure "Profile Download and installation - Download rejection" hereunder with reason code 'PPR not allowed'.
8. If the ProfileMetadata contains PPR(s), the LPAd SHALL check if the PPR(s) is/are allowed based on the Rules Authorisation Table defined in section 2.9.2.3. If one or more PPR(s) are not allowed, the LPAd SHALL continue the Sub-procedure "Profile Download and installation – Download rejection" hereunder with reason code ‘PPR not allowed’. If any PPR is subject to additional End User consent according to the RAT, LPAd SHALL ask for the End User consent by showing relevant information concerning the PPR(s). This information SHALL include the consequences of the Profile Policy Rule to the End User. This message SHALL be formulated in a descriptive and non-discriminatory manner (e.g. for "Non-Delete" Profile Policy Rule: "The profile that you are about to install can be deleted only under the terms you have agreed with your service provider. Approve installation YES/NO?").

   If the End User does not agree to the Profile Policy Rules, the LPAd SHALL continue the Sub-procedure "Profile Download and installation – Download rejection" hereunder with reason code 'End User rejection'.

   Authenticated Confirmation SHALL be enforced except in the case of initial Device setup.

   The request for this End User consent for the installation of Profile Policy Rules and Profile download hereunder MAY be combined into a single prompt therefore requiring a single confirmation by the End User. Whether combined or separated, these can be performed either at this step or after the BPP has been downloaded by the LPAd, since the same Profile Metadata will also be available then.
   If the Confirmation Code Required Flag is set in either the Activation Code Token or in the smdpSigned2, then the LPAd SHALL prompt the End User to enter the Confirmation Code which was provided by the Operator. When prompting, the LPAd MAY also display the ProfileName or any relevant information contained in the Profile Metadata to help the End User identify the Profile to be downloaded during this RSP session. If the Confirmation Code is not required, the LPAd MAY ask for the user confirmation (e.g. simple 'Yes' or 'No' or 'Not Now') by showing the ProfileName or any relevant information contained in the Profile Metadata to the End User either at this step or after the BPP has been downloaded by the LPAd, since the same Profile Metadata will also be available then.

   The case where the End User does not agree to the download of the Profile (e.g. by selecting 'No' or 'Not Now'), is described in the Sub-procedure "Profile Download and Installation – Download Rejection" hereunder.

If the End User does not respond to the LPAd prompt within an implementation-dependent timeout interval, the LPAd SHALL cancel the Profile download by performing the sub-procedure "Profile Download and Installation – Download Rejection" hereunder with the reason 'Timeout'.

If required, the LPAd SHALL calculate the hash of the Confirmation Code as follows:

Hashed Confirmation Code = SHA256 (SHA256 (Confirmation Code) | TransactionID), where '|' means concatenation of data.

If Profile download has not been rejected in the steps above, the procedure SHALL continue with the Sub-procedure "Profile Download and installation – Download confirmation"

3.1.3.1 Sub-procedure Profile Download and Installation – Download Rejection

This procedure can occur due to an End User rejection or timeout of a Profile download at the following steps of the protocol:

- after the response to "ES9+.Authenticate Client" (section 3.1.3), and
- after the response to "ES9+.GetBoundProfilePackage" (section 3.1.3.2).

The LPAd MAY provide additional places where the End User would be offered such possibility.

This procedure can also occur if the LPAd detects a mismatch in some of the Profile Metadata returned by "ES9+.Authenticate Client" and "ES9+.GetBoundProfilePackage".

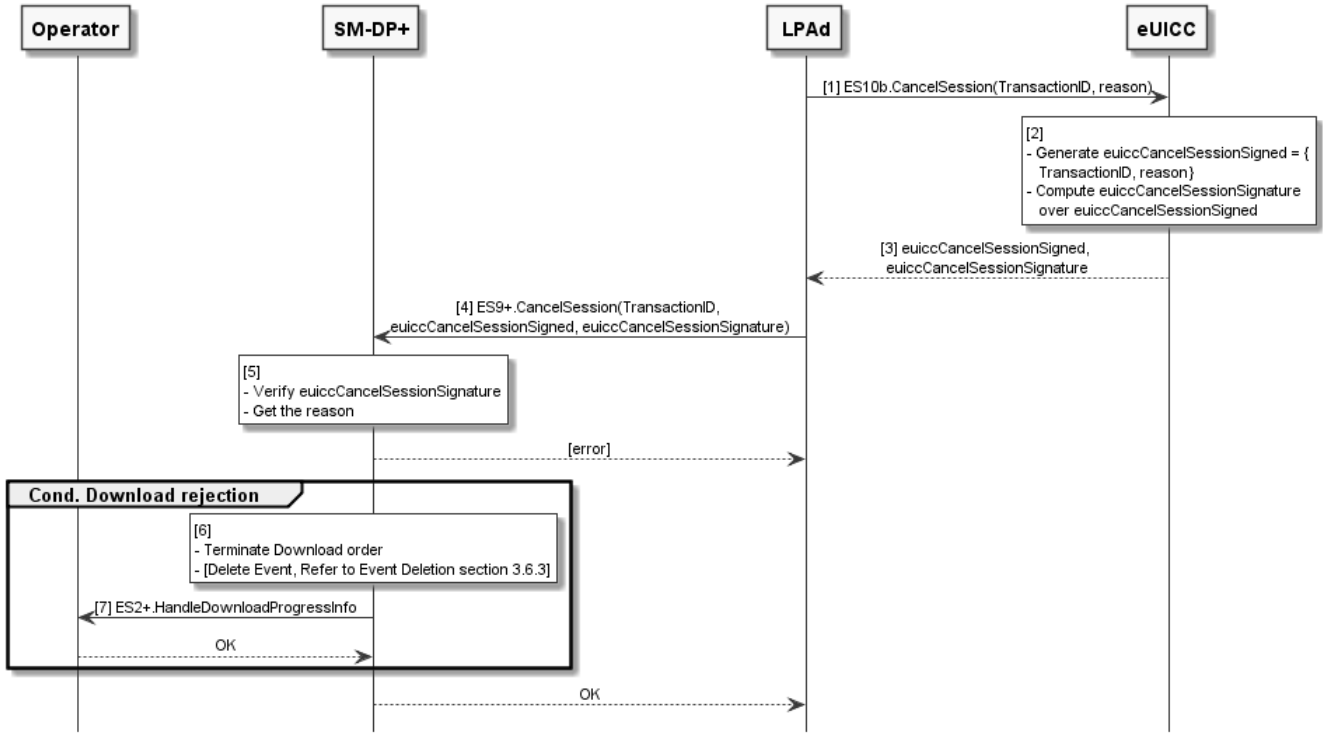


Figure 12: Sub-procedure Profile Download and Installation - Download Rejection

Start Conditions:

The End User has not agreed to the download of the Profile (e.g. by selecting 'No'), the PPR(s) in the ProfileMetadata is/are not allowed according to the Rules Authorisation Table, PPR1 is present in the ProfileMetadata and an Operational Profile is already installed on the eUICC, the ProfileMetadata in the response to "ES9+.Authenticate Client" does not match the ProfileMetadata in the Bound Profile Package, or the LPAd has encountered an error while installing the Bound Profile Package.

Procedure:

1. The LPAd SHALL call the "ES10b.CancelSession" function with input data including the TransactionID and the reason 'End User rejection', 'End User postponed' or 'Timeout', ‘PPR not allowed’, 'Metadata mismatch', 'Load BPP execution error' or 'undefinedReason'.
2. On reception of this function call, the eUICC SHALL:

   - Generate the euiccCancelSessionSigned data object containing the TransactionID and the reason provided by the LPAd.
   - Compute the euiccCancelSessionSignature over euiccCancelSessionSigned using the SK.EUICC.ECDSA corresponding to the euiccCiPKIdToBeUsed as received during the Common mutual authentication procedure.

3. The eUICC SHALL return the euiccCancelSessionSigned and euiccCancelSessionSignature.
4. The LPAd SHALL call the "ES9+.CancelSession" function with input data including the TransactionID, the euiccCancelSessionSigned and the euiccCancelSessionSignature.
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. The LPAd MAY retry this step with a different TransactionID if the provided value was incorrect.
   - 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 be stopped by the LPAd.
   - Verify that the received smdpOid corresponds to the SM-DP+ (i.e. is the same value as the one contained in the CERT.DPauth.ECDSA used during the Common Mutual Authentication Procedure). If the value doesn't match, the SM-DP+ SHALL return a function execution status 'Failed' with relevant status code and the procedure SHALL be stopped by the LPAd.

   If the reason contained in euiccCancelSessionSigned indicates 'End user postponed' or 'Timeout', the SM-DP+ SHALL simply return a function execution status 'Executed-Success' and keep the corresponding Profile download order in the 'Released' state available for a further retry, and the procedure SHALL be stopped.
   If the reason contained in euiccCancelSessionSigned indicates any other condition the SM-DP+ SHALL perform the following steps.

6. The SM-DP+ SHALL set the Profile associated with the on-going RSP session in 'Error' state (section 3.1.6); and if this procedure is executed in the context of option (b), the SM-DP+ SHALL execute the SM-DS event deletion procedure described in section 3.6.3.
7. The SM-DP+ SHALL call the "ES2+.HandleDownloadProgressInfo" function with the relevant notificationPointId set and an operation status indicating 'Failed' with status code value depending on the given cancel reason. The cancel session reason code mapping to status code is given in section 5.3.5.

   The SM-DP+ SHALL return a function execution status 'Executed-Success' and the procedure SHALL be stopped.

3.1.3.2 Sub-procedure Profile Download and Installation – Download Confirmation

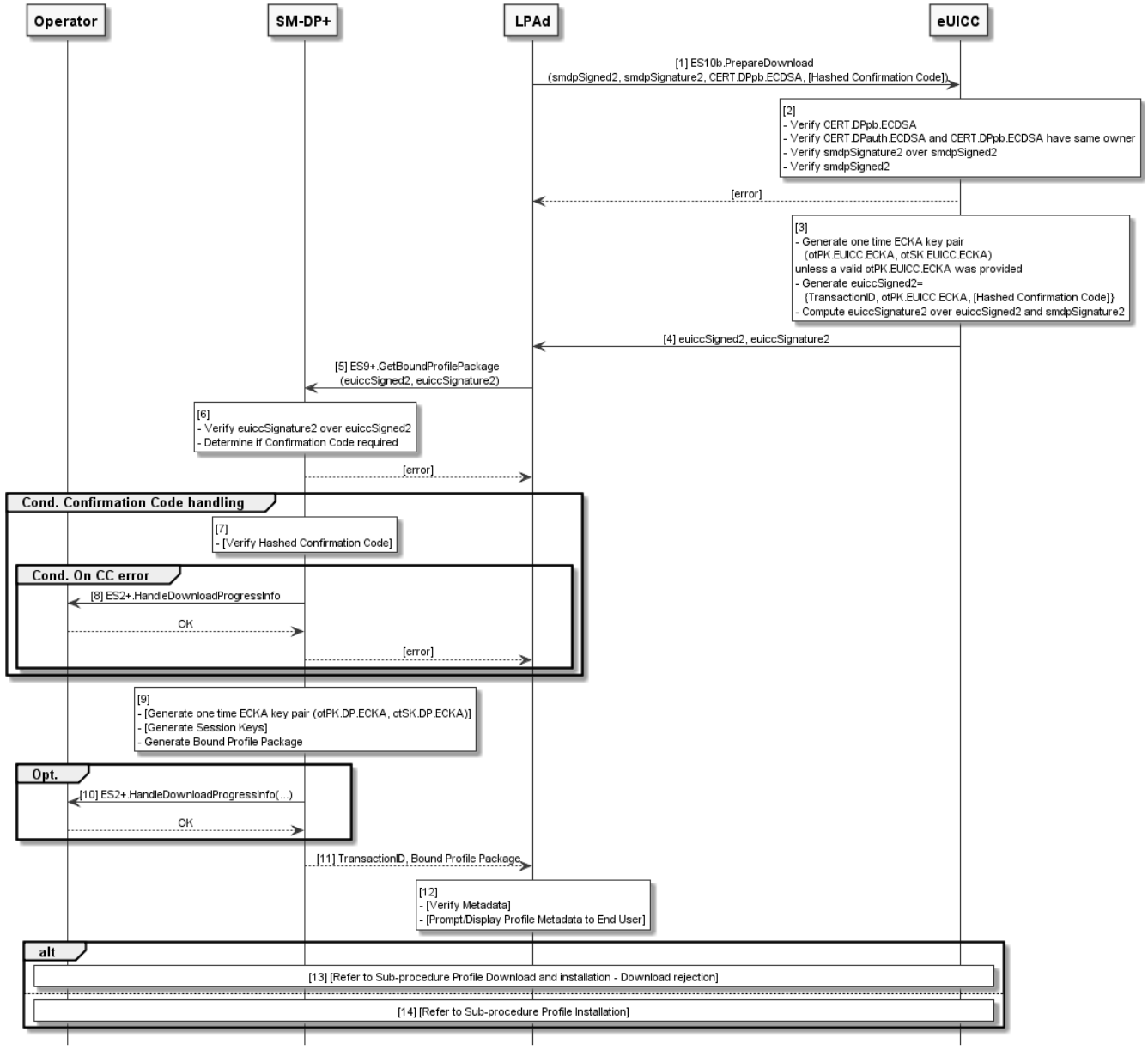


Figure 13: Sub-procedure Profile Download and Installation – Download Confirmation

Start Conditions:

The End User has agreed to the download of the Profile (e.g. by selecting 'Yes').

Procedure:

1. The LPAd SHALL call the "ES10b.PrepareDownload" function with input data including the smdpSigned2, smdpSignature2, CERT.DPpb.ECDSA and optionally the Hashed Confirmation Code calculated during the last step of the profile download and installation procedure (section 3.1.3).
2. On reception of the "ES10b.PrepareDownload" function, the eUICC SHALL:

   - Verify that CERT.DPpb.ECDSA is valid according to section 4.5.2.2.
   - Verify that CERT.DPauth.ECDSA and CERT.DPpb.ECDSA belong to the same entity (i.e. same OID in, same subjectAltName).
   - Verify smdpSignature2 performed over smdpSigned2 and euiccSignature1 using the PK.DPpb.ECDSA contained in CERT.DPpb.ECDSA.
   - Verify that the TransactionID contained in smdpSigned2 matches the TransactionID of the on-going RSP session.
   - Verify that the Hashed Confirmation Code is provided by the LPAd if the Confirmation Code Required Flag is set in smdpSigned2.

   If any of the verifications fail, the eUICC SHALL return a relevant error status and the procedure SHALL be stopped.

3. Otherwise the eUICC SHALL:

   - If bppEuiccOtpk is provided in smdpSigned2 and it corresponds to a stored one-time key pair (otPK.EUICC.ECKA, otSK.EUICC.ECKA) for this SM-DP+, the eUICC SHALL use this key pair for the RSP session. Otherwise it SHALL generate a new one-time key pair (otPK.EUICC.ECKA, otSK.EUICC.ECKA) using the curve indicated by the Key Parameter Reference Value of CERT.DPpb.ECDSA.
   - Generate the euiccSigned2 data structure containing the TransactionID, otPK.EUICC.ECKA and optionally the Hashed Confirmation Code.
   - Compute the euiccSignature2 over euiccSigned2 and smdpSignature2 using SK.EUICC.ECDSA. When generating the euiccSignature2, the eUICC SHALL use the same private key as in the AuthenticateServer response.

4. The eUICC SHALL return the euiccSigned2 and euiccSignature2.
5. The LPAd calls the "ES9+.GetBoundProfilePackage" function with input data including the euiccSigned2, euiccSignature2.
6. On reception of the "ES9+.GetBoundProfilePackage" function, the SM-DP+ SHALL verify the euiccSignature2 performed over euiccSigned2 and smdpsignature2 using the PK.EUICC.ECDSA associated with the RSP session identified by TransactionID.
   If a Confirmation Code is required the SM-DP+ SHALL continue with step 7. Otherwise, the SM-DP+ SHALL continue at step 9.
7. If a Confirmation Code verification is required, the SM-DP+ SHALL:

   - Retrieve the hashed Confirmation Code stored for this order by "ES2.ConfirmOrder" and calculate the expected hash value as

expected hash value =
SHA256(stored hashed Confirmation Code | TransactionID)

- Verify that the received Hashed Confirmation Code matches the expected hash value.
- In case the Confirmation Code verification has failed, the SM-DP+ SHALL increment the count of Confirmation Code attempts for the Profile. If the maximum number of retries has been exceeded, the SM-DP+ SHALL set the Profile corresponding to the Profile download order in 'Error' state (section 3.1.6) and execute step 8.

8. (Conditional) The SM-DP+ SHALL notify the Operator by calling "ES2+.HandleDownloadProgressInfo" with the corresponding identification point reached (in that case it SHALL be 'Confirmation Code check') and the operation status (Execution status = 'Failed' with the relevant status code value).

   If any verification in step 6 or 7 failed, the SM-DP+ SHALL return an error status to the LPAd and the procedure SHALL be stopped. Unless the maximum number of retries for Confirmation Code entries has been exceeded, the LPAd MAY retry by restarting the Profile download and installation procedure.

9. Otherwise, the SM-DP+ SHALL perform the following:

   - If a re-usable BPP exists, only the signature for InitialiseSecureChannel needs to be recalculated and the next steps can be skipped.
   - Generate a one-time ECKA key pair (otPK.DP.ECKA, otSK.DP.ECKA) using the curve indicated by the Key Parameter Reference Value of CERT.DPpb.ECDSA.
   - Generate Session Keys using the CRT, otPK.eUCC.ECKA, and otSK.DP.ECKA according to Annex G.
   - Prepare the Bound Profile Package according to the description given in section 2.5.4, including optionally the Profile Protection Keys (PPK).

10. (Optional step) Depending on the agreed behaviour with the Operator (out of scope of this specification), the SM-DP+ SHALL notify the Operator that the Profile is downloaded 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 'BPP download'), the timestamp when this point was reached, and the execution result of this step.

    NOTE: This notification step MAY be done asynchronously.

11. The SM-DP+ responds back to the LPAd with the TransactionID and the Bound Profile Package and set the Profile corresponding to the Profile download order in 'Downloaded' state (section 3.1.6).
12. On reception of the SM-DP+ response, the LPAd MAY perform additional processing using the Profile Metadata contained within the Bound Profile Package:

    - If the LPAd previously used the Profile Metadata returned by "ES9+.AuthenticateClient" (i.e., in step (7 and 8) of the procedure described in section 3.1.3), then

- The LPAd SHALL verify that the Profile Policy Rules have not changed. If this verification fails, the LPAd SHALL execute the Sub-procedure "Profile Download and installation – Download Rejection" (section 3.1.3.1) with reason code 'PPR not allowed'.
- The LPAd SHOULD verify that all other Profile Metadata elements it used in that earlier step (such as the Profile Name, Icon, etc.) have not changed. The LPAd MAY inform the End User and offer the End User to postpone or reject the Profile installation. Alternatively, the LPAd MAY stop the download by executing the Sub-procedure "Profile Download and installation – Download Rejection" (section 3.1.3.1) with reason code 'Metadata mismatch'.

- If the LPA has not previously captured the End User consent related to Profile Metadata and any Profile Policy Rules, as defined in section 3.1.3 step (8), it SHALL do so at this point.
- The LPAd MAY display any relevant part of the Profile Metadata to help the End User identify the Profile to be installed during this transaction. Based on this information, the LPAd MAY offer the End User to postpone or reject the Profile installation. If the End User does not respond to the LPAd prompt within an implementation-dependent timeout interval, the LPAd SHALL cancel the Profile download by performing the sub-procedure "Profile Download and Installation – Download Rejection" described in section 3.1.3.1 with the reason 'Timeout'.

13. If End User has postponed or rejected the Profile installation, the Sub-procedure "Profile Download and Installation – Download Rejection" described in section 3.1.3.1 SHALL be executed.
14. Otherwise sub-procedure Profile installation described in section hereafter SHALL be executed.

3.1.3.3 Sub-procedure Profile Installation

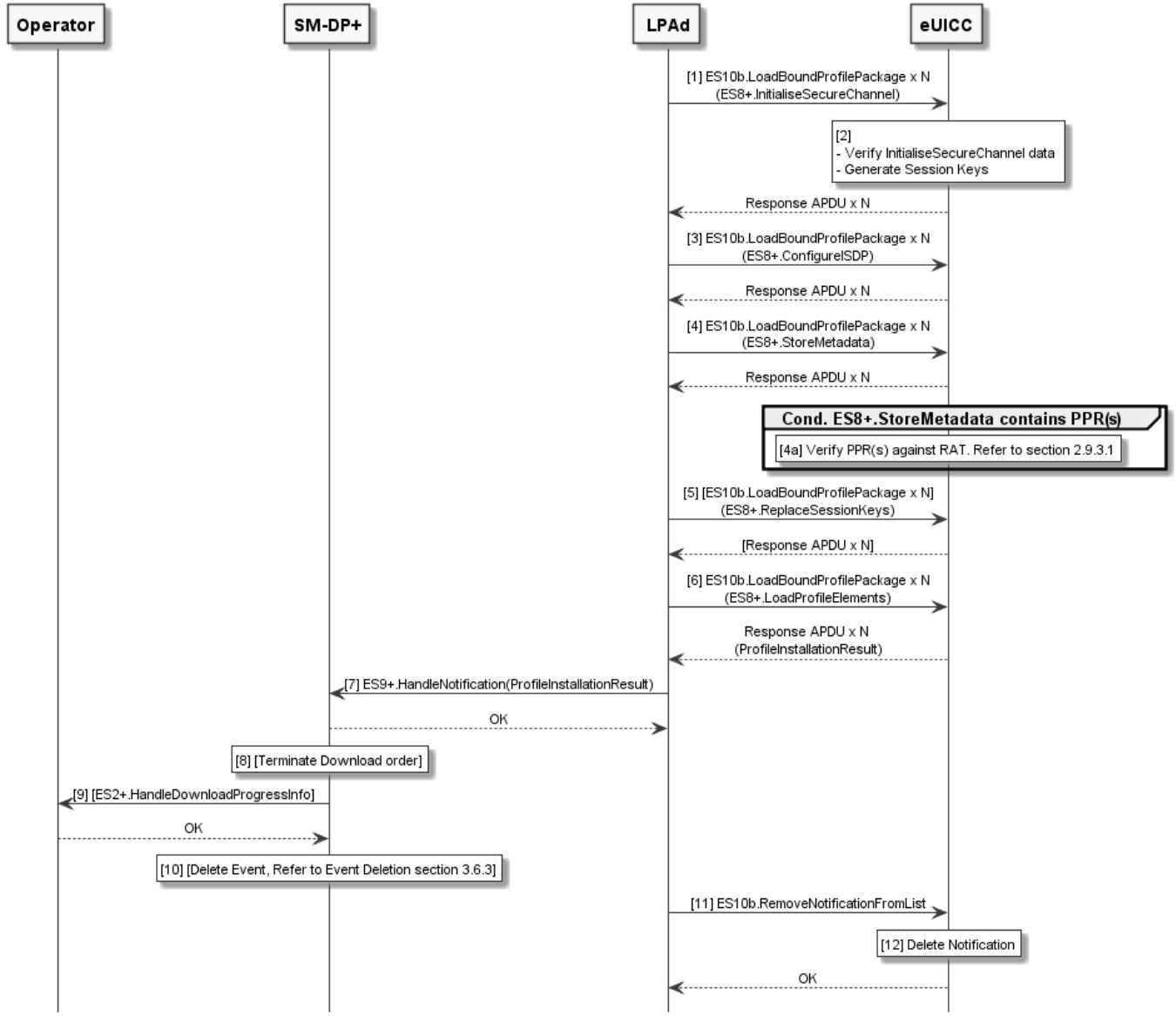


Figure 14: Sub-procedure Profile Installation

In this sub-procedure the LPAd generates the Segmented Bound Profile Package according to the description in section 2.5.5 and transfers it to the eUICC using a sequence of "ES10b.LoadBoundProfilePackage" commands. If the LPAd is unable to perform the segmentation (e.g., because of an error in the BPP structure), or if any call of "ES10b.LoadBoundProfilePackage" returns status words other than '90 00' or '91 XX', the LPAd SHALL perform the Sub-procedure "Profile Download and installation – Download rejection" with reason code 'Load BPP execution error'.

1. The LPAd SHALL transfer the "ES8+.InitialiseSecureChannel" function call included in the Bound Profile Package to the eUICC by repeatedly calling the "ES10b.LoadBoundProfilePackage" function. The input data of the "ES8+.InitialiseSecureChannel" function includes the CRT, otPK.DP.ECKA and smdpSign2.

2. The eUICC SHALL verify the received "ES8+.InitialiseSecureChannel" according to the description in section 5.5.1. If the verification succeeds, the eUICC SHALL generate Session Keys using the input data received in previous step according to Annex G.
3. The LPAd SHALL transfer the "ES8+.ConfigureISDP" function call included in the Bound Profile Package to the eUICC by repeatedly calling the "ES10b.LoadBoundProfilePackage" function.
4. The LPAd SHALL transfer the "ES8+.StoreMetadata" function call included in the Bound Profile Package to the eUICC by repeatedly calling the "ES10b.LoadBoundProfilePackage" function.

4. a) If "ES8+.StoreMetadata" contains PPR(s), the eUICC SHALL verify each PPR according to PPR verification section 2.9.3.1.

5. If the Profile Protection Keys (PPK) were included in the Bound Profile Package, the LPAd SHALL transfer the "ES8+.ReplaceSessionKeys" function call to the eUICC by repeatedly calling the "ES10b.LoadBoundProfilePackage" function. The input data of the "ES8+.ReplaceSessionKeys" function includes the PPK. Once the eUICC receives "ES8+.ReplaceSessionKeys" function call, it SHALL decrypt the Profile Protection Keys and replace the current SCP03t Session Keys with the decrypted Profile Protection Keys.
6. The LPAd SHALL transfer the Profile Elements included in the "ES8+.LoadProfileElements" functions by repeatedly calling the "ES10b.LoadBoundProfilePackage" function.

   If all the Profile Elements are successfully processed and installed, with or without any warning, the last response of the "ES10b.LoadBoundProfilePackage" function SHALL deliver the Profile Installation Result, comprising the `ProfileInstallationResultData` as defined in section 2.5.6 and the `EuiccSignPIR` which is the signature generated across the `ProfileInstallationResultData` data object using SK.EUICC.ECDSA).
   Installation notifications as configured in StoreMetadata, if any, SHALL be generated.

   Otherwise, if an error occurs during the transfer and processing of the Profile Elements, or any previous ES8+ function, which does not result in error status words, the eUICC SHALL stop the procedure and SHALL report to the LPAd with a response of the "ES10b.LoadBoundProfilePackage" function including the Profile Installation Result.

   The eUICC SHALL store the Profile Installation Result in its non-volatile memory before delivering it to the LPAd.
   The eUICC SHALL erase the otSK.EUICC.ECKA attached to this RSP session no later than the successful completion of the BPP installation.

7. The LPAd calls the "ES9+.HandleNotification" function with input data including the Profile Installation Result which was given by the eUICC in the step 6.
8. On reception of the "ES9+.HandleNotification" function, the SM-DP+ SHALL:

   - Acknowledge the reception of the notification to the LPAd.
   - Retrieve the pending download order identified by the TransactionID. If TransactionID is unknown, the SM-DP+ SHALL terminate its processing.

- (Conditional) Terminate the pending download order and set the corresponding Profile in state 'Installed' or 'Error' (section 3.1.6) as indicated by the Profile Installation Result.

9. (Conditional) The SM-DP+ SHALL call the "ES2+.HandleDownloadProgressInfo" with input data including eid, iccid, ProfileType, completionTimestamp, resultData, identification of the point reached (in that case it SHALL be 'BPP installation') and operationStatus set accordingly to Profile Installation Result.
10. (Conditional) If this procedure is executed in the context of option (b), the SM-DP+ SHALL execute the SM-DS event deletion procedure (section 3.6.3).
11. On reception of the acknowledgement message from the SM-DP+ the LPAd SHALL call "ES10b.RemoveNotificationFromList" with corresponding seqNumber as input parameter.
12. The eUICC SHALL delete the Profile Installation Result from its non-volatile memory.

3.1.4 Limitation for Profile Installation

Several profiles MAY be installed on the eUICC, subject to non-volatile memory limitations.

3.1.5 Error Handling Within the Profile Download Procedure

The Profile download and installation procedure comprises a sequence of operations between the SM-DP+, the LPA, and the eUICC over a period of time. In addition to errors reported by ES9+ and ES10b functions, other conditions MAY impact the successful execution of this procedure. The LPA SHOULD indicate such failures to the user; however, the specific presentation of these errors is out of the scope of this document.

The LPA SHOULD NOT initiate a new Profile download and installation procedure while there is an active download RSP session. However, in the event that this does occur, the eUICC SHALL discard its session state (including generated eUICC challenge, any downloaded Profile Metadata, Profile contents, and Profile Installation Result) with the possible exception that an unused otPK/otSK.EUICC.ECKA MAY be stored for future retry, when a new RSP session is started with "ES10b.GetEUICCChallenge".

The eUICC MAY discard its session state if a Profile switch occurs during a Profile download and installation procedure.

If an eUICC Memory Reset or eUICC Test Memory Reset is successfully processed during a Profile download and installation procedure, the eUICC SHALL discard its session state.

If the eUICC receives a BPP segment with an unrecognized leading tag (see section 2.5.5) during Profile download it SHALL return status words of '6A 88' (Reference data not found) and SHALL not discard the download session state.

The Profile download and installation procedure MAY fail because of a communications failure between the LPA and the SM-DP+. The LPA MAY retry for a period of time. The LPA SHALL reset its own Profile download session state when all retry attempts have failed.

While the SBPP is sent to the eUICC using "ES10b.LoadBoundProfilePackage", the eUICC MAY reject any other ES10 command with status words '69 85' (Conditions of use not satisfied), except "ES10b.GetEUICCChallenge" (indicating the start of a new download

session) or "ES10b.CancelSession" (indicating the termination of the current download session).

The Profile download and installation procedure could fail while the LPA is sending SBPP TLVs to the eUICC using "ES10b.LoadBoundProfilePackage" for reasons other than an error status reported by the eUICC. Examples of such failures during the download process include:

- In the case of a removable eUICC card, the End User could remove the card.
- The End User could switch off the power or remove the battery.
- A software fault could cause a crash of the LPA, host Device, and/or baseband processor.

The LPA SHOULD provide an appropriate error indication to the End User when possible (e.g., when power is restored). The specific presentation of such an error notification is out of scope of this document.

3.1.6 Profile Lifecycle at SM-DP+

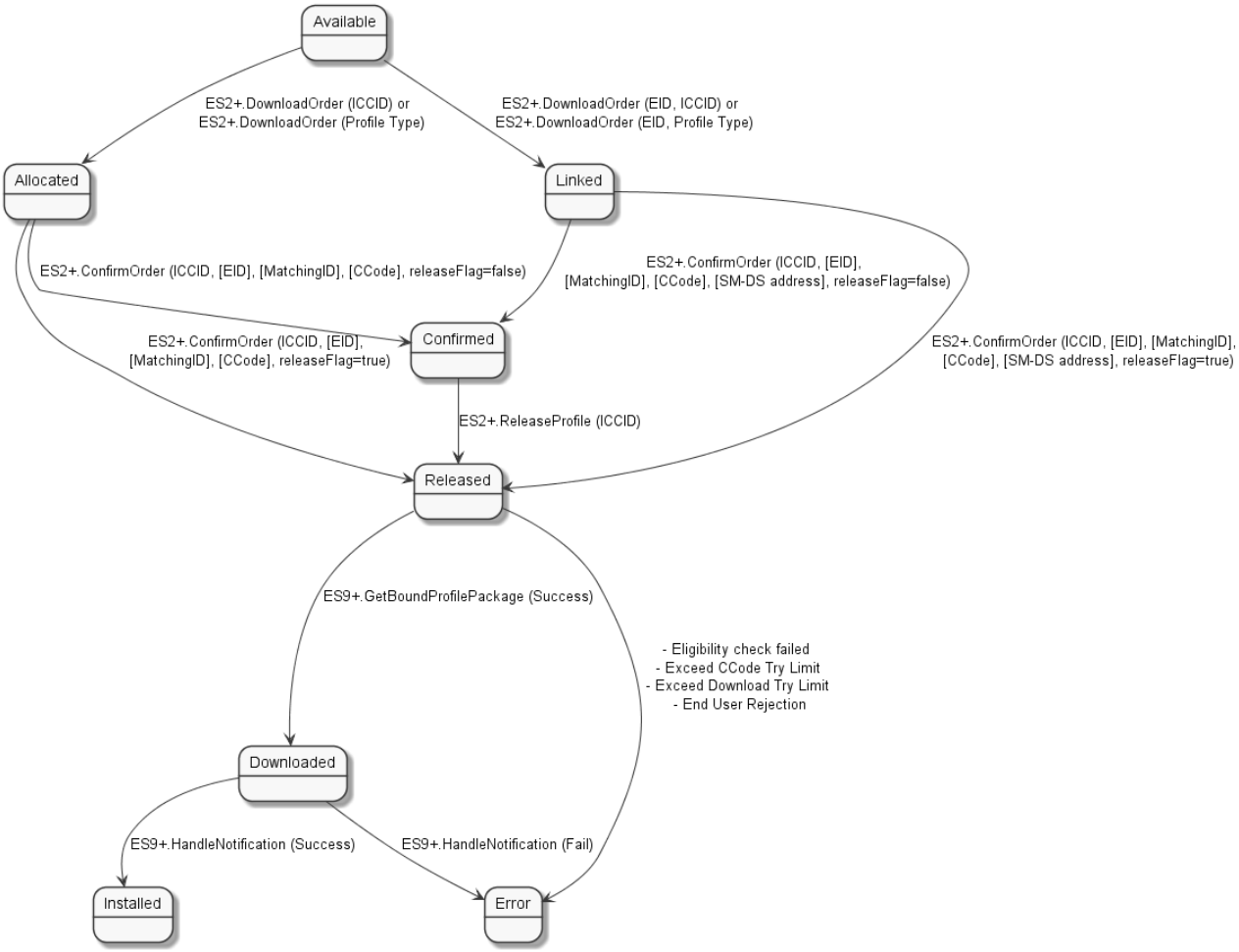
The previous sections provide detailed procedures associated with Remote Provisioning. Each Profile has state information on the SM-DP+ associated with it during the provisioning into an eUICC. The Profile lifecycle state can be one of the states listed in the following table.

Additional states and additional or customised ES2+ functions MAY be agreed between the Operator and the SM-DP+.

State Name	Description
Available	The Profile is available in the inventory of the SM-DP+.
Allocated	The Profile is reserved for downloading without being linked to an EID.
Linked	The Profile is reserved for downloading and is linked to an EID.
Confirmed	The Profile is reserved for downloading (linked or not linked to an EID) with Matching ID and Confirmation Code if required.
Released	The Profile is ready for download and installation after Network Configuration by the Operator (e.g.: HLR Registration).
Downloaded	The Bound Profile was delivered to the LPA.
Installed	The Profile was successfully installed on the eUICC.
Error	The Profile has not been installed because of one of the following error cases: - Confirmation Code Retry Limit exceeded - Download Retry Limit exceeded - End User Rejection - Error during download and installation
Unavailable	The Profile cannot be reused anymore by the SM-DP+.

Table 6b: Profile State in the SM-DP+

The following two state transition diagrams show the Profile lifecycle state on the SM-DP+ and provide the details of the actions previously performed on a Profile together with the possible next action.



NOTE: "ES2+.HandleDownloadProgressInfo" does not have any impact on the Profile state.

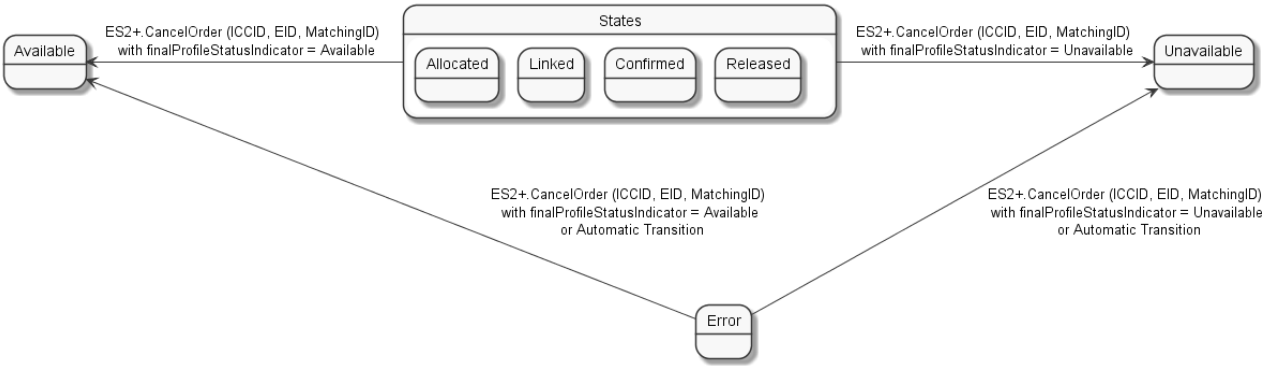


Figure 15: Profile Instance Lifecycle State Transit Diagram at SM-DP+

3.2 Local Profile Management

The End User initiates Local Profile Management procedures using the LUI. As specified in SGP.21 [4], User Intent (either Simple Confirmation or Authenticated Confirmation) is required for all procedures directed to Operational Profiles. The specific implementation of User Intent verification by the LPA is out of scope of this specification.

The LPA implementation SHALL be capable of enforcing User Intent at two different levels: for each Profile Management Operation or upon entry to the LUI.

- User Intent MAY be enforced for individual Profile Management Operations. In this case no confirmation is required to start the LUI but an Authenticated Confirmation or a Simple Confirmation is required at some point in any Local Profile Management Operation (Add Profile, Enable Profile, Disable Profile, etc.).
- User Intent MAY be enforced upon entry to the LUI. In this case, an Authenticated Confirmation SHALL be obtained when the End User starts the LUI and when the End User has been inactive with the LUI for an implementation-specific period of time. Only a Simple Confirmation is required for any subsequent operation.

In all cases, if the End User refuses a request for Authenticated Confirmation or Simple Confirmation, then the associated operation SHALL be stopped.

User Intent verifications MAY be combined to simplify the user experience and avoid repeated input steps for the End User. For instance, when performing a Profile download with an Activation Code, the Authenticated Confirmation for download and Simple Confirmation for Enabling the Profile MAY be combined. In the case of combined verifications, it SHALL be clear to the End User what Operations will be performed, and the highest level of confirmation SHALL be obtained.

3.2.1 Enable Profile

Normal Case:

This procedure is used to enable a Profile already downloaded and installed on an eUICC.

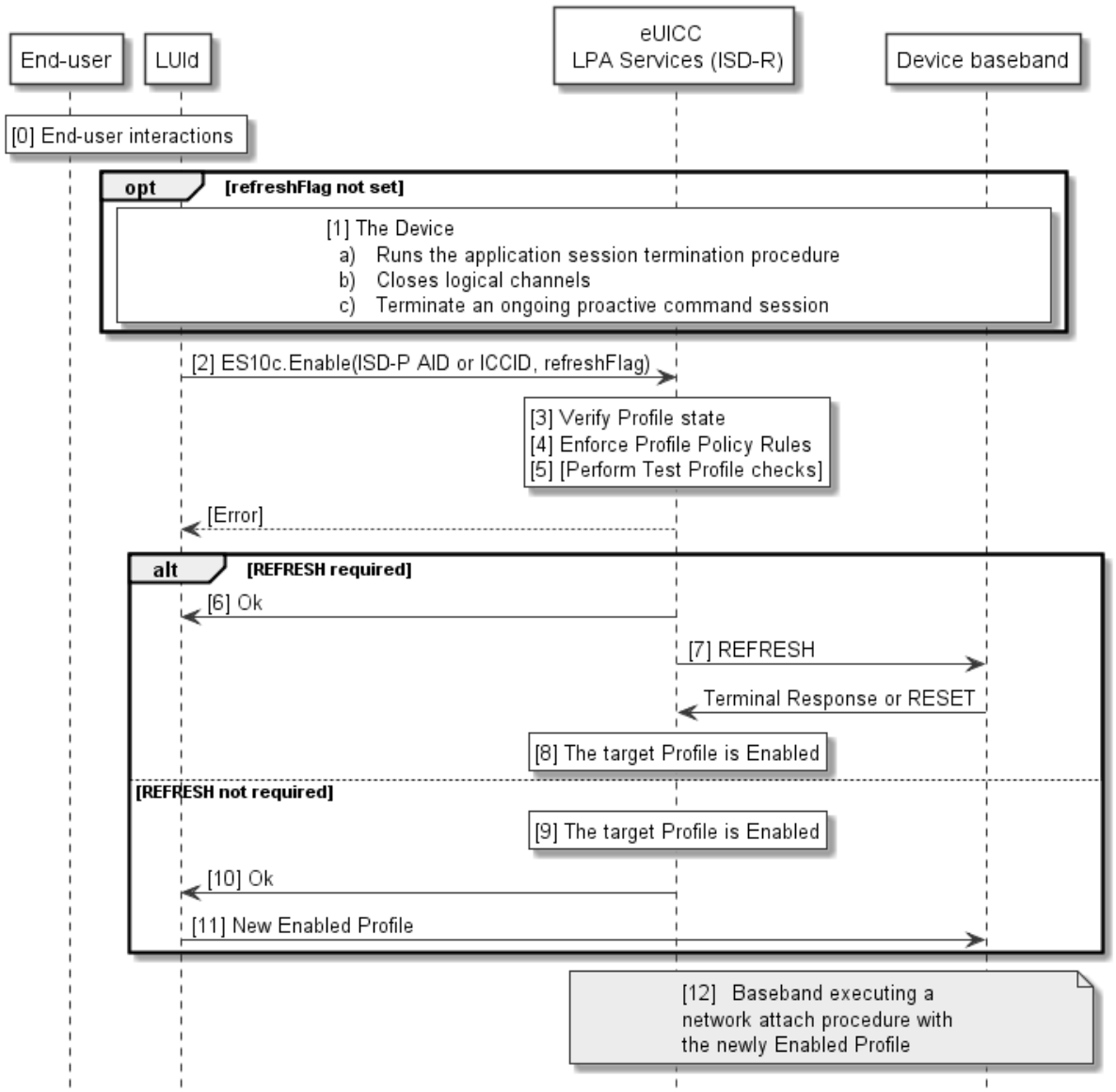


Figure 16: Enable Profile

Start Conditions:

When the Profile to be enabled is an Operational Profile:

- End User Intent is verified as defined in SGP.21 [4] for Simple Confirmation.

When the Profile to be enabled is a Test Profile:

- The Device is in Device Test Mode.

When the Profile to be enabled is a Provisioning Profile:

- The currently-enabled Operational Profile, if any, is unsuitable to provide the connectivity required for an operation such as Add Profile.

Procedure:

0. The End User is presented a user interface that displays the list of installed Profiles within the eUICC, with their current states (Enabled or Disabled), as described in "List Profiles" procedure (section 3.2.4). The End User selects the Profile to be enabled. The LPA MAY check the Profile Policy Rules of the Profiles and give appropriate warnings to the End User (e.g. that due to Profile Policy Rules the Profile cannot be enabled). The enabling of a Provisioning Profile can be initiated by the LPAd itself without any End User interaction.
1. Before the LPAd calls the EnableProfile function with the refreshFlag not set, the Device has the responsibility to ensure that the relevant conditions for use are met. i.e. the Device:

    a) SHALL run the application session termination procedure in accordance with ETSI TS 102 221 [1] for every active application of the currently enabled Profile.
    b) SHALL close all logical channels that were used to select these applications.
    c) SHALL terminate an ongoing proactive command session.

2. The LPAd SHALL call the "ES10c.EnableProfile" (section 5.7.16) function of the ISD-R with its relevant input data, which includes the indication if a REFRESH proactive command is needed.
3. The ISD-R SHALL verify the state of the target Profile. If the target Profile is not in Disabled state, the ISD-R SHALL return a response indicating a failure, and the procedure SHALL be stopped.
4. If the target Profile is not a Test Profile, the ISD-R SHALL check the Profile Policy Rules of the currently Enabled Profile (if any). If it has a Profile Policy Rule "Disabling not allowed", the ISD-R SHALL return a response indicating a failure, and the procedure SHALL end.
5. If the currently Enabled Profile is a Test Profile, the ISD-R SHALL check if the target Profile is either another Test Profile or the Operational profile that was previously in Enable state. If this is not the case, the ISD-R SHALL return a response indicating a failure, and the procedure SHALL end.

If the refreshFlag is set, steps 6 to 8 SHALL be executed.

6. The ISD-R SHALL return a response indicating result OK back to the LUId.
7. The eUICC SHALL send a REFRESH proactive command.
8. Upon reception of the Terminal Response or after the RESET, the ISD-R SHALL disable the currently Enabled Profile (if any) and then enable the targeted Profile.

If the refreshFlag is not set, steps 9 to 11 SHALL be executed.

9. The ISD-R SHALL disable the currently Enabled Profile (if any) and then enable the target Profile.
10. The ISD-R SHALL return a response indicating result OK back to the LUId.
11. The Device SHALL discard any cached file content including EF_{ICCID} and EF_{DIR}, PIN state, and any proactive command session. The LPA signals the baseband that a new Profile was enabled. The Device SHALL proceed with the UICC activation procedure including TERMINAL PROFILE, as defined in ETSI TS 102 221 [1] clause 14.5.1.

12. The baseband triggers the execution of a network attach procedure with the newly Enabled Profile.

In case of any error after this step, indicating that the currently Enabled Profile cannot provide connectivity, there SHALL not be any fall-back to the previously Enabled Profile. Further action SHALL remain under the responsibility of the End User.

3.2.2 Disable Profile

Normal Case:

This procedure is used to disable an Enabled Profile already downloaded and installed on an eUICC.

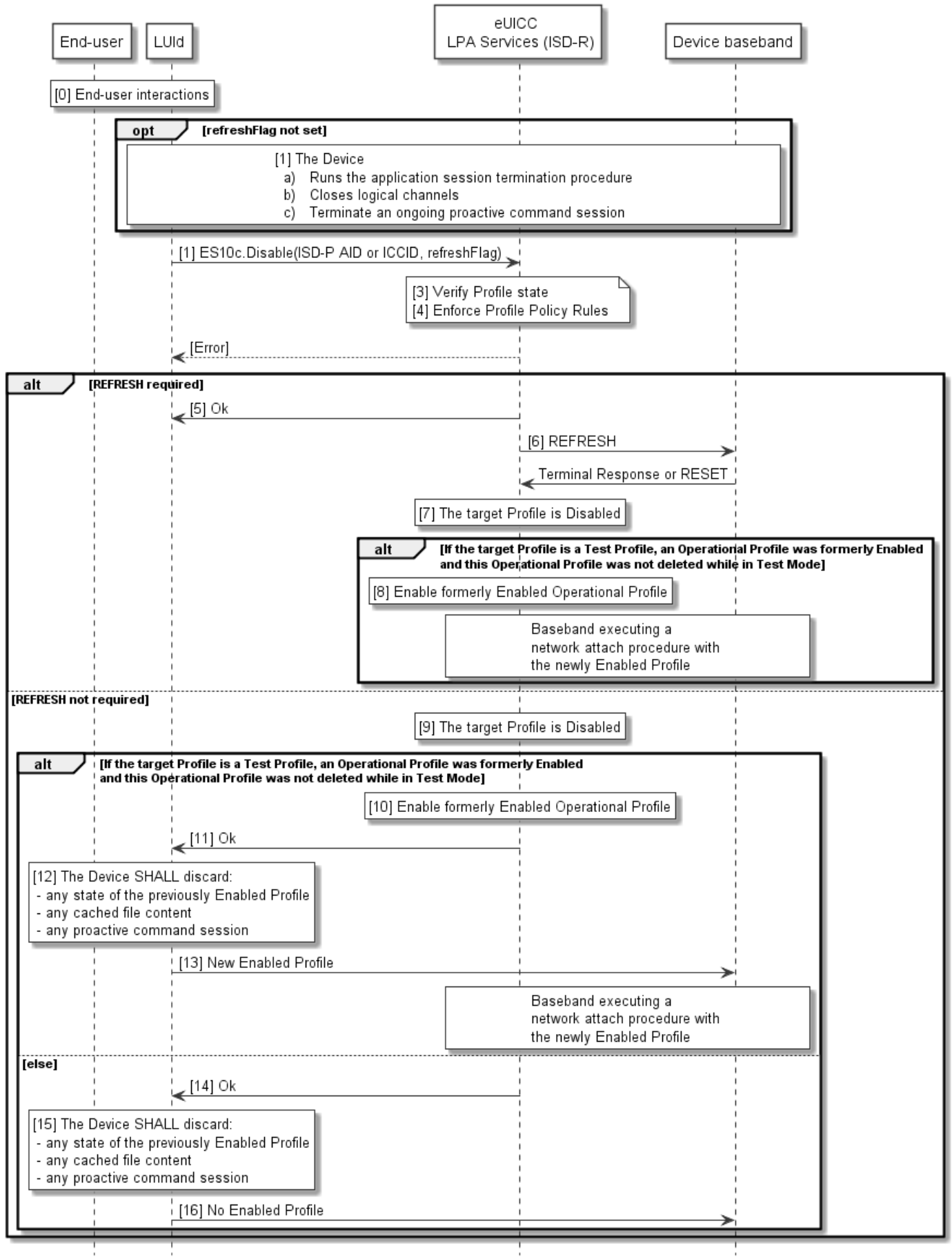

End-user
LUId
eUICC
LPA Services (ISD-R)
Device baseband
[0] End-user interactions
opt
[refreshFlag not set]
[1] The Device
a) Runs the application session termination procedure
b) Closes logical channels
c) Terminate an ongoing proactive command session
[1] ES10c.Disable(ISD-P AID or ICCID, refreshFlag)
[3] Verify Profile state
[4] Enforce Profile Policy Rules
[Error]
alt
[REFRESH required]
[5] Ok
[6] REFRESH
Terminal Response or RESET
[7] The target Profile is Disabled
alt
[If the target Profile is a Test Profile, an Operational Profile was formerly Enabled and this Operational Profile was not deleted while in Test Mode]
[8] Enable formerly Enabled Operational Profile
Baseband executing a network attach procedure with the newly Enabled Profile
[REFRESH not required]
[9] The target Profile is Disabled
alt
[If the target Profile is a Test Profile, an Operational Profile was formerly Enabled and this Operational Profile was not deleted while in Test Mode]
[10] Enable formerly Enabled Operational Profile
[11] Ok
[12] The Device SHALL discard:
- any state of the previously Enabled Profile
- any cached file content
- any proactive command session
[13] New Enabled Profile
Baseband executing a network attach procedure with the newly Enabled Profile
[else]
[14] Ok
[15] The Device SHALL discard:
- any state of the previously Enabled Profile
- any cached file content
- any proactive command session
[16] No Enabled Profile

Figure 17: Disable Profile

Start Conditions:

When the Profile to be disabled is an Operational Profile:

- End User Intent is verified as defined in SGP.21 [4] for Simple Confirmation.

When the Profile to be disabled is a Test Profile:

- The Device is in Device Test Mode.

When the Profile to be disabled is a Provisioning Profile:

- The operation requiring connectivity from the Provisioning Profile, such as a Profile download, has completed.

Procedure:

0. The End User is presented a user interface that displays the list of installed Profiles within the eUICC, with their current states (Enabled or Disabled), as described in "List Profiles" procedure (section 3.2.4) The End User selects the Profile to be Disabled. The disabling of a Provisioning Profile or a Test Profile can be initiated by the LPAd itself without any End User interaction. The LPA MAY check the Profile Policy Rules of the Profile and give appropriate warnings to the End User (e.g. that due to Profile Policy Rules the Profile will automatically be deleted after disabling).
1. Before the LPAd calls the DisableProfile function with the refreshFlag not set, the Device has the responsibility to ensure that the relevant conditions for use are met. I.e. the Device:
    a) SHALL run the application session termination procedure in accordance with ETSI TS 102 221 [1] for every active application of the currently enabled Profile.
    b) SHALL close all logical channels that were used to select these applications.
    c) SHALL terminate an ongoing proactive command session.
2. The LPAd SHALL call the "ES10c.DisableProfile" (section 5.7.17) function of the ISD-R with its relevant input data, which includes the indication if a REFRESH proactive command is needed.
3. The ISD-R SHALL verify the state of the target Profile. If the target Profile is not in the Enabled state, the ISD-R SHALL return a response indicating a failure, and the procedure SHALL be stopped.
4. The ISD-R SHALL check the Profile Policy Rules of the currently Enabled Profile. If it has a Profile Policy Rule "Disabling not allowed", the ISD-R SHALL return a response indicating a failure, and the procedure SHALL end.

If refreshFlag is set, steps 5 to 8 SHALL be executed.

5. The ISD-R SHALL return a response indicating result OK back to the LUId.
6. The ISD-R SHALL send a REFRESH proactive command.
7. Upon reception of the Terminal Response or after the RESET, the ISD-R SHALL disable the currently Enabled Profile.

8. If the target Profile is a Test Profile, an Operational Profile was in Enabled state before the Test Profile was enabled and such Operational Profile was not deleted during the time in which the Test Profile was Enabled, this previous Operational Profile SHALL be enabled again.

If refreshFlag is not set, steps 9 to 16 SHALL be executed.

9. The ISD-R SHALL disable the currently Enabled Profile.

   If the target Profile is a Test Profile, an Operational Profile was in Enabled state before the Test Profile was enabled and such Operational Profile was not deleted during the time in which the Test Profile was Enabled:

   10. This previous Operational Profile SHALL be enabled again.
   11. The ISD-R SHALL return a response indicating result OK back to the LUId.
   12. The Device SHALL discard any state of the previously Enabled Profile, any cached file content including EF_{ICCID} and EF_{DIR}, PIN state, and any proactive command session.
   13. The LPA SHALL signal the baseband that a new Profile was Enabled. The baseband triggers the execution of a network attach procedure with the newly Enabled Profile.

   If the target Profile was an Operational Profile or no installed Operational Profile was in Enabled state before the Test Profile was enabled:

   14. The ISD-R SHALL return a response indicating result OK back to the LUId.
   15. The Device SHALL discard any state of the previously Enabled Profile, any cached file content including EF_{ICCID} and EF_{DIR}, PIN state, and any proactive command session.
   16. The LPA SHALL signal the baseband that the Profile was disabled.

3.2.3 Delete Profile

This procedure is used to delete a Profile already downloaded and installed on an eUICC.

The conditions under which the LPAd MAY delete a Provisioning Profile 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.

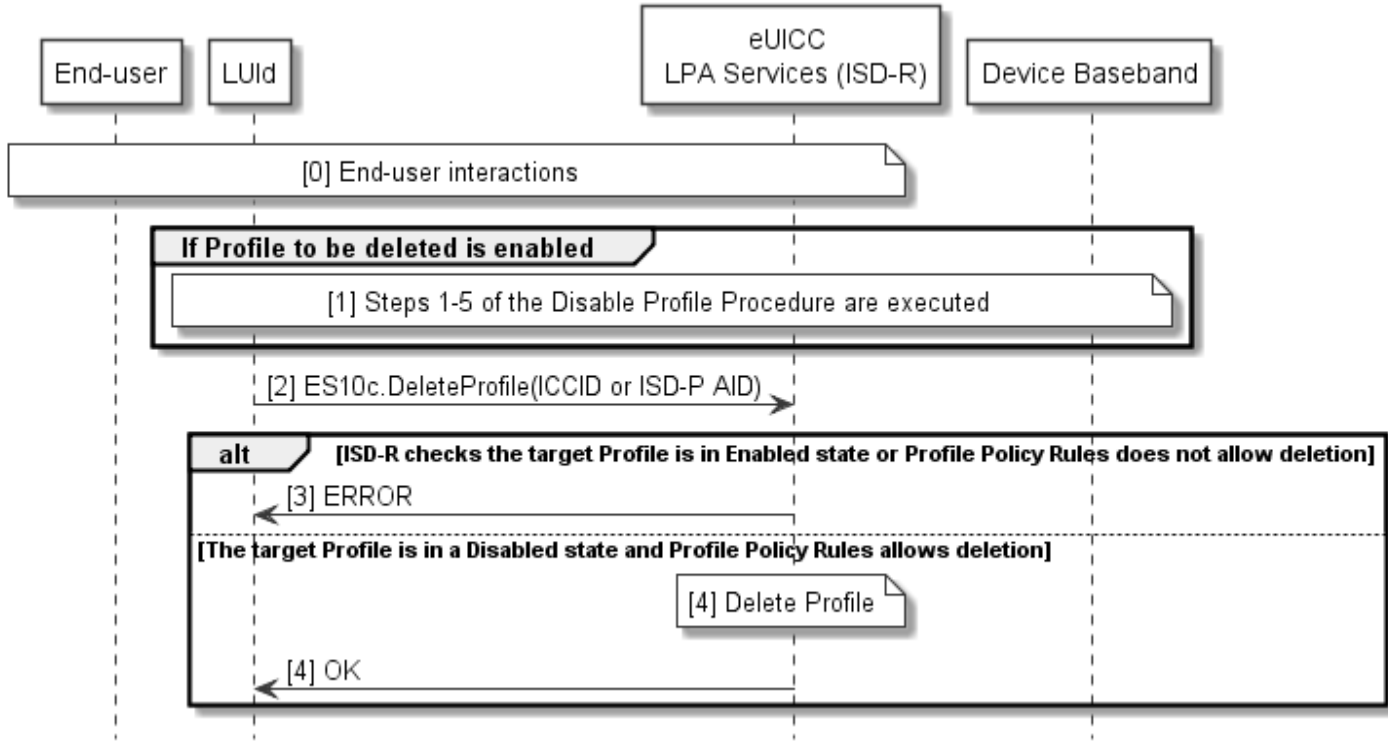


Figure 18: Delete Profile

Start Conditions:

When the Profile to be deleted is an Operational Profile:

- End User Intent is verified as defined in SGP.21 [4] for Authenticated Confirmation.

When the Profile to be deleted is a Test Profile:

- The Device is in Device Test Mode.
- The Test Profile to be deleted is not a pre-loaded Test Profile, or the eUICC implementation permits deletion of the preloaded Test Profiles.

Procedure:

0. The End User is presented a user interface that displays the list of installed Profiles within the eUICC, with their current states (Enabled or Disabled), as described in "List Profiles" procedure (section 3.2.4**Error! Reference source not found.**) The End User selects the Profile to be deleted and acknowledges the consequences. The deletion of a Provisioning Profile can be initiated by the LPAd itself without any End User interaction. The LPA MAY check the Profile Policy Rules of the Profile and give appropriate warnings to the End User (e.g. that due to Profile Policy Rules the Profile cannot be deleted).
1. If the identified Profile to be deleted is Enabled then steps 1-9 of the disable profile procedure SHALL be executed as defined in section 3.2.2.
2. The LPAd SHALL call the "ES10c.DeleteProfile" function of the ISD-R with its relevant input data.
3. The ISD-R SHALL verify the state of the target Profile and check its Profile Policy Rules. If the target Profile is in the Enabled state or the Profile Policy Rules do not

allow deletion, the ISD-R SHALL return a response indicating a failure, and the procedure SHALL be stopped.

4. The eUICC SHALL delete the Profile.
5. The ISD-R SHALL return a response indicating result OK back to the LPAd.

3.2.4 List Profiles

This procedure is used by the LPAd to list the Profiles, and their current states, pre-installed or previously downloaded and installed on an eUICC, in human readable format. The procedure is initiated by the LPAd either implicitly (e.g. at first Device boot up) or explicitly (e.g. through LUI user interface options).

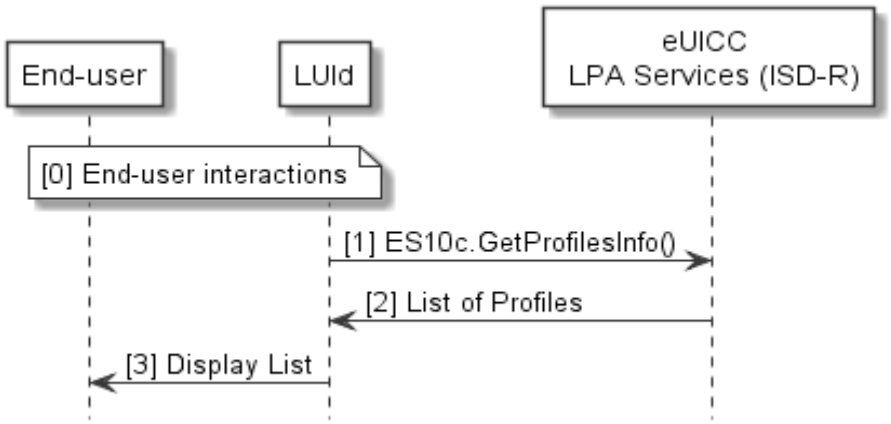


Figure 19: List Profiles

Start Conditions:

- None.

Procedure:

0. The LPAd is started on the Device. The user MAY be presented with the user interface options.
1. Either as part of the LPA launch procedure or through explicit user menu selection, the LPAd SHALL call "ES10c.GetProfilesInfo" to request the list of Profiles from the LPAd Services.
2. The eUICC SHALL return the Profile Metadata and status of the Profile(s) as defined in section 5.7.15.
3. The LUId SHALL display a subset of the set of installed Profiles along with their current states (Enabled or Disabled) to the End User in human readable format. This subset could be empty. The displayed subset SHALL include the Operational Profiles if the Device is not in Device Test Mode. It SHALL include the Test Profiles if the Device is in Device Test Mode. It SHALL not include the Provisioning Profiles.

End Conditions:

Any Profile information presented to the user SHALL always be in human readable format.

3.2.5 Add Profile

This procedure will allow the End User to add a single Profile. This procedure will not enable the downloaded Profile, nor disable an Enabled Profile. Network connectivity is assumed. The download can be initiated by the input of an Activation Code, by retrieval of a pending Profile download Event from the SM-DS, or by retrieval of a pending Profile download from the Default SM-DP+. The LPAd MAY implement a combination of these methods, as applicable, as a composite Add Profile operation.

When the End User initiates the Add Profile procedure and the Profile Metadata indicates that the Profile is not an Operational Profile, the LPAd SHOULD notify the End User and stop the procedure.

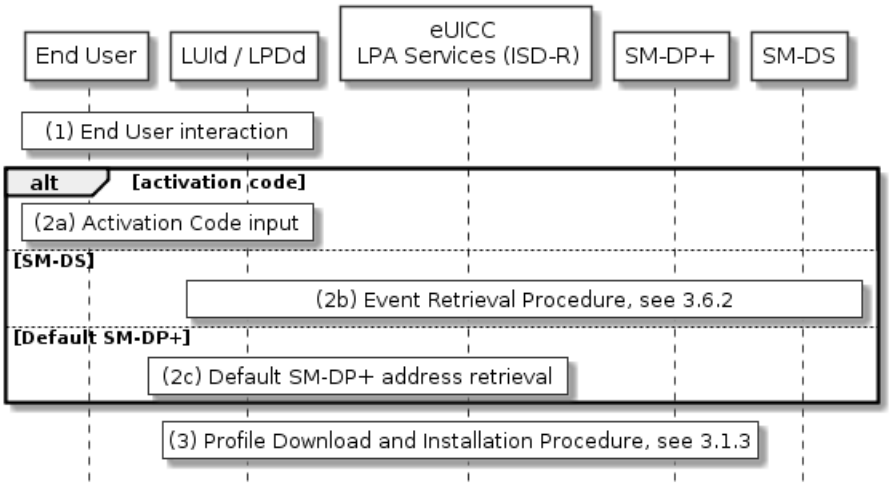


Figure 20: Add Profile

Start Conditions:

- The download of a new Profile is allowed on the eUICC.
- End User Intent is verified as defined in SGP.21 [4] for Authenticated Confirmation.

Procedure:

1. The End User initiates the Add Profile operation within the LUId.
2. The LPAd obtains the parameters for the Profile to be downloaded:
    a. If an Activation Code is used, the LUId SHALL obtain the Activation Code from the End User (e.g., by manual entry or QR code scanning).
    b. If the SM-DS is used, the LPAd SHALL retrieve the SM-DP+ address and EventID from the SM-DS using the Event Retrieval Procedure (section 3.6.2).
    c. If the Default SM-DP+ is used, the LPAd SHALL retrieve the Default SM-DP+ address from the eUICC.
3. The Profile is downloaded via the Profile download and installation procedure as defined in section 3.1.3.

End Conditions:

1. The Profile has been installed on the End User's eUICC.
2. The Profile Metadata within the eUICC is updated with the Profile Metadata from the installed Profile.

3.2.6 Set/Edit Nickname

This procedure is used to add or change the Profile Nickname associated to a Profile already downloaded and installed on an eUICC.

This procedure is not applicable to Provisioning Profiles.

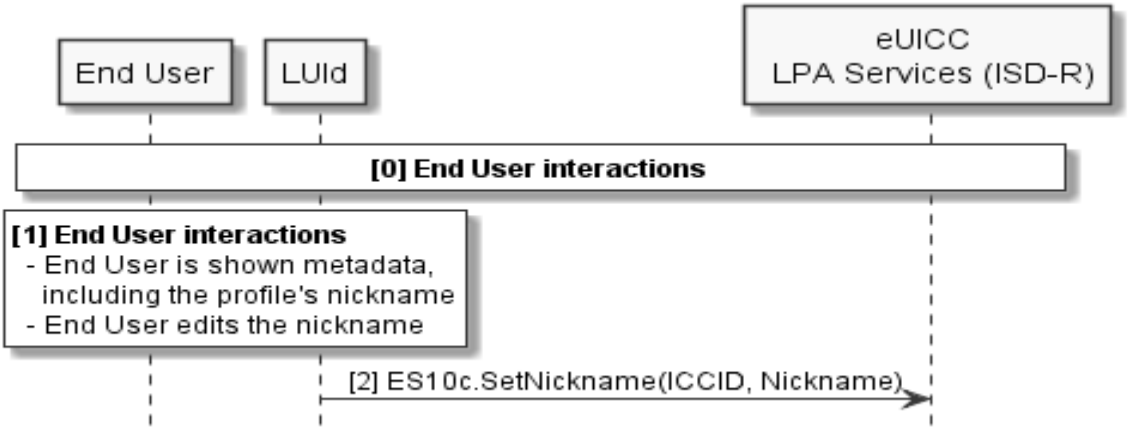


Figure 21: Set/Edit Nickname

Start Conditions:

When the Profile to be renamed is an Operational Profile:

- End User Intent is verified as defined in SGP.21 [4] for Simple Confirmation.

When the Profile to be renamed is a Test Profile:

- The Device is in Device Test Mode.

Procedure:

0. The End User is presented a user interface that displays the list of installed Profiles within the eUICC, with their current states (Enabled or Disabled), as described in "List Profiles" procedure (section 3.2.4) The End User selects the Profile to be modified.
1. Through the LUId, the End User:

   a) Is shown the relevant Profile Metadata, including the associated Profile Nickname (if any), along with the Profile status.
   b) Edits the Profile Nickname.

2. The LUId calls the function "ES10c.SetNickname" with the relevant ICCID and edited Nickname.

End Conditions:

The new Profile Nickname is stored in the Profile Metadata of the relevant Profile.

3.3 Local eUICC Management

3.3.1 Retrieve EID

The EID SHALL be made available for the End User on the user interface by the LPAd. The EID is retrieved by the LPAd over the ES10c interface using the function "ES10c.GetEID" as described in section 5.7.20.

As appropriate for the type of Device, the LPA SHOULD present the EID to the End User as a text string and/or a QR code. The text representation SHALL comprise 32 digits, where each digit is represented by one character in the set [0123456789]. The QR code representation SHALL also be prefixed with "`EID:`" and SHALL be encoded in alphanumeric mode according to ISO/IEC 18004 [15].

When included in an Octect16 ASN.1 object, the first, third, fifth… digits SHALL be put into the highest four bits of the first, second, third… bytes.

NOTE: Presentation of the EID on the package of a Device should use the same QR code format. The EID should also be printed on the package as a barcode.

3.3.2 eUICC Memory Reset

This procedure is used to delete all the Operational Profiles and their associated Profile Metadata stored on the eUICC regardless of their status. The procedure is initiated by the End User using the LUI of the LPAd.

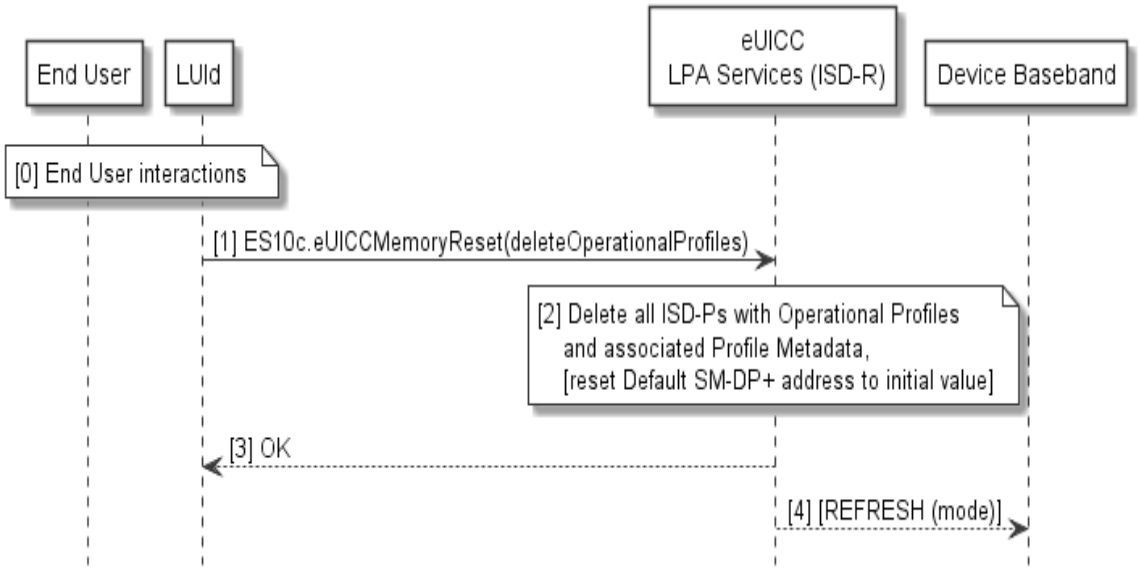


Figure 22: eUICC Memory Reset

Start Conditions:

- End User Intent is verified as defined in SGP.21 [4] for Authenticated Confirmation.

Procedure:

0. The End User initiates the eUICC Memory Reset and acknowledges the consequences.

1. The LPAd SHALL call the "ES10c.eUICCMemoryReset" function of the ISD-R as defined in section 5.7.19, specifying that Operational Profiles are to be erased.
2. If there is an Enabled Operational Profile and a proactive session is ongoing:
    a) The function SHALL return the appropriate error code and stop its execution.
    b) The LPAd MAY take implementation-dependent actions to terminate the proactive command session, and MAY call again the "ES10c.eUICCMemoryReset" function without any further End User interaction.

Otherwise the ISD-R SHALL:

    c) Delete all ISD-Ps with Operational Profiles and their associated data and Profile Metadata.
    d) If required by the command: reset the Default SM-DP+ address to its initial value.
3. The ISD-R SHALL return a response indicating result OK back to the LUId.
4. If there was an Enabled Profile, the ISD-R SHALL send a REFRESH proactive command to the Device.

End Conditions:

The Operational Profiles and their associated Profile Metadata are deleted from the eUICC.

3.3.3 eUICC Test Memory Reset

This procedure is used to delete all the field-loaded (non-preloaded) Test Profiles and their associated Profile Metadata stored on the eUICC regardless of their status. The procedure is initiated by the End User using the LUI while the Device is in Test Mode.

This procedure is only required if the Device supports Test Mode.

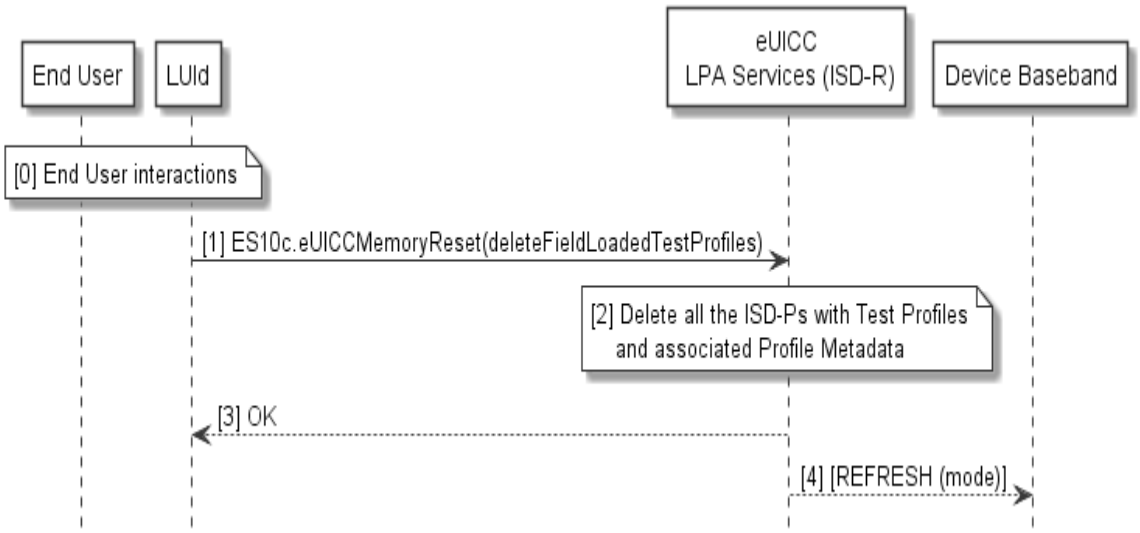


Figure 23: eUICC Test Memory Reset

Start Conditions:

- The Device is in Test Mode
- End User Intent is verified as defined in SGP.21 [4] for Simple Confirmation.

Procedure:

0. The End User initiates the eUICC Test Memory Reset and acknowledges the consequences.
1. The LPAd SHALL call the "ES10c.eUICCMemoryReset" function of the ISD-R as defined in section 5.7.19, specifying that field-loaded (non-preinstalled) Test Profiles are to be erased.
2. If the eUICC does not support Test Profiles then the ISD-R SHALL return an OK result to the LPA and the procedure SHALL be stopped. Otherwise:
    a) If there is an Enabled Test Profile and a proactive session is ongoing:
        i. The function SHALL return the appropriate error code and stop its execution.
        ii. The LPAd MAY take implementation-dependent actions to terminate the proactive command session, and MAY call again the "ES10c.eUICCMemoryReset" function without any further End User interaction.
    b) Otherwise, the ISD-R SHALL delete all the selected ISD-Ps with their Profiles and their associated data and Profile Metadata.
3. The ISD-R SHALL return a response indicating result OK back to the LUId.
4. If there was an Enabled Profile, the ISD-R SHALL send a REFRESH proactive command to the Device.

End conditions:

The Test Profiles and their associated Profile Metadata are deleted from the eUICC.

3.3.4 Set/Edit Default SM-DP+ Address

This procedure is used to set or update the Default SM-DP+ address set in the eUICC.

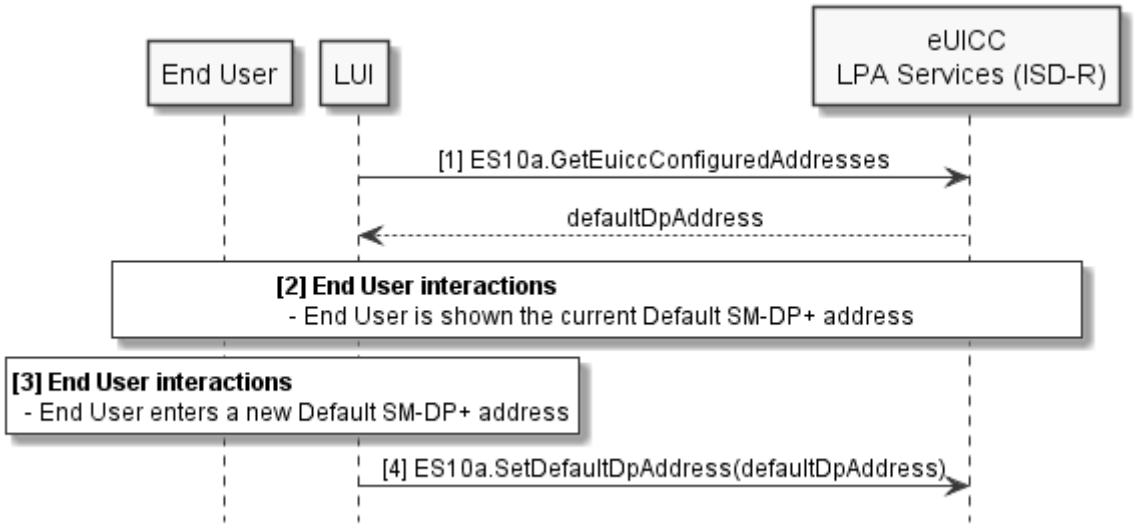


Figure 23a: Set/Edit Default SM-DP+ Address

Start Conditions:

- End User Intent is verified as defined in SGP.21 [4] for Simple Confirmation.

Procedure:

1. The LUId calls the function "ES10a.GetEuiccConfiguredAddress" to retrieve the Default SM-DP+ address currently set in the eUICC. The Default SM-DP+ address MAY be an empty value.
2. The End User is presented a user interface that displays the current Default SM-DP+ address.
3. Through the LUId, the End User enters a new Default SM-DP+ address. The LUId SHALL allow to set an empty value.
4. The LUId calls the function "ES10a.SetDefaultDpAddress" with the new Default SM-DP+ address.

End Conditions:

The Default SM-DP+ address is updated with the value set by the End User.

3.4 Device and eUICC Initialisation

3.4.1 eUICC Initialisation

The eUICC SHALL indicate its support of eUICC functionality in ATR Global Interface byte as defined in ETSI TS 102 221 [6]. If the indication is received by the LPAd, the LPAd MAY obtain additional eUICC information, such as SVN.

The eUICC initialisation SHALL follow the procedure as defined in ETSI TS 102 221 [6]. If the eUICC contains an Enabled Profile, the eUICC initialisation procedure SHALL be completed.

If the eUICC does not contain an Enabled Profile, but only a limited file system as described in section 3.4.3, the Device SHALL be able to initialise the eUICC and to perform a Terminal profile command indicating at least that REFRESH (UICC Reset Mode) proactive command is supported.

3.4.2 RSP Device Capabilities

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

The Device SHALL report its support of LPA functions using the Terminal Capability command data defined in ETSI TS 102 221 [6]. This command SHALL be sent before the SELECT ISD-R command defined in section 5.7.1.

Within the Terminal Capability template (tag 'A9'), the tag '83' is used for indicating the Device’s support for eUICC related functions.

The LPA support is indicated in the first byte within the TLV object under tag '83':

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
-	-	-	-	-	-	-	1	Local User Interface (LUId) for eUICC supported
-	-	-	-	-	-	-	0	Local User Interface (LUId) for eUICC not supported
-	-	-	-	-	-	1	-	Local Profile Download (LPDd) for eUICC supported
-	-	-	-	-	-	0	-	Local Profile Download (LPDd) for eUICC not supported

-	-	-	-	-	1	-	-	Local Discovery Service (LDSd) for eUICC supported
-	-	-	-	-	0	-	-	Local Discovery Service (LDSd) for eUICC not supported
-	-	-	-	1	-	-	-	LUIe based on SCWS supported
-	-	-	-	0	-	-	-	LUIe based on SCWS not supported
x	x	x	x	x	-	-	-	RFU

Table 7: eUICC-related Device Capabilities

Subsequent bytes are RFU.

For LPAd implementations according to this version of the specification, b1, b2 and b3 SHALL either all be set to 1 or all be set to 0.

The eUICC SHALL only enable the functions of ES10c if the Device indicates support for the LUId.

The eUICC SHALL only enable the functions of ES10b if the Device indicates support for the LPDd.

The eUICC SHALL only enable the functions of ES10a if the Device indicates support for the LDSd.

The conditions for enabling the LPAe are defined in section 5.7.1.

3.4.3 eUICC File Structure

If there is no Enabled Profile on the eUICC, the eUICC SHALL ensure a default file system is available to the Device. This file system SHALL contain at least the MF and MAY contain the MF-level EFs shown below.

- $EF_{ENV\text{-}CLASSES}$
- EF_{UMPC}

It SHALL not be possible to modify either file via ES6 of a Profile.

$EF_{ENV\text{-}CLASSES}$ SHALL never be present in any Profile Package, however if present, it SHALL be ignored by the eUICC when installing the Profile.

EF_{UMPC} MAY be present in a Profile Package. If present and EF_{UMPC} is also present in the default file system, the second byte of the default file SHALL be modified by the content of the Profile Package when this Profile is enabled. The eUICC SHALL ignore the content of all the other bytes of the file present in the Profile Package. If present and EF_{UMPC} is not present in the default file system, the whole file from the Profile SHALL be taken into account.

When a Profile is enabled, the eUICC SHALL present a file system comprising that Profile's file system and the EFs listed above if existing.

3.4.4 Device Power-on Profile Discovery

When appropriate for the class and usage of the device, the LPA SHALL conditionally perform Profile discovery when the Device is powered on, rebooted, or reset. In addition the

LPA MAY support an End User configurable parameter that enables or disables this operation.

When it is supported, the initial value of the configuration parameter SHALL be 'Enabled', and its value SHALL be persistent across Device reset and power cycles.

The specific point at which power-on Profile discovery occurs and the means by which the LPA is launched to perform Profile discovery are Device-specific and out of the scope of this specification. It SHALL be performed if all of the following conditions are satisfied:

- Power-on Profile discovery is appropriate for the class and usage of the Device. (For example, this could be inappropriate for an open-market cellular-enabled notebook computer).
- No Operational Profile is installed on the eUICC.
- The value of the configuration parameter is 'Enabled'.

When all of these conditions are satisfied the LPA SHALL perform the following steps:

1. If there is a configured Default SM-DP+ address, then the LPA SHALL initiate the Profile download and installation procedure as defined in section 3.1.3, using the default SM-DP+ address and an empty string for the Matching ID.
2. If no Operational Profile was downloaded in step 1 and there is a configured SM-DS address, then the LPA SHALL initiate the event retrieval procedure as defined in section 3.6.2, with no EventID, and process any retrieved Profile download(s).
3. If no Operational Profile was downloaded in either steps 1 or 2, the LPA MAY prompt the End User to add a Profile using an Activation Code (such as by manual entry or QR code scanning).

The LPA SHALL verify User Intent for the resulting Profile download(s), if any. This SHALL be an Authenticated Confirmation when this does not occur during Device setup. The means by which the LPA detects Device setup is out of the scope of this specification.

3.5 Notifications

This section describes the procedure to provide a report (in the context of this procedure referred to as "Notification") to a remote entity (Recipient Address) that a Profile Management Operation is successfully performed on the eUICC.

Two types of Notifications are defined: Profile Installation Results, generated as an answer to the Profile installation, see section 2.5.6; and Other Notifications, generated due to the 'NotificationConfigurationInformation' data object defined in the "ES8+.StoreMetadata" function. The eUICC SHALL manage the storage of these two types of Notification separately.

A Notification is made up of the following fields:

- Sequence Number (generated by the eUICC)
- Profile Management Operation (i.e. the event whose occurrence SHALL be notified)
- Recipient Address
- ICCID
- Additional data in case of a Profile Installation Result, see section 2.5.6

- eUICC signature

Sequence Number is used to protect the recipient against replay attacks. Each time a new Notification is generated by the eUICC, the Sequence Number SHALL be incremented. There SHALL be a single Sequence Number counter per eUICC, which SHALL be used across all Profiles and for both types of Notifications. Neither an eUICC Memory Reset nor eUICC reset SHALL affect this Sequence Number.

The ICCID is used to identify the Profile upon which the operation, leading to the generation of the Notification, has been performed. ICCID SHALL be set if known by the eUICC.

Each Profile Metadata MAY contain Notification Configuration Information for Other Notifications, made up of the following:

- Profile Management Operation
- Recipient Address

When the Profile Management Operation indicated in the Notification Configuration Information is performed, the eUICC SHALL generate and store a Notification.

When an eUICC Memory Reset is performed, a Delete Notification SHALL be generated for all the deleted Profiles for which the delete Profile is indicated in the Notification Configuration List.

When an Enable Profile is performed and the currently enabled Operational Profile is implicitly disabled as a consequence of the enable Profile, the Notifications on both the disable Profile and enable Profile SHALL be generated, provided that each Operation is indicated in the Notification Configuration List.

The LPA retrieves the pending Notifications and SHALL send the Notifications one at a time. Each Notification is sent on a best-effort basis, as described below, to the recipient when connectivity is available.

After the LPA receives the acknowledgement from the SM-DP+ that the Notification was received, it SHALL communicate to the eUICC to remove that Notification.

When the eUICC needs to store a new Other Notification, if there is not enough room the eUICC SHALL delete one or more of the previously stored Other Notifications in order of their Sequence Number, beginning with the lowest.

The LPA SHALL sort the Notifications and group them by SM-DP+. The LPA SHALL send Notifications of a group one by one according to the sequence number, lowest number (oldest Notification) first. The next Notification of a group SHALL not be sent until LPA receives a success or failure response from the SM-DP+ for the previous Notification. The LPA MAY send Notifications from different groups sequentially or in parallel (i.e. there is no need to wait for the acknowledgment of a Notification from one group before sending a Notification from another group).

This specification does not define the usage of the Notifications received by the SM-DP+, but the SM-DP+ SHOULD only disclose data as agreed with the Profile Owner.

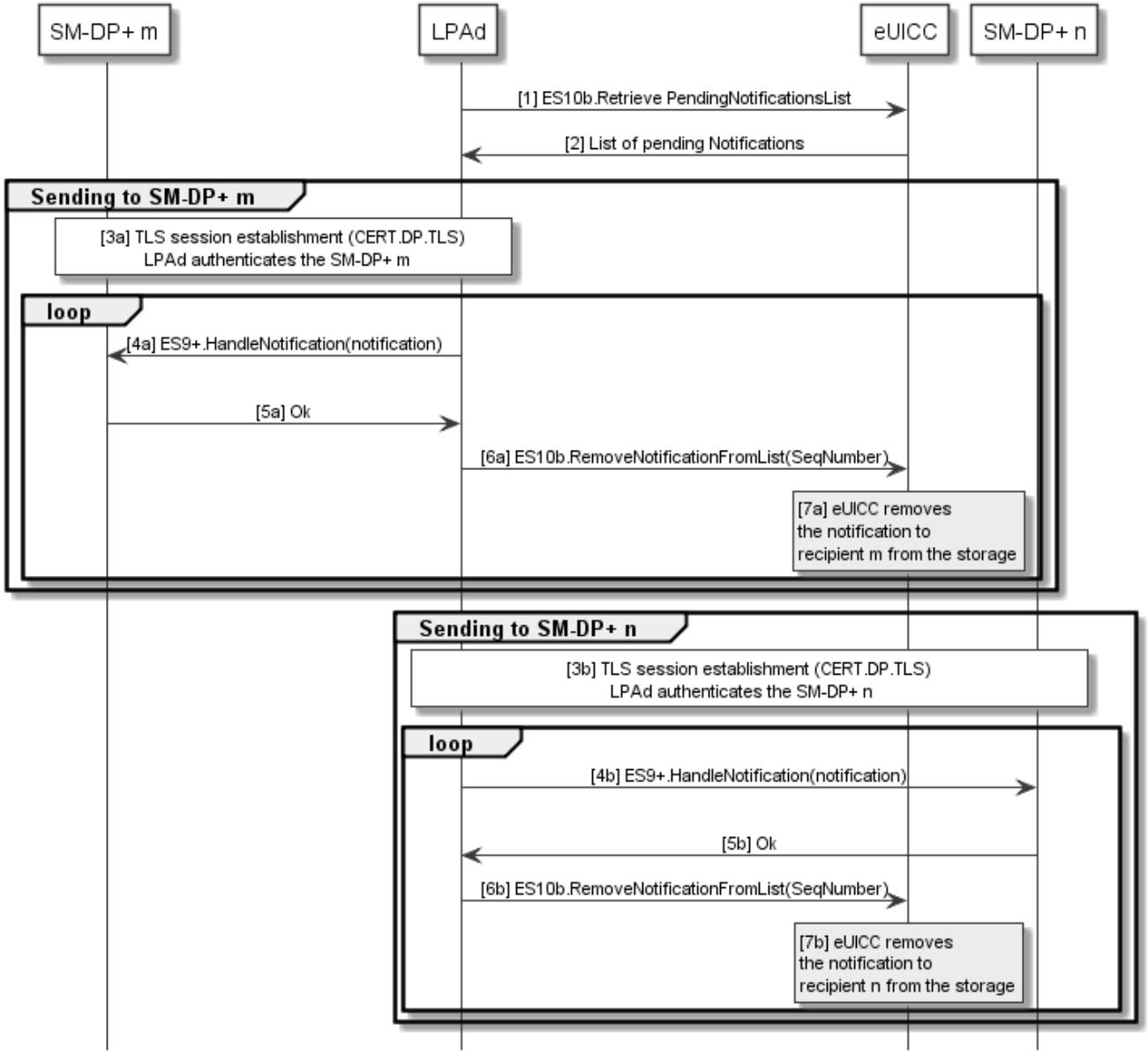


Figure 24: Sending of Notifications

The figure above illustrates the sending of Notifications to two distinct SM-DP+ in a sequence manner for representation easiness. But the LPA MAY sends Notifications to SM-DP+ n and SM-DP+ m in parallel.

Start Conditions:

A Profile has been Enabled, Disabled, Installed or Deleted.

Procedure:

1. The LPAd queries the eUICC for the Pending Notifications List.
2. The eUICC provides the LPAd with the Pending Notifications List.
3. The LPAd establishes a TLS secure channel with the relevant SM-DP+.
4. The LPAd sends each Notification to the SM-DP+.
5. The SM-DP+ acknowledges Notification reception.

6. The LPAd calls the "ES10b.RemoveNotificationFromList" function.
7. The eUICC removes the Notification from the Pending Notifications List.

Steps 4 – 7 SHALL be repeated per each Notification in the Pending Notifications List.

3.6 SM-DS

3.6.1 Event Registration

For Profile download event registrations, the SM-DP+ SHALL set the ForwardingIndicator to true.

3.6.1.1 Event Registration without Cascade

This procedure applies when the SM-DP+ is directly connected to the Root SM-DS.

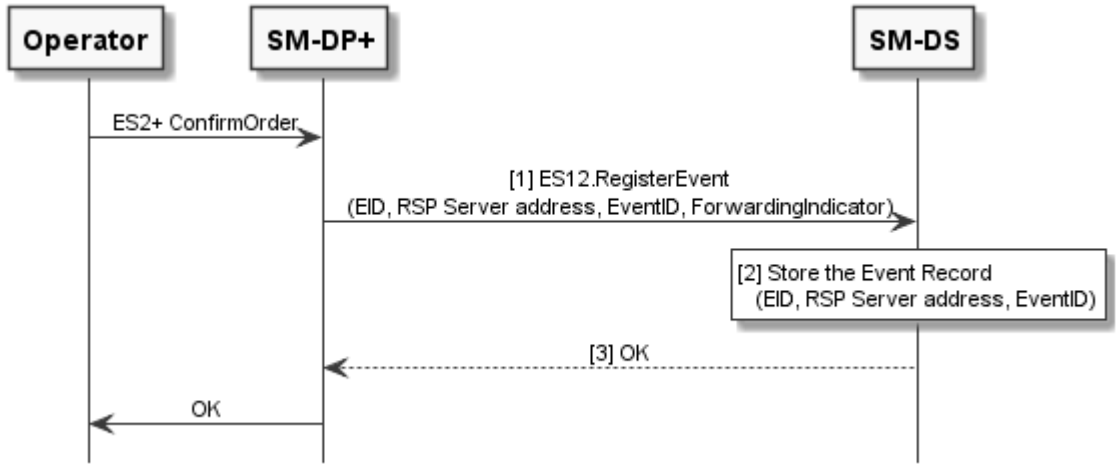


Figure 25: Event Registration Procedure without Cascade

Start Conditions:

The Operator places a Profile download to the SM-DP+ with a Root SM-DS Address.

The SM-DP+ generates an EventID that is used to uniquely identify within its context the Profile download order.

EventIDs SHALL be unique per SM-DP+ and SHALL not be reused.

NOTE: This allows the LPA to keep a trace of already processed events and detect events still pending at an SM-DS that have been already processed.

The SM-DP+ and SM-DS are mutually authenticated. The SM-DP+ OID has been retrieved from the TLS certificate used for mutual authentication.

Procedure:

1. The SM-DP+ calls "ES12.RegisterEvent" function including EID, RSP Server address, EventID, and ForwardingIndicator.

2. The SM-DS stores the received Event Record, consisting of EID, RSP Server address, and EventID together with the SM-DP+ OID retrieved from the SM-DP+ certificate. The value of ForwardingIndicator SHALL be ignored by the Root SM-DS.
3. The SM-DS acknowledges the registration.

3.6.1.2 Event Registration with Cascade

This procedure applies when the SM-DP+ is only connected to an alternative SM-DS, which in turn is connected to the root SM-DS, and the ForwardingIndicator is set to true.

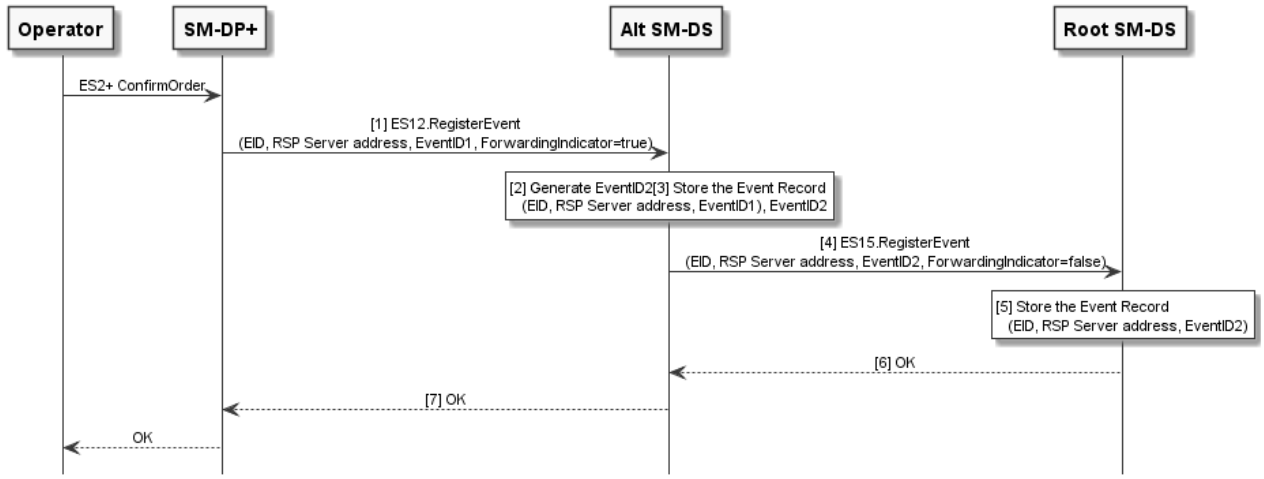


Figure 26: Event Registration Procedure with Cascade

Start Conditions:

This procedure applies when the SM-DP+ is only connected to an alternative SM-DS, which in turn is connected to the root SM-DS.

The requirements for EventIDs in section 3.6.1.1 SHALL also apply to EventID1 and EventID2; the latter SHALL be unique, and not be re-used, per Alternative SM-DS.

Procedure:

1. The SM-DP+ calls "ES12.RegisterEvent" function including EID, RSP Server address of SM-DP+, EventID1, and ForwardingIndicator set to 'true'.
2. As the ForwardingIndicator indicates forwarding of the registration, the Alternative SM-DS generates a new EventID2.
3. The Alternative SM-DS stores the received Event Record, consisting of EID, RSP Server address, and EventID1 together with EventID2 and the SM-DP+ OID retrieved from the SM-DP+ certificate.
4. The Alternative SM-DS calls "ES15.RegisterEvent" function of the Root SM-DS including EID, RSP Server address of the Alternative SM-DS, generated EventID2 and ForwardingIndicator set to 'false'.
5. The Root SM-DS stores the received Event Record, consisting of EID, RSP Server address, and EventID2 together with the SM-DS OID retrieved from the Alternative SM-DS certificate.
6. The Root SM-DS acknowledges the registration.
7. The Alternative SM-DS acknowledges the registration.

3.6.2 Event Retrieval

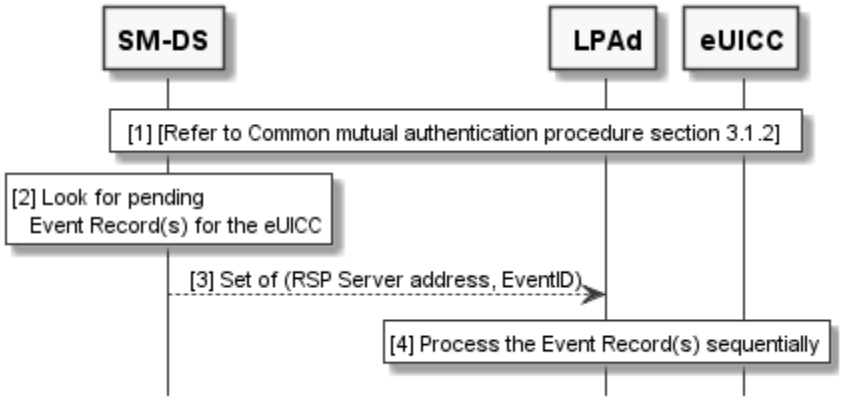


Figure 27: Event Retrieval Procedure

Start Conditions:

In addition to the start conditions required by the common mutual authentication procedure defined in section 3.1.2, Event(s) are registered on the SM-DS by one or more SM-DP+(s)/SM-DS(s).

The event retrieval procedure is used in the following cases:

a) To retrieve Events from an SM-DS when there is no EventID. This includes, but is not limited to, the following trigger conditions:

   - The End User MAY manually query for pending Event Records. The LUI MAY implement this query in combination with other related operations, for example, as a composite 'Add Profile' operation.
   - The LPA MAY query the SM-DS as part of Device power-on Profile discovery as described in section 3.4.4.

b) To retrieve an Event from an SM-DS with a specific EventID. This corresponds to the retrieval of a cascaded Event from the alternative SM-DS.

Procedure:

1. The common mutual authentication procedure defined in section 3.1.2 SHALL be executed. When this procedure is used for SM-DS, SM-XX, CERT.XXauth.ECDSA, PK.XXauth.ECDSA, SK.XXauth.ECDSA, and ESXX are SM-DS, CERT.DSauth.ECDSA, PK.DSauth.ECDSA, SK.DSauth.ECDSA, and ES11, respectively.

   In addition, the LPAd SHALL build the ctxParams1 data object to provide the MatchingID and Device Info to the eUICC for signature. The value of the MatchingID SHALL be set as follows:

   - For case a), the MatchingID value SHALL be empty.
   - For case b), the MatchingID value SHALL be set to the EventID found in the Event Record that LPAd is processing.

2. After having successfully authenticated the eUICC at the end of the step (1), the SM-DS SHALL check if there are pending Event Records matching the following criteria:

   - If MatchingID is empty, the EID in an Event Record matches the EID in the CERT.EUICC.ECDSA obtained during step (1) (Case a)).
   - If MatchingID has a value: the EventID2 and the EID in an Event Record match the EventID in the MatchingID and the EID in the CERT.EUICC.ECDSA obtained during step(1) (case b)).

   If no Event Record is found an appropriate status is returned.

3. The SM-DS responds back to the LPAd with the RSP Server address and EventID pair(s) of the pending Event Record(s), if any. The RSP Server address and EventID pair(s) SHALL be ordered as registered in the SM-DS.
4. The LPAd SHALL process the received Event Records in the order received in step (3), by sequentially contacting each RSP Server with the corresponding EventID.

3.6.3 Event Deletion

3.6.3.3 Event Deletion without Cascade

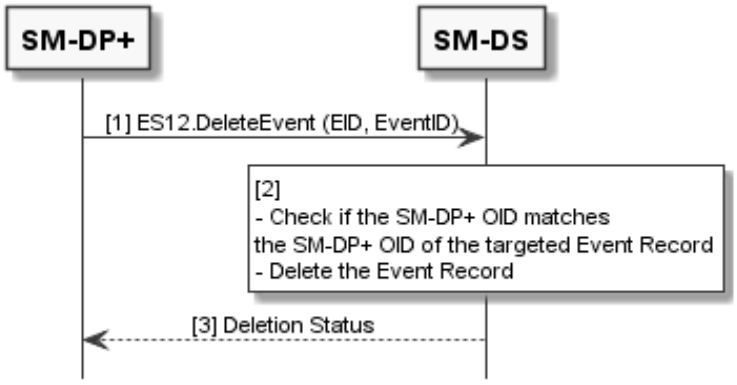


Figure 28: Event Deletion Procedure without Cascade

Start Conditions:

An Event Record is stored in the SM-DS, which is identified by the EID, EventID and SM-DP+ OID.

The related Registration was not cascaded.

The SM-DP+ and SM-DS are mutually authenticated. The SM-DP+ OID has been retrieved from the TLS certificate used for mutual authentication.

Procedure:

1. The SM-DP+ calls "ES12.DeleteEvent" function including the EID and EventID.
2. The SM-DS SHALL delete the Event Record identified by the SM-DP+ OID and EventID. If no Event Record is found, the SM-DS SHALL return the relevant error status.

3. The SM-DS SHALL return the deletion status.

3.6.3.4 Event Deletion with Cascade

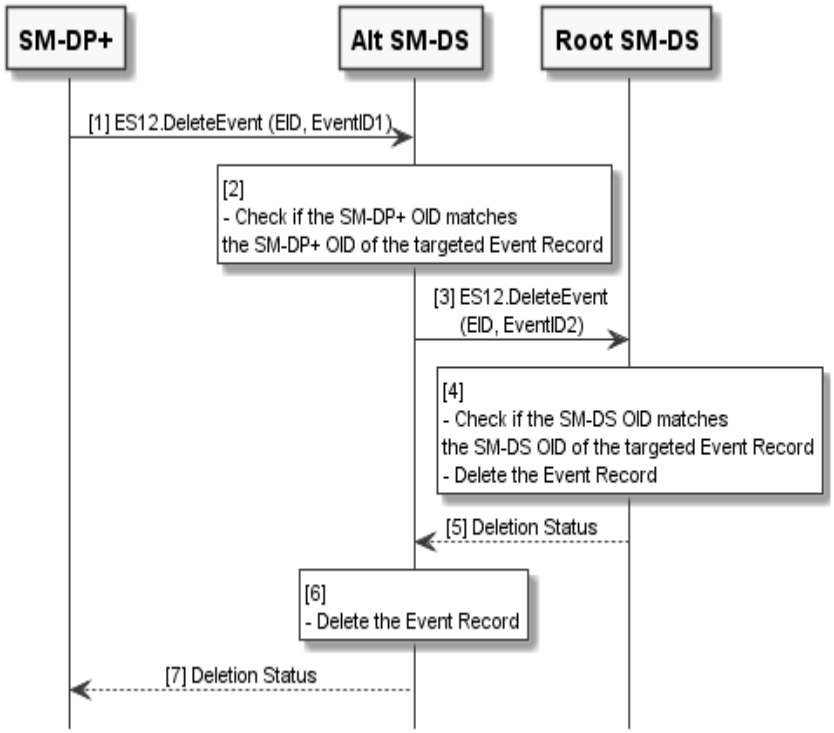


Figure 29: Event Deletion Procedure with Cascade

Start Conditions:

Start condition is same as for section 3.6.3.1 except the following:

The related Registration was cascaded.

Procedure:

1. The SM-DP+ calls "ES12.DeleteEvent" function including the EID and EventID1.
2. The Alternative SM-DS SHALL check if there is a stored Event Record identified by the SM-DP+ OID and EventID1. If no Event Record is found, the SM-DS SHALL return the relevant error status.
3. As EventID2 is not empty, i.e. the Alternative SM-DS previously forwarded the registration according to 3.6.1.2, the Alternative SM-DS calls "ES12.DeleteEvent" function including the EID and EventID2 which were used in the registration to the Root SM-DS.
4. The Root SM-DS SHALL delete the Event Record identified by the SM-DS OID and EventID2. If no Event Record is found, the Root SM-DS SHALL return the relevant error status.
5. The Root SM-DS SHALL return the deletion status.
6. If the deletion of the Event Record in the Root SM-DS is successful, the Alternative SM-DS SHALL delete the associated Event Record from its storage.
7. The Alternative SM-DS SHALL return the deletion status.

4 Data Elements

4.1 Activation Code

The Activation Code SHALL be coded to be the concatenation of the following strings listed in the following table:

Name	MOC	Description
AC_Format	M	Format of the Activation Code. SHALL be set to "1" for this Format of the Activation Code and any subsequent backward compatible Format
Delimiter	M	SHALL be set to "$"
SM-DP+ Address	M	FQDN (Fully Qualified Domain Name) of the SM-DP+ (e.g., smdp.gsma.com) restricted to the Alphanumeric mode character set defined in table 5 of ISO/IEC 18004 [15] excluding '$'
Delimiter	M	SHALL be set to "$"
AC_Token	M	MatchingID as described in section (4.1.1)
Delimiter	C	SHALL be present and set to "$" if any of the following optional parameters is present
SM-DP+ OID	O	SM-DP+ OID in the CERT.DPauth.ECDSA
Delimiter	C	SHALL be present and set to "$" if any of the following optional parameters is present
Confirmation Code Required Flag	O	SHALL be present and set to "1" if Confirmation Code is required; otherwise it SHALL be absent

Table 8: Activation Code Structure

The maximum length of the Activation Code SHALL be 255 characters, but in practise it is recommended to consider the user experience when choosing the length.

A Phase 1 Device SHALL ignore a delimiter and any further parameters beyond the "Confirmation Code Required Flag".

If a Phase 1 Device encounters an AC_Format other than "1" the Activation Code SHALL be treated as invalid.

Examples of the Activation Code are as follows:

- 1$SMDP.GSMA.COM$04386-AGYFT-A74Y8-3F815
  (if SM-DP+ OID and Confirmation Code Required Flag are not present)
- 1$SMDP.GSMA.COM$04386-AGYFT-A74Y8-3F815$$1
  (if SM-DP+ OID is not present and Confirmation Code Required Flag is present)
- 1$SMDP.GSMA.COM$04386-AGYFT-A74Y8-3F815$1.3.6.1.4.1.31746$1
  (if SM-DP+ OID and Confirmation Code Required flag are present)
- 1$SMDP.GSMA.COM$04386-AGYFT-A74Y8-3F815$1.3.6.1.4.1.31746
  (If SM-DP+ OID is present and Confirmation Code Required Flag is not present)
- 1$SMDP.GSMA.COM$$1.3.6.1.4.1.31746

(If SM-DP+ OID is present, Activation token is left blank and Confirmation Code Required Flag is not present)

When entered manually, the Activation Code SHALL be used as defined above.

When provided in a QR code according to ISO/IEC 18004 [15], the Activation Code SHALL be prefixed with "LPA:"

4.1.1 Matching ID

The MatchingID is mandatory information (but MAY be zero-length) that SHALL be set-up between the Operator and the SM-DP+, to identify the context of a specific management order given to the SM-DP+. The MatchingID is generated during the download initiation procedure (section 3.1.1).

The MatchingID is equivalent to the "Activation Code Token" as defined is SGP.21 [4].

The format and content of the MatchingID is subject to the following constraints:

The MatchingID, when not a zero-length value, SHALL be a unique identifier in the context of the Operator and the SM-DP+ to:

- Match a download order initiated by the Operator with a Profile Download request coming from an LPD.
- As a protection for the SM-DP+: the SM-DP+ SHALL only process requests containing a MatchingID known to the SM-DP+ (and therefore inherently valid).

It SHALL consist only of upper case alphanumeric characters (0-9, A-Z) and the "-" in any combination.

NOTE: This selection allows more compact alphanumeric QR code encoding and is expected to be supported for manual entry.

4.2 Device Information

During the Profile download and installation procedure, any Device Information provided by the LPA to the eUICC SHALL be signed by the eUICC, and then provided by the eUICC to the SM-DP+ for the purpose of Device eligibility check. The SM-DP+/Operator is free to use or ignore this information at their discretion.

Device Information includes:

- Device type allocation code: TAC
- Device capabilities: The Device SHALL set all the capabilities it supports
  - Radio access technologies, including release.
  - Contactless: the SWP and HCI interfaces as well as the associated APIs
  - Optional RSP functions supported:
    - Update CRL on the eUICC
- IMEI (optional)

Device Information

DeviceInfo is coded using ASN.1 DER as follows:

```
DeviceInfo ::= SEQUENCE {
   tac Octet4,
   deviceCapabilities DeviceCapabilities,
   imei Octet8 OPTIONAL
}

DeviceCapabilities ::= SEQUENCE { -- Highest fully supported release for each
definition
  -- The device SHALL set all the capabilities it supports
   gsmSupportedRelease VersionType OPTIONAL,
   utranSupportedRelease VersionType OPTIONAL,
   cdma2000onexSupportedRelease VersionType OPTIONAL,
   cdma2000hrpdSupportedRelease VersionType OPTIONAL,
   cdma2000ehrpdSupportedRelease VersionType OPTIONAL,
   eutranSupportedRelease VersionType OPTIONAL,
   contactlessSupportedRelease VersionType OPTIONAL,
   rspCrlSupportedVersion VersionType OPTIONAL
}
```

The TAC and IMEI are defined in 3GPP TS 23.003 [35].

The TAC SHALL be represented as a string of 4 octets that is coded as a Telephony Binary Coded Decimal String as defined in 3GPP TS 29.002 [63].

The IMEI (including the check digit) SHALL be represented as a string of 8 octets that is coded as a Telephony Binary Coded Decimal String as defined in 3GPP TS 29.002 [63], except that the last octet contains the check digit (in high nibble) and an 'F' filler (in low nibble). It SHOULD be present if the Device contains a non-removable eUICC.

The capabilities SHALL be represented as follows:

- **gsmSupportedRelease** – if GSM/GERAN is supported, this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If GSM/GERAN is not supported this SHALL not be present.
- **utranSupportedRelease** – if UMTS/UTRAN is supported, this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If UMTS/UTRAN is not supported this SHALL not be present.
- **cdma2000onexSupportedRelease** – if cdma2000 1X is supported, this SHALL be encoded as the octet string {1, 0, 0}. If cdma2000 1X is not supported this SHALL not be present.
- **cdma2000hrpdSupportedRelease** – if cdma2000 HRPD is supported, this SHALL be encoded as the octet string {R, 0, 0}. If cdma2000 HRPD is not supported this SHALL not be present. The value *R* SHALL represent the EVDO revision as follows:
    - Rev 0 SHALL be encoded as 1
    - Rev A SHALL be encoded as 2
    - Rev B SHALL be encoded as 3
- **cdma2000ehrpdSupportedRelease** – if cdma2000 eHRPD, is supported this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If cdma2000 eHRPD is not supported this SHALL not be present.

- **eutranSupportedRelease** – if LTE/E-UTRAN is supported, this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If LTE/E-UTRAN is not supported this SHALL not be present.
- **contactlessSupportedRelease** – if NFC is supported, this SHALL be the highest (*version*, *revision*) number of TS.26 [40], encoded as the octet string {*version*, *revision*, 0}. If NFC is not supported this SHALL not be present.
- **rspCrlSupportedVersion** – if load eUICC CRL as defined in section 5.7.12 is supported, this SHALL be the highest SGP.22 release number supported by the device. If this function is not supported, this field SHALL not be present.

4.3 eUICC Information

During the Profile download and installation procedure, eUICC information needs to be provided to the LPA and forwarded to the SM-DP+. The LPA MAY request eUICC information either in preparation of the Profile download, or at any time in order to obtain the eUICC firmware versions, available non-volatile memory and eUICC capabilities for the Device. eUICC information SHALL be generated by the eUICC and is intended to be signed:

- Profile Package Version: Indicates the highest version number of the SIMalliance eUICC Profile Package: Interoperable Format Technical Specification [5] supported by the eUICC.
- SVN: Indicates the highest Specification Version Number of this specification supported by the eUICC. The SVN SHALL have the same three digit number as the highest supported specification version. Example of value: '2.0.0'.
- Firmware version: indicates the version information of the eUICC's platform and the OS, defined as for the EID in SGP.02 [2]. This value is issuer specific.
- Available amount of non-volatile memory: Indicates the current total available memory for Profile download and installation. The value is expressed in bytes.
- UICC capabilities: Contains the UICC capabilities supported by the eUICC.
- Java card version: optional, indicates the latest version of ETSI TS 102 241 [53], if supported by the eUICC.
- GlobalPlatform version: optional, indicates the latest version of GlobalPlatform Card Specification [8] supported by the eUICC, if different from the one referenced in this specification.
- RSP capabilities: Contains the optional RSP capabilities supported by the eUICC.
- List of supported GSMA CI Key Identifiers for RSP Server signature verification.
- List of GSMA CI Key Identifiers for which eUICC has a signed certificate that can be used for signature verification by the RSP Server.
- Category: optional, indicates the eUICC category as described below.

Category
Basic eUICCs SHALL be compliant with at least the following features: • Memory size available when no Profiles are installed (EEPROM) : 64kB • ISO interface PPS 96 • BIP over HTTPS features

Medium eUICCs SHALL be compliant with at least the following features: • Memory size available when no Profiles are installed (EEPROM) : 384kB • ISO interface PPS 97 • BIP over HTTPS features • Processor >= 25MHz • Crypto processor >= 100MHz • Memory Protection Unit
Contactless eUICCs SHALL be compliant with at least the following features: • Memory size available when no Profiles are installed (EEPROM) : 1024kB • ISO interface PPS 97 • BIP over HTTPS features • Processor >= 25MHz • Crypto processor >= 100MHz • Memory Protection Unit • In combination with an appropriate enabled Operational NFC Profile, a contactless eUICC SHALL support all requirements specified in the SGP.03 GSMA NFC UICC Requirements Specification [55]

The eUICC information comprises EUICCInfo1 and EUICCInfo2 as defined in Annex H.

4.4 Profile Metadata

During the Profile download and installation procedure, Profile Metadata needs to be provided to the LPAd for display and to the eUICC. Profile Metadata is generated by the SM-DP+ in plain text to be readable by the LPA. Profile Metadata is also contained protected in BPP to be loaded into the eUICC, so that the LPA will be able to access the same information any time after the Profile has been successfully loaded into the eUICC, using the "ES10c.GetProfilesInfo" function.

Profile Metadata values, like any other Profile data, are under the responsibility of, and defined by, the Profile owner. Profile Metadata is communicated to the SM-DP+ by means which are out of scope of this specification.

Profile Metadata includes:

- ICCID of the Profile
- Profile Name (corresponds to "Short description" in SGP.21 [4]) as a plain text information: content free information defined by the Operator/Service Provider
- Operator/Service Provider name, as a plain text information: content free information defined by the Service Provider (e.g. 'Orange', 'AT&T'…)
- End User’s Profile Nickname
- Icon

- Profile Class: indicates the sort of profile among the defined values: 'Test', 'Operational' and 'Provisioning' (section 4.4.1)
- Notification Configuration Information, defined in section 3.5 "Notifications"
- Profile owner, including MCC, MNC, GID1 and GID2 if the Profile is not PIN protected
- Profile Policy Rules (PPRs). See section 2.9.1 and 4.4.2

Profile Metadata is merely to be displayed to the End User to provide information about the Profile to be installed. But it is out of scope of this implementation what the LPA does exactly with this Profile Metadata, e.g. the LPA can display all or only part of this information.

4.4.1 Profile Class

A Profile can be defined as a Test Profile, an Operational Profile or a Provisioning Profile. The Profile Class is set in the Profile Metadata and indicates to the LPA and the eUICC which rules to apply.

4.4.2 Profile Policy Rules

The PPRs are provided within the ES8+.StoreMetadata function of the Bound Profile Package. The `pol` field of the `ProfileHeader` PE of the SIMalliance Profile Package (UPP in section 2.5.2) SHALL not be used.

The PPRs defined in this document are coded using the ASN.1 data type `PprIds`, see section 2.8.1.1. `pprUpdateControl` has no meaning when provided in ES8+.StoreMetadata.

4.5 Keys and Certificates

This section describes keys and certificates used in this specifications.

4.5.1 Cryptographic Keys

Key name	Key name	Nature
PK.EUICC.ECDSA	Public key of the eUICC used to verify an eUICC signature. This key is included inside the eUICC Certificate CERT.EUICC.ECDSA	ECC
SK.EUICC.ECDSA	Private key of the eUICC used to generate signatures. This key is loaded inside ECASD.	ECC
PK.DPauth.ECDSA	Public key of the SM-DP+ used to verify an SM-DP+ signature. This key is included inside the SM-DP+ Certificate CERT.DPauth.ECDSA.	ECC
SK.DPauth.ECDSA	Private key of the SM-DP+ used to generate signatures for authentication to the eUICC.	ECC
PK.DPpb.ECDSA	Public key of the SM-DP+ used to verify an SM-DP+ signature included in the BPP. This key is included inside the SM-DP+ Certificate CERT.DPpb.ECDSA.	ECC
SK.DPpb.ECDSA	Private key of the SM-DP+ used to generate signatures for Profile binding.	ECC

PK.DSauth.ECDSA	Public key of the SM-DS used to verify an SM-DS signature. This key is included inside the SM-DS Certificate CERT.DSauth.ECDSA.	ECC
SK.DSauth.ECDSA	Private key of the SM-DS used to provide signatures for authentication to the eUICC.	ECC
PK.EUM.ECDSA	Public key of the EUM used to verify EUICC Certificates. This key is included inside the EUM Certificate CERT.EUM.ECDSA.	ECC
SK.EUM.ECDSA	Private key of the EUM used to sign EUICC Certificates	ECC
PK.CI.ECDSA	Public key of the CI used to verify EUM, SM-DS and SM-DP+ Certificates.	ECC
SK.CI.ECDSA	Private key of the CI used to sign EUM, SM-DS and SM-DP+ Certificates.	ECC
otPK.EUICC.ECKA	One-time public key of the EUICC used for key agreement.	ECC
otSK.EUICC.ECKA	One-time private key of the EUICC used for key agreement.	ECC
otPK.DP.ECKA	One-time public key of the SM-DP+ used for key agreement.	ECC
otSK.DP.ECKA	One-time private key of the SM-DP+ used for key agreement.	ECC
SK.DP.TLS	Private key of the SM-DP+ used to generate signatures for authentication to the LPA.	ECC
PK.DP.TLS	Public key of the SM-DP+ used to verify an SM-DP+ signature.	ECC
SK.DS.TLS	Private key of the SM-DS used to generate signatures for authentication to the LPA.	ECC
PK.DS.TLS	Public key of the SM-DS used to verify an SM-DS signature.	ECC

Table 9: Cryptographic Keys

The curves used are defined in section 2.6.7.1.

4.5.2 Certificates

A Certificate Issuer issues certificates for Remote SIM Provisioning system entities and acts as a trusted root for the purpose of authentication of the entities of the system. The specification supports X.509 certificate format as defined in Section 4.5.2.1.

The following certificates SHALL be signed and issued by a GSMA CI:

- GSMA CI Certificate (CERT.CI.ECDSA)
- EUM Certificates (CERT.EUM.ECDSA)
- SM-DP+ Certificate (CERT.DPauth.ECDSA and CERT.DPpb.ECDSA)
- SM-DP+ TLS Certificate (CERT.DP.TLS)
- SM-DS Certificate (CERT.DSauth.ECDSA)
- SM-DS TLS Certificate (CERT.DS.TLS)

The following certificate SHALL be signed and issued by the EUM:

- eUICC Certificate (CERT.EUICC.ECDSA)

Even though each eUICC SHALL support at least two sets of elliptic curve parameters (section 2.6.7.1), which can be chosen from by an RSP server for its signatures and ECKA, an eUICC SHALL have at least one CERT.EUICC.ECDSA.

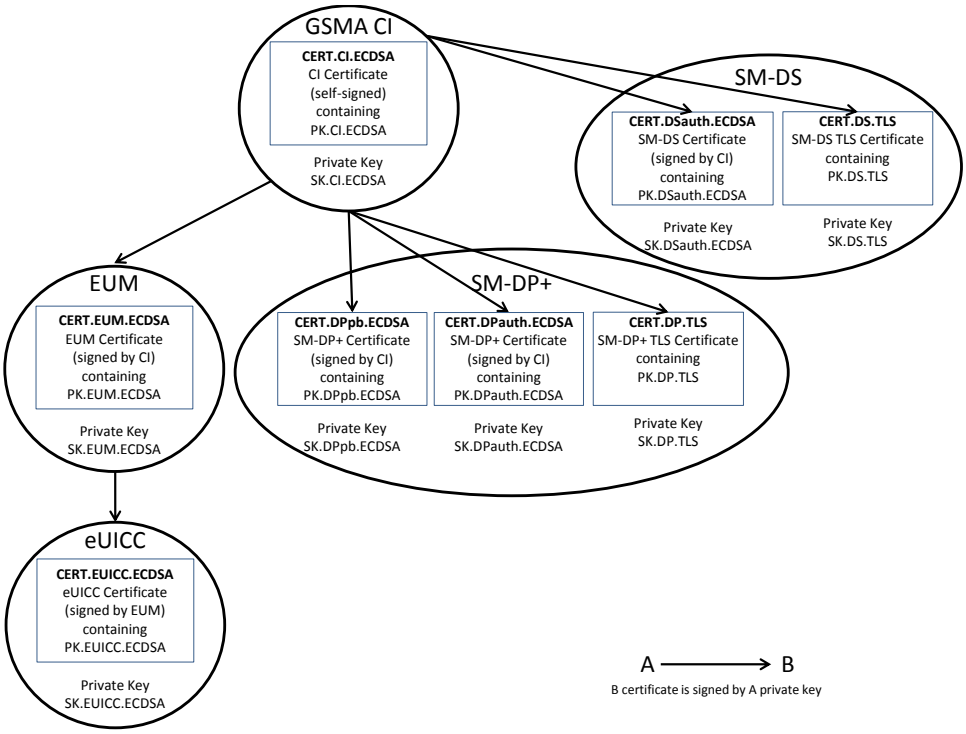


Figure 30: Certificate Chains

The Algorithm Identifiers of all certificates of a certificate chain SHALL point to the same curve.

The SM-DP+ has 2 ECDSA Certificates (CERT.DPauth.ECDSA and CERT.DPpb.ECDSA). The CERT.DPauth.ECDSA is used for authentication to the eUICC, and the CERT.DPpb.ECDSA is used for Profile binding.

The certificates CERT.CI.ECDSA, CERT.EUICC.ECDSA, CERT.EUM.ECDSA, CERT.DPauth.ECDSA, CERT.DPpb.ECDSA, CERT.DP.TLS, CERT.DSauth.ECDSA, and CERT.DS.TLS exchanged over ES9+, ES10b, ES8+ and ES11 are described in the next sections.

4.5.2.1 X.509 Certificate Profile

This section describes the X.509 certificate profile. Those Certificates SHALL follow RFC 5280 [17], with the specific coding given in this section.

In particular:

- 'Issuer' and 'Subject' fields SHALL be limited to standard attributes defined in ITU-T X.520 [24] and RFC 4519 [28].
- Certificates SHALL contain all extensions defined in their respective profile, except if stated otherwise.

Entities SHALL perform Certificate verification according to section 4.5.2.2.

NOTE: Certificates are described using table representation for easiness, but conform to the ASN.1 format given in RFC 5280 [17].

4.5.2.1.0 Certificates description

4.5.2.1.0.1 Certificate Issuer

Field	Value Description	
tbsCertificate	Data to be signed	
	Field	**Value Description**
	version	Version SHALL be 3 (value is 2) as extensions are used in this specification.
	serialNumber	Certificate serial number
	signature	Contains the algorithm identifier used by the issuer to compute the value of the field 'signatureValue' (section 4.5.2.1.1). NOTE: The algorithm identifier value SHALL be the same as the one of the field 'signatureAlgorithm'.
	issuer	This SHALL be identical to 'subject' field value.
	validity	Validity period of the certificate. Period when the CI is allowed to issue certificates.
	subject	Distinguished Name of the CI. Example of CI DN: cn = Symantec Class 3 Public Primary Certification Authority - G4 ou = Symantec Trust Network o = Symantec Corporation c = US
	subjectPublicKeyInfo	Contains the algorithm identifier, parameters and public key value. Algorithm identifier and parameters SHALL be set according to section 4.5.2.1.1. subjectPublicKeyInfo.subjectPublicKey contains the public key value and SHALL be coded as defined in RFC 5480 [27].
	extensions	Extension for Subject Key Identifier (RFC 5280 [17] section 4.2.1.2): extnID = id-ce- subjectKeyIdentifier critical = false extnValue = Public Key Identifier
		Extension for Key usage (RFC 5280 [17] section 4.2.1.3): extnID = id-ce-keyUsage critical = true extnValue = { keyCertSign (5), --[Mandatory] cRLSign(6) --[Optional] }

		Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4): extnID = id-ce-certificatePolicies critical = true extnValue = id-rspRole-ci (Annex H) To indicate the GSMA CI role.
		Extension for Basic Constraints (RFC 5280 [17] section 4.2.1.9): extnID = id-ce- basicConstraints critical = true extnValue = { cA = true }
		Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6): extnID = id-ce-subjectAltName critical = false extnValue = { registeredID (8) = CI OID }
		Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13). extnID = id-ce-cRLDistributionPoints critical = false extnValue = section 4.5.2.1.3
signatureAlgorithm	Section 4.5.2.1.1	
signatureValue	Signature computed accordingly to one of the possible algorithm listed in section 4.5.2.1.1	

Table 10: CERT.CI.ECDSA

NOTE: The CERT.CI.ECDSA is a self-signed certificate, there is no need to include the Extension for Authority Key Identifier.

4.5.2.1.0.2 eUICC

Field	Value Description	
tbsCertificate	Data to be signed	
	Field	**Value Description**
	version	Version SHALL be 3 (value is 2) as extensions are used in this specification.
	serialNumber	Serial number that SHALL be unique for each certificate issued with a given CERT.EUM.ECDSA.
	signature	Contains the algorithm identifier used by the EUM to compute the value of the field 'signatureValue'. Apply rules defined in the table related to CERT.CI.ECDSA.

	issuer	Distinguished Name of the EUM that has signed the EUICC Certificate. It SHALL match the ‘subject’ field of the EUM Certificate CERT.EUM.ECDSA.
	validity	Validity period of the certificate. There is no life time defined for this certificate. eUICC Certificates never expire. Expiration Date to be set to 99991231235959Z as stated in RFC 5280 [17]
	subject	Distinguished Name of the EUICC. It SHALL include, at least, 'organization' and 'serialNumber' attributes. Others attributes MAY be included for information. The 'organization' attribute SHOULD have the same value as the 'organization' attribute of the EUM. 'serialNumber' SHALL be the EID as an hexadecimal PrintableString. Example of an eUICC DN: o = ACME serialNumber = 8904903212345123451234567890123 5
	subjectPublicKeyInfo	Contains the algorithm identifier, parameters and public key value. Apply rules defined in the table related to CERT.CI.ECDSA.
	extensions	Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1): extnID = id-ce- authorityKeyIdentifier critical = false extnValue = keyIdentifier [0] To identify the PK.EUM.ECDSA that has to be used to verify this certificate.
		Extension for Subject Key Identifier (RFC 5280 [17] section 4.2.1.2): extnID = id-ce- subjectKeyIdentifier critical = false extnValue = keyIdentifier [0] Contains the identifier of the PK.EUICC.ECDSA bound in this certificate.
		Extension for Key usage (RFC 5280 [17] section 4.2.1.3): extnID = id-ce-keyUsage critical = true extnValue = digitalSignature (0)
		Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4): extnID = id-ce-certificatePolicies critical = true extnValue = id-rspRole-euicc (Annex H) To indicate that this is an eUICC Certificate.

signatureAlgorithm	Section 4.5.2.1.1
signatureValue	Signature computed accordingly to one of the possible algorithm listed in section 4.5.2.1.1

Table 11: CERT.EUICC.ECDSA

4.5.2.1.0.3 EUM

Field	Value Description	
tbsCertificate	Data to be signed	
	Field	**Value Description**
	version	Version SHALL be 3 (value is 2) as extensions are used in this specification.
	serialNumber	Serial number that SHALL be unique for each certificate issued with a given CERT.CI.ECDSA.
	signature	Contains the algorithm identifier used by the issuer to compute the value of the field 'signatureValue'. Apply rules defined in the table related to CERT.CI.ECDSA.
	issuer	Distinguished Name of the GSMA CI that has signed the EUM Certificate. Example of CI DN: cn = Symantec Class 3 Public Primary Certification Authority - G4 ou = Symantec Trust Network o = Symantec Corporation c = US
	validity	Validity period of the certificate. Period when the EUM is allowed to issue eUICC Certificates.
	subject	Distinguished Name of the EUM. It SHALL include at least 'organization' and 'commonName' attributes. Example of EUM DN: c = US l = New York o = ACME cn = ACME Public CA e = admin.pki@acme.com
	subjectPublicKeyInfo	Contains the algorithm identifier, parameters and public key value. Apply rules defined in the table related to CERT.CI.ECDSA.
	extensions	Extension for Authority Key Identifier (RFC 5280 [17]): section 4.2.1.1): extnID = id-ce-authorityKeyIdentifier critical = false extnValue = keyIdentifier [0] To identify the PK.CI.ECDSA that has to be used to verify this certificate.

		Extension for Subject Key Identifier (RFC 5280 [17] section 4.2.1.2): extnID = id-ce- subjectKeyIdentifier critical = false extnValue = keyIdentifier [0] Contains the identifier of the PK.EUM.ECDSA bound in this certificate.
		Extension for Key usage (RFC 5280 [17] section 4.2.1.3): extnID = id-ce-keyUsage critical = true extnValue = extnValue = { keyCertSign (5), --[Mandatory] cRLSign(6) --[Optional] }
		Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4): extnID = id-ce-certificatePolicies critical = true extnValue = id-rspRole-eum (Annex H) To indicate that this is an EUM Certificate.
		Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6): extnID = id-ce-subjectAltName critical = false extnValue = { registeredID (8) = EUM OID }
		Extension for Basic Constraints (RFC 5280 [17] section 4.2.1.9): extnID = id-ce- basicConstraints critical = true extnValue = { cA = true pathLenConstraint = 0 } To indicate that this certificate is a sub-ca limited to issue only "leaf" certificate for the eUICC.
		Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13). extnID = id-ce-cRLDistributionPoints critical = false extnValue = section 4.5.2.1.3

		Extension for Name Constraints (see RFC 5280 [17] section 4.2.1.10)
		extnID = id-ce-nameConstraints
		critical = true
		extnValue NameConstraints ::= {
		permittedSubtrees ::= {
		{
		base directoryName : rdnSequence : {
		{
		{
		type { <id-at-organizationName oid> }
		value : <organization name>
		}
		},
		{
		{
		type { < id-at-serialNumber oid> }
		value: <iin>
		}
		}
		},
		minimum 0
		}
		}
		}
		This restriction contains the organization name(s) and IIN(s) that the EUM owning this Certificate is allowed to set in the eUICC Certificates. This restriction applies on the subject name (containing 'organization' and 'serialNumber' attributes).
		The <iin> value is composed of the 1st to 8th digits of the EID.
		The extension MAY contain several possible 'organization' / 'IIN' value pairs.
		Field 'minimum' has no meaning in this specification.

signatureAlgorithm	Section 4.5.2.1.1	
signatureValue	Signature computed accordingly to one of the possible algorithm listed in section 4.5.2.1.1	

Table 12: CERT.EUM.ECDSA

4.5.2.1.0.4 SM-DP+ ECDSA

Field	Value Description	
tbsCertificate	Data to be signed	
	Field	**Value Description**

	version	Version SHALL be 3 (value is 2) as extensions are used in this specification.
	serialNumber	Serial number that SHALL be unique for each certificate issued with a given CERT.CI.ECDSA.
	signature	Contains the algorithm identifier used by the issuer to compute the value of the field 'signatureValue'. Apply rules defined in the table related to CERT.CI.ECDSA.
	issuer	Distinguished Name of the GSMA CI that has signed the SM-DP+ Certificate.
	validity	Validity period of the Certificate.
	subject	Distinguished Name of the SM-DP+. It SHALL include at least 'organization' and 'commonName' attributes. Example of an SM-DP+ DN: c = US l =New York o = ACME cn = ACME DP e = dp@acme.com
	subjectPublicKeyInfo	Contains the algorithm identifier, parameters and public key value. Apply rules defined in the table related to CERT.CI.ECDSA.
	extensions	Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1): extnID = id-ce- authorityKeyIdentifier critical = false extnValue = keyIdentifier [0] To identify the PK.CI.ECDSA that has to be used to verify this Certificate.
		Extension for Subject Key Identifier (RFC 5280 [17] section 4.2.1.2): extnID = id-ce-subjectKeyIdentifier critical = false extnValue = keyIdentifier [0] Contains the identifier of the public key bound in this Certificate.
		Extension for Key usage (RFC 5280 [17] section 4.2.1.3): extnID = id-ce-keyUsage critical = true extnValue = digitalSignature (0)

		Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4): extnID = id-ce-certificatePolicies critical = true extnValue = id-rspRole-dp-pb (Annex H)for CERT.DPpb.ECDSA, or extnValue = id-rspRole-dp-auth (Annex H) for CERT.DPauth.ECDSA, 'extnValue' SHALL be one of the above OIDs to indicate the role of this SM-DP+ Certificate (authentication to the eUICC or Profile binding).
		Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6): extnID = id-ce-subjectAltName critical = false extnValue = { registeredID (8) = SM-DP+ OID}
		Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13). extnID = id-ce-cRLDistributionPoints critical = false extnValue = section 4.5.2.1.3
signatureAlgorithm	Section 4.5.2.1.1	
signatureValue	Signature computed according to one of the possible algorithm listed in section 4.5.2.1.1	

Table 13: CERT.DPauth.ECDSA / CERT.DPpb.ECDSA

All the field values of the CERT.DPauth.ECDSA and CERT.DPpb.ECDSA SHALL be identical except for the following fields:

- serialNumber
- extension (extnValue=id-ce-certificatePolicies)
- subjectPublicKeyInfo
- signatureValue
- extension (extnID = id-ce-subjectKeyIdentifier)

4.5.2.1.0.5 SM-DP+ TLS

Field	Value Description	
tbsCertificate	Data to be signed	
	Field	**Value Description**
	version	Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.
	serialNumber	Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.

	signature	Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.
	issuer	Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.
	validity	Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.
	subject	Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA. Both CERT.DPauth.ECDSA, CERT.DPpb.ECDSA and CERT.DP.TLS SHALL have identical attributes except 'cn' that SHALL contain the SM-DP+ hostname (FQDN).
	subjectPublicKeyInfo	Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.
	extensions	Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1): Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.
		Extension for Subject Key Identifier (RFC 5280 [17] section 4.2.1.2): Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.
		Extension for Key usage (RFC 5280 [17] section 4.2.1.3): Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.
		Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4): extnID = id-ce-certificatePolicies critical = false extnValue = id-rspRole-dp-tls (Annex H) To indicate that this is an SM-DP+ Certificate for TLS.

		Extension for Extended Key usage (RFC 5280 [17] 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.

		Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6): extnID = id-ce-subjectAltName critical = false extnValue = { dNSName (2) = SM-DP+ hostname (FQDN) registeredID (8) = SM-DP+ OID}
		Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13). Refer to Table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA
signatureAlgorithm	Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.	
signatureValue	Refer to table defining CERT.DPauth.ECDSA / CERT.DPpb.ECDSA.	

Table 14: CERT.DP.TLS

4.5.2.1.0.6 SM-DS ECDSA

Field	Value Description	
tbsCertificate	Data to be signed	
	Field	**Value Description**
	version	Version SHALL be 3 (value is 2) as extensions are used in this specification.
	serialNumber	Serial number that SHALL be unique for each certificate issued with a given CERT.CI.ECDSA.
	signature	Contains the algorithm identifier used by the issuer to compute the value of the field 'signatureValue'. Apply rules defined in the table related to CERT.CI.ECDSA.
	issuer	Distinguished Name of the GSMA CI that has signed the SM-DS Certificate.
	validity	Validity period of the Certificate.
	subject	Distinguished Name of the SM-DS. It SHALL include at least 'organization' and 'commonName' attributes. Example of an SM-DS DN: c = US l = New York o = ACME cn = ACME SM-DS e = ds@acme.com
	subjectPublicKeyInfo	Contains the algorithm identifier, parameters and public key value. Apply rules defined in the table related to CERT.CI.ECDSA.

	extensions	Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1): extnID = id-ce-authorityKeyIdentifier critical = false extnValue = keyIdentifier [0] To identify the PK.CI.ECDSA that has to be used to verify this Certificate.
		Extension for Subject Key Identifier (RFC 5280 [17] section 4.2.1.2): extnID = id-ce-subjectKeyIdentifier critical = false extnValue = keyIdentifier [0] Contains the identifier of the PK.DSauth.ECDSA bound in this Certificate.
		Extension for Key usage (RFC 5280 [17] section 4.2.1.3): extnID = id-ce-keyUsage critical = true extnValue = digitalSignature (0)
		Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4): extnID = id-ce-certificatePolicies critical = true extnValue = id-rspRole-ds-auth (Annex H) To indicate that this is an SM-DS ECDSA Certificate for authentication to the eUICC.
		Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6): extnID = id-ce-subjectAltName critical = false extnValue = { registeredID (8) = SM-DS OID}
		Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13). extnID = id-ce-cRLDistributionPoints critical = false extnValue = section 4.5.2.1.3
signatureAlgorithm	Section 4.5.2.1.1	
signatureValue	Signature computed according to one of the possible algorithm listed in section 4.5.2.1.1	

Table 15: CERT.DSauth.ECDSA

4.5.2.1.0.7 SM-DS TLS

Field	Value Description	
tbsCertificate	Data to be signed	
	Field	**Value Description**
	version	Refer to table defining CERT.DSauth.ECDSA.
	serialNumber	Refer to table defining CERT.DSauth.ECDSA.
	signature	Refer to table defining CERT.DSauth.ECDSA.
	issuer	Refer to table defining CERT.DSauth.ECDSA.
	validity	Refer to table defining CERT.DSauth.ECDSA.
	subject	Refer to table defining CERT.DSauth.ECDSA. Both CERT.DSauth.ECDSA and CERT.DS.TLS SHALL have identical attributes except 'cn' that SHALL contain the SM-DS hostname (FQDN).
	subjectPublicKeyInfo	Refer to table defining CERT.DSauth.ECDSA.
	extensions	Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1): Refer to table defining CERT.DSauth.ECDSA.
		Extension for Subject Key Identifier (RFC 5280 [17] section 4.2.1.2): Refer to table defining CERT.DSauth.ECDSA.
		Extension for Key usage (RFC 5280 [17] section 4.2.1.3): Refer to table defining CERT.DSauth.ECDSA.
		Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4): extnID = id-ce-certificatePolicies critical = false extnValue = id-rspRole-ds-tls (Annex H) To indicate that this an SM-DS TLS Certificate.

		Extension for Extended Key usage (RFC 5280 [17] 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.

		Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6): extnID = id-ce-subjectAltName critical = false extnValue = { dNSName (2) = SM-DS hostname (FQDN) registeredID (8) = SM-DS OID}
		Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13). Refer to Table defining CERT.DSauth.ECDSA
signatureAlgorithm	Refer to table defining CERT.DSauth.ECDSA.	
signatureValue	Refer to table defining CERT.DSauth.ECDSA.	

Table 16: CERT.DS.TLS

4.5.2.1.1 Algorithm Identifiers and Parameters

This section provides the values to be set in 'AlgorithmIdentifier.algorithm' and 'AlgorithmIdentifier.parameters' fields of the certificate 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.

'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 [18]
- 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]
- For FRP256V1: "iso(1) member-body(2) fr(250) type-org(1) 223 101 256 1" as defined in ANSSI ECC [20]

For sections 'signature' and 'signatureAlgorithm' the following settings SHALL apply:

- 'AlgorithmIdentifier.algorithm' field SHALL be set to: "iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) ecdsa-with-SHA256(2)" as defined in RFC 5758 [25] and RFC 5759 [26]
- 'AlgorithmIdentifier.parameters' field SHALL be omitted as defined in RFC 5758 [25] section 3.2.

4.5.2.1.3 Extension CRL Distribution Points

This specification defines two DistributionPoints. The first DistributionPoint (called DistributionPoint-A) is for retrieving the CRL-A. The second DistributionPoint (called DistributionPoint-B) is for retrieving the CRL-B (section 4.6). The CRL-A, respectively the CRL-B, SHALL include the extension 'Issuing Distribution Point' set with the DistributionPoint-A value, respectively DistributionPoint-B. This extension SHALL contain

one DistributionPoint for each CRL the certificate belongs to (i.e. CERT.EUM.ECDSA SHALL only contain the DistributionPoint-B, CERT.DPAuth.ECDSA SHALL contain the DistributionPoint-A and the DistributionPoint-B).

DistributionPoint SHALL only have the 'distributionPoint' field set. Optional 'reasons' field is not used; each revoked certificate SHALL have its own reason set. And 'cRLIssuer' field is not used as the CRL SHALL be issued by the Certificate Issuer.

The 'distributionPoint' field MAY contain several general names, each describing a different mechanism to obtain the same CRL (the field 'nameRelativeToCRLIssuer' is not used in this specification). But 'distributionPoint' SHALL contain at least a general name of type URI with an HTTP scheme, indicating that the CRL can be retrieved as an HTTP resource.

In the case the CRL-A would have to be split into several parts (section 4.6.1), the DistributionPoint-A SHALL allow to retrieve all the CRL-A parts.

4.5.2.2 Certificate Verification

Any of the certificates described in section 4.5.2 SHALL be verified according to the description given in this section.

If any of these verifications fail, the certificate SHALL be considered as invalid and the operation for which it was used, SHALL be rejected.

Every certificate SHALL:

- Have a valid signature.
- Be signed by a GSMA CI, or a trusted chain up to a GSMA CI. Certificate path validation SHALL follow the process defined in RFC 5280 [17], using the Subject Key Identifier and Authority Key Identifier fields. As a consequence Certificates SHALL have the 'Subject Key Identifier' and 'Authority Key Identifier' extensions set, except the GSMA CI certificate that SHALL only have the 'Subject Key Identifier' extensions set.
- Not have been revoked, and no certificate in the trust chain has been revoked (It should be noted that the eUICC might not support certificate revocation or might not have been provided with the latest CRL to perform this verification).
- Not have expired (It should be noted that the eUICC and the Device might not have reliable access to the current time to perform this verification).
- Have all the 'critical' extension defined for its profile.

In addition to those verifications, and specifically to those certificates:

CERT.EUICC.ECDSA:

- The extension 'Key usage' SHALL be set with the value 'digitalSignature'.
- The extension 'Certificate Policies' SHALL be set with the OID 'id-rspRole-euicc' to indicate an eUICC Certificate.
- Verify that the 'subject' field complies to the 'Name constraints' extension contained in the CERT.EUM.ECDSA:

- 'organization' attribute value SHALL be one of the possible organization names contained in the 'Name constraints'
- 'serialNumber' attribute value (containing EID) SHALL start with an IIN (1st to 8th digits) corresponding to the respective organization name contained in the 'Name constraints'

This verification SHALL be done as defined in section 7.1 of RFC 5280 [17]

CERT.EUM.ECDSA:

- The extension 'Key usage' SHALL be set with the value 'keyCertSign'.
- The extension 'Certificate Policies' SHALL be set with the OID 'id-rspRole-eum' to indicate an EUM Certificate.
- The extension 'Basic Constraints' SHALL be set to cA=true. The path length restriction SHALL be set to 0.

CERT.DPauth.ECDSA / CERT.DPpb.ECDSA:

- The extension 'Key usage' SHALL be set with the value 'digitalSignature'.
- The extension 'Certificate Policies' SHALL be set with one of the following OID:
    - 'id-rspRole-dp-auth' to indicate an SM-DP+ Certificate for authentication with the eUICC.
    - 'id-rspRole-dp-pb' to indicate an SM-DP+ Certificate for profile binding.
- The extension 'subjectAltName' SHALL be set with the SM-DP+ OID.

CERT.DP.TLS:

- The extension 'Key usage' SHALL be set to 'digitalSignature (0)'.
- The extension 'extended key usage' SHALL be set to 'id-kp-serverAuth' and 'id-kp-clientAuth', and no other extended key usage.
- The extension 'Certificate Policies', if present, SHALL be set with the OID 'id-rspRole-dp-tls' to indicate an SM-DP+ Certificate for TLS.

CERT.DSauth.ECDSA:

- The extension 'Key usage' SHALL be set with the value 'digitalSignature'.
- The extension 'Certificate Policies' SHALL be set with the OID 'id-rspRole-ds-auth' to indicate an SM-DS Certificate for authentication with the eUICC.

CERT.DS.TLS:

- The extension 'key usage' SHALL be set to 'digitalSignature (0)'.
- The extension 'extended key usage' SHALL be set to 'id-kp-serverAuth' and 'id-kp-clientAuth', and no other extended key usage.
- The extension 'Certificate Policies', if present, SHALL be set with the OID 'id-rspRole-ds-tls' to indicate an SM-DS Certificate for TLS.

4.6 Certificate Revocation List

A Certificate Revocation List (CRL) is issued by a GSMA CI and contains one or more revoked certificates from among all the unexpired certificates it has issued. Note that the RSP architecture supports multiple GSMA CIs. Each GSMA CI MAY individually issue a CRL.

The CRL SHALL follow RFC 5280 [17] with the specific coding and rules given in this section.

The GSMA CI SHALL maintain two different CRLs:

- CRL referencing only CERT.CI.ECDSA, CERT.DPauth.ECDSA, CERT.DPpb.ECDSA, CERT.DP.TLS, CERT.DSauth.ECDSA and CERT.DS.TLS revoked certificates (in order to minimise its size, as this list has as its destination the LPA and eUICC). This CRL is noted CRL-A in this document.
- CRL referencing CERT.CI.ECDSA, CERT.DPauth.ECDSA, CERT.DPpb.ECDSA, CERT.DP.TLS, CERT.DSauth.ECDSA, CERT.DS.TLS and CERT.EUM.ECDSA revoked certificates. This CRL is noted CRL-B in this document.

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

Only the CRLs issued by a GSMA CI SHALL be considered by the SM-DP+, SM-DS, LPA and eUICC in the context of this specification.

NOTE: CRL is described using table representation for easiness, but conforms to the ASN.1 format given in RFC 5280 [17].

Field	Value Description	
tbsCertList	Data to be signed	
	Field	**Value Description**
	version	Version SHALL be 2.
	signature	Contains the algorithm identifier and parameters used by the GSMA CI to compute the value of the field 'signatureValue' (section 4.5.2.1.1). Note: The algorithm identifier value and parameters values SHALL be the same as the one of the field 'signatureAlgorithm'.
	issuer	Distinguished Name of the GSMA CI that has signed the CRL. This DN SHALL be the same as the one used for issuing certificates.
	thisUpdate	Time stamp of the CRL.
	nextUpdate	Indicates the date by which the next CRL will be issued. The next CRL could be issued before the indicated date (if any revocation occurs before this date), but it will not be issued any later than the indicated date.

	revokedCertificates	Sequence of "revocation entry" as described in table 18.
	crlExtensions	Extension for Authority Key Identifier (RFC 5280 [17] section 5.2.1): extnID = id-ce- authorityKeyIdentifier critical = true extnValue = keyIdentifier [0] To identify the PK.CI.ECDSA that has to be used to verify this CRL.
		Extension for CRL Number (RFC 5280 [17] section 5.2.3): extnID = id-ce-cRLNumber critical = false extnValue = CRL number, starting from 0.

		Extension for Issuing Distribution Point (RFC 5280 [17] section 5.2.5):
		extnID = id-ce- issuingDistributionPoint
		critical = true
		extnValue = {
		distributionPoint [0] -- section 4.5.2.1.3
		}
		Others fields (onlyContainsUserCerts, onlyContainsCACerts, indirectCRL, onlyContainsAttributeCerts) MAY be absent or set to false.

		Extension for Total Partial CRL Number (section 4.6.2) extnID = id-rsp-totalPartialCrlNumber critical = false extnValue = TotalPartialCRLNumber In the case the CRL-A would have to be cut into partial-CRL (section 4.6.1), this extension SHALL be set. This gives the total number of partial-CRL resulting of the splitting process.
		Extension for Partial CRL Number (section 4.6.2) extnID = id-rsp-partialCrlNumber critical = false extnValue = PartialCRLNumber This extension SHALL be set jointly with 'id-rsp-totalPartialCrlNumber'. This gives the sequence number of the partial-CRL resulting of the splitting process, starting from 1.
signatureAlgorithm	Section 4.5.2.1.1	
signatureValue	Signature computed accordingly to one of the possible algorithm listed in signatureAlgorithm field	

Table 17: CRL Description

Delta CRL MAY be used by a GSMA CI. In that case the CRL SHALL include the extension 'Delta CRL Indicator' and 'Freshest CRL' as defined in RFC 5280 [17].

Field	Value Description
userCertificate	Serial number of the revoked certificate
revocationDate	Date of revocation
crlEntryExtensions	Extension for Reason code(1) (RFC 5280 [17] section 5.3.1): extnID = id-ce-cRLReasons critical = false extnValue = a reason code
	Extension for Certificate Expiration Date (this extension is defined in section 4.6.2) extnID = id-rsp-expDate critical = true extnValue = Expiration date of the revoked certificate. The date SHALL follow coding rules defined in RFC 5280 [17] for any date. Also see section 4.6.2 for more details on usage by the eUICC.
NOTE 1: in order to limit the size of the CRL, this extension SHALL not be set in the CRL-A. But this extension SHALL be set in the CRL-B.	

Table 18: Revocation entry

4.6.1 CRL publication rules

Each GSMA CI SHALL issue a new CRL 1) periodically (even if no new revocation has occurred during the period), and 2) as soon as at least one additional certificate is revoked.

NOTE: The publication periodicity is not defined in this document. This SHALL be defined in GSMA eUICC PKI Certificate Policy [45].

The SM-DP+, SM-DS or any relying party SHALL retrieve the new CRL at the date indicated by the 'nextUpdate' of the previous published CRL, or as soon as notified that a new CRL has been issued. The means by which the SM-DP+, the SM-DS or any relying party can subscribe to CRL update notifications, and how the GSMA CI notifies those CRL updates are out of scope of this release of this specification.

The CRL-A and CRL-B base lists SHALL be complete (i.e. contain all previously revoked Certificates, plus the newly revoked Certificates). Delta CRLs as defined in RFC 5280 [17] MAY be used by the GSMA CI. This Delta CRLs mechanism is especially useful for CRL-A in order to limit the size of the CRL to be loaded onto the eUICC.

In addition, the size of the DER encoded CRL-A SHALL not exceed 1024 bytes (similar limitation as for BPP loading). In the case the list is greater than this limit, the GSMA CI SHALL split it in pieces none of which are greater than this limit. Each resulting partial-CRL SHALL have the two specific extensions 'totalPartialCrlNumber' and 'partialCrlNumber' and be individually signed as any CRL.

The GSMA CI SHALL manage Extension for CRL Number accordingly to RFC 5280 [17]. This number has to be incremented by one at each new CRL publication.

4.6.2 Specific CRL Extensions

This section provides the definition of specific CRL extensions used in this specification.

```
-- Definition of the extension for Certificate Expiration Date
id-rsp-expDate OBJECT IDENTIFIER ::= {id-rspExt 1}
ExpirationDate ::= Time

-- Definition of the extension id for total partial-CRL number
id-rsp-totalPartialCrlNumber OBJECT IDENTIFIER ::= {id-rspExt 2}
TotalPartialCrlNumber ::= INTEGER


-- Definition of the extension id for the partial-CRL number
id-rsp-partialCrlNumber OBJECT IDENTIFIER ::= {id-rspExt 3}
PartialCrlNumber ::= INTEGER
```

4.6.3 eUICC Considerations

The eUICC faces two general issues regarding time management and CRL.

- The eUICC does not have any time reference internally and can only rely on time provided from an off-card entity (with the question on reliability of this information). This limitation prevents the eUCC to verify that a certificate has expired.
- The eUICC is in general very constrained with regard to its non-volatile memory size. But the list of revoked certificate is expected to be growing continually. This may be a challenge to store such information on the eUICC (it is even worse considering that the eUICC will have to manage a CRL per GSMA CI).

This section provides rules for the eUICC implementation to address these two concerns.

The CRL-A 'tbsCertList.thisUpdate' field contains a time reference that can be considered as reliable because it is signed by the GSMA CI. Once the CRL-A has been verified by the eUICC, this time stamp SHALL be used to verify if a certificate has expired. Any certificate with an expiration date that is before the time reference of the last received valid CRL-A SHALL be considered has expired by the eUICC.

Each CRL entry contains an extension that gives the expiry date of the revoked Certificate. When the eUICC receives a new valid CRL-A, the eUICC MAY remove from its internal storage any revoked certificate reference having an expiration date which is before the time reference contained in the received CRL-A. This allows the eUICC to perform clean-up of its internal storage over the time.

4.7 Confirmation Code

A Profile download order MAY be protected by a specific Confirmation Code. The Confirmation Code is provided by the Operator to the SM-DP+ and the End User during the Profile download initiation procedure (section 3.1.1). The means by which the Confirmation Code is provided to the End User is out of scope of this specification.

During the Profile download and installation procedure (section 3.1.3), if the Profile download order is protected by a Confirmation Code, the SM-DP+ SHALL verify that the Confirmation Code provided by the End User matches the Confirmation Code provided by the Operator.

In addition, the SM-DP+ SHALL protect against excessive incorrect entries of the Confirmation Code. The maximum number of Confirmation Code retries allowed to an End

User is defined by the Operator and communicated to the SM-DP+ by means out of scope of this specification.

Once the maximum number of Confirmation Code retries is exceeded for a Profile download order, the Profile download order SHALL be terminated and the SM-DP+ SHALL communicate the final status to the Operator. The Operator is free to request a new Profile download order corresponding to the same Profile, with the same or a different Confirmation Code.

5 Functions

This section specifies the Functions associated with the Remote SIM Provisioning and Management of the eUICC for consumer Devices.

5.1 Overview of Functions per Interface

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 LPAe (LPDe) for the delivery of the Profile Package.
- ES10a: Used by the LPAd to get the configured addresses from the eUICC for Root SM-DS, and optionally the default SM-DP+.
- ES10b: Used by the LPAd to transfer a Profile Package to the eUICC.
- ES10c: Used by the LPAd for local End User management of Profiles installed on the eUICC (e.g. Enable, Disable, Delete).
- ES11: Interface between the LPAe and an SM-DS (Alternative SM-DS or Root SM-DS) for the Event retrieval.

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.
- ES15: Interface between an Alternative SM-DS and the Root SM-DS for the Event management.

Device to Server Interfaces

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

ESop (interface between the End User and the Operator), ESeum (Interface between the EUM and the eUICC) and ESeu (Interface between the End User and the LUI) are out of scope of this document.

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+	DownloadOrder	SM-DP+
	ConfirmOrder	SM-DP+
	CancelOrder	SM-DP+
	ReleaseProfile	SM-DP+
ES6	UpdateMetadata	ISD-P
ES8+	InitialiseSecureChannel ReplaceSessionKeys ConfigureISDP StoreMetadata LoadProfileElements	ISD-R/ISD-P
ES9+	InitiateAuthentication GetBoundProfilePackage AuthenticateClient CancelSession	SM-DP+
ES10a	GetEuiccConfiguredAddresses SetDefaultDpAddress	ISD-R (LPA Services)
ES10b	GetEUICCChallenge GetEUICCInfo PrepareDownload LoadBoundProfilePackage ListNotification RetrieveNotificationsList RemoveNotificationFromList LoadCRL AuthenticateServer CancelSession GetRAT	ISD-R (LPA Services)

ES10c	GetProfilesInfo EnableProfile DisableProfile DeleteProfile eUICCMemoryReset GetEID SetNickname	ISD-R (LPA Services)

Table 19: Request-Response Functions

Notification Handler Functions:

Interface	Notification handler functions	Function handler/recipient
ES2+	HandleDownloadProgressInfo	Operator
ES9+	HandleNotification	SM-DP+

Table 20: Notification Handler Functions

5.2 Server to Server Function Commonalities

Each functions represents an entry points 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
Hexadecimal String	String of even length composed of characters between '0' to '9' and 'A' to 'F' or 'a' to 'f'.	
AID	The AID (Application Identifier) of an Executable Load File, an Executable Module, a security domain, or an Application.	Hexadecimal string representation of 5 to 16 bytes.

DATETIME	Any date and time used within any interface of this specification.	String format as specified by W3C: YYYY-MM-DDThh:mm:ssTZD Where: YYYY = four-digit year MM = two-digit month (01=Jan, etc.) DD = two-digit day of month (01-31) hh = two digits of hour (00 -23) mm = two digits of minute (00 - 59) ss = two digits of second (00 - 59) TZD = time zone designator (Z, +hh:mm or -hh:mm) Ex: 2001-12-17T09:30:47Z
EID	The EID type is for representing an eUICC-ID. An eUICC-ID is primarily used in the "Embedded UICC Remote Provisioning and Management System" to identify an eUICC. See SGP.02 [2] section 5.2.4.6 for EID description.	String of 32 decimal characters
FQDN	The FQDN type is for representing a Fully Qualified Domain Name (e.g. smdp.gsma.com).	String, as a list of domain labels concatenated using the full stop (dot, period) character as separator between labels. Labels are restricted to the Alphanumeric mode character set defined in table 5 of ISO/IEC 18004 [15]
ICCID	The ICCID type is for representing an ICCID (Integrated Circuit Card Identifier). The ICCID is primarily used to identify a Profile. ICCID is defined according to ITU-T recommendation E.118 [21]. However, the ICCID SHALL either consist of 19 or 20 digits.	String representation of 19 or 20 digits, where the 20th digit MAY optionally be the padding character F. Ex: 89470100000123456784F A 19 digit ICCID with and without padding SHALL identify the same Profile.
OID	An Object Identifier.	String representation of an OID, i.e. of integers separated with dots (e.g.: '1.2', '3.4.5')
VERSION	The Version type is for indicating a version of any entity used within this specification. A version is defined by its major, minor and revision number.	String representation of three integers separated with dots (e.g.: ‘1.15.3’).

Table 21: Common data types

5.2.2 Request-Response Function

As defined in SGP.02 [2] subject to the following constraint:

- The Validity Period defined in SGP.02 [2] is not used in this specification.

5.2.3 Notification Handler Function

As defined in SGP.02 [2].

5.2.4 Functions Input Header

As defined in SGP.02 [2] subject to the following constraint:

The field `Validity Period` SHALL not be present in `Functions Input Headers`.

5.2.5 Functions Output Header

As defined in SGP.02 [2] subject to the following constraint:

The fields `Processing Start`, `Processing End` and `Acceptable Validity Period` SHALL not be present in `Functions Output Headers`.

5.2.6 Status Code

This specification relies on subject codes and reason codes as defined in SGP.02 [2]. In addition this specification defines the 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 [2])
    1.1. Function Requester (as defined in SGP.02 [2])
    1.2. Function Provider (as defined in SGP.02 [2])
    1.3. Protocol (as defined in SGP.02 [2])
        1.3.1. Protocol Format (as defined in SGP.02 [2])
        1.3.2. Protocol Version (as defined in SGP.02 [2])
    1.6. Function (as defined in SGP.02 [2])
8. eUICC Remote Provisioning (as defined in SGP.02 [2])
    8.1. eUICC (as defined in SGP.02 [2])
        8.1.1. EID (as defined in SGP.02 [2])
        8.1.2. EUM Certificate
        8.1.3. eUICC Certificate
    8.2. Profile (as defined in SGP.02 [2])
        8.2.1. Profile ICCID (as defined in SGP.02 [2])
        8.2.5. Profile Type (as defined in SGP.02 [2])
        8.2.6. Matching ID

8.2.7. Confirmation Code

8.2.8. PPR

8.2.9. Profile Metadata

8.2.10. Bound Profile Package

8.8. SM-DP+

8.8.1. SM-DP+ Address

8.8.2. Security configuration

8.8.3. Specification Version Number (SVN)

8.8.4. SM-DP+ Certificate

8.8.5 Download order

8.9. SM-DS

8.9.1. SM-DS Address

8.9.2. Security configuration

8.9.3. Specification Version Number (SVN)

8.9.4 SM-DS Certificate

8.9.5. Event Record

8.10. RSP Operation

8.10.1. TransactionId

8.11. GSMA CI

8.11.1. Public Key (PK)

5.2.6.2 Reason Code

Hereunder are listed the reason codes used in this specification:

1. Access Error (as defined in SGP.02 [2])

1.1. Unknown (Identification or Authentication) (as defined in SGP.02 [2])

1.2. Not Allowed (Authorisation) (as defined in SGP.02 [2])

2. Format Error (as defined in SGP.02 [2])

2.1. Invalid (as defined in SGP.02 [2])

2.2. Mandatory Element Missing (as defined in SGP.02 [2])

2.3. Conditional Element Missing (as defined in SGP.02 [2])

3. Conditions of Use Not Satisfied (as defined in SGP.02 [2])

    3.1. Unsupported (as defined in SGP.02 [2])

    3.3. Already in Use (Uniqueness) (as defined in SGP.02 [2])

    3.7. Unavailable (as defined in SGP.02 [2])

    3.8. Refused (as defined in SGP.02 [2])

    3.9. Unknown (as defined in SGP.02 [2])

    3.10. Invalid Association

    3.11. Value has Changed

4. Processing Error (as defined in SGP.02 [2])

    4.2. Execution Error (as defined in SGP.02 [2])

    4.3. Stopped on Warning (as defined in SGP.02 [2])

    4.8. Insufficient Memory (as defined in SGP.02 [2])

    4.10 Time to Live Expired

5. Transport Error (as defined in SGP.02 [2])

    5.1. Inaccessible (as defined in SGP.02 [2])

6. Security Error (as defined in SGP.02 [2])

    6.1. Verification Failed (as defined in SGP.02 [2])

    6.3. Expired

    6.4. Maximum number of retries exceeded

5.2.6.3 Common Function Status Code

As defined in SGP.02 [2].

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

The ES2+ 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+.

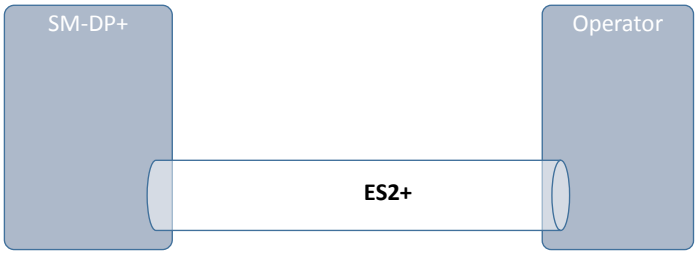


Figure 31: ES2+

The Operator communicates with the SM-DP+ through a secure connection. The level of security requested on this interface and the level of data encryption is defined in GSMA SAS SM specification [23].

The SM-DP+ MAY perform additional functions, which are out of scope of this specification.

5.3.1 Function: DownloadOrder

Related Procedures: Download initiation

Function Provider Entity: SM-DP+

Description:

This function is used to instruct the SM-DP+ of a new Profile download request.

The EID is optional and MAY not be known at this stage. In this case the SM-DP+, with the Operator, MAY verify if the EID acquired during the Download and installation procedure is compatible with the requested Profile Type (see also Annex F). If the SM-DS or Default SM-DP+ is to be used for the Profile download, then the EID SHOULD be present; if not present, the EID SHALL be provided later in "ES2+.ConfirmOrder".

Upon reception of this function call, the SM-DP+ SHALL:

- Reserve an ICCID in its inventory. If the ICCID was provided as input data, the reservation SHALL use this value. Otherwise the reservation SHALL be done corresponding to the requested Profile Type with a value available in the SM-DP+'s inventory.
- Optionally, if not already done, the SM-DP+ performs the 'Profile generation' and 'Profile protection' steps, as described in section 2.5.3, for the Profile identified by its ICCID.
- If the EID is known, the ICCID is linked to this EID and the Profile state SHALL be set to "Linked". Otherwise, the Profile state SHALL be set to "Allocated".

The SM-DP+ 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 ICCID has been reserved.
- A 'Function execution status' indicating 'Failed' with a status code as defined in table 24 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
eid	Identification of the targeted eUICC.	EID	1	O
iccid	Identification of the Profile to download and install in the eUICC.	ICCID	1	C
profileType	Identification of the Profile Type to download and install in the eUICC.	String	1	C

Table 22: DownloadOrder Additional Input Data

NOTE: The Operator can provide the ICCID and/or the Profile Type. In case where the Profile Type is provided, the SM-DP+ is free to select one of the Profiles that matches the Profile Type.

Additional Output Data:

Output data name	Description	Type	No.	MOC
iccid	Identification of the Profile to download and install in the eUICC. If ICCID was provided as an input data, the returned value SHALL be the same. If not provided as an input data the returned value SHALL be one of the values available in the SM-DP+ inventory and corresponding to the Profile Type.	ICCID	1	M

Table 23: DownloadOrder Additional Output Data

Specific status codes

Subject Code	Subject	Reason code	Reason	Description
8.2.1	Profile ICCID	3.9	Unknown	Indicates that the Profile, identified by this ICCID is unknown to the SM-DP+.
8.2.1	Profile ICCID	1.2	Not Allowed (Authorisation)	Indicates that the function caller is not allowed to perform this function on the target Profile.
8.2.1	Profile ICCID	3.3	Already in Use	Indicates that the Profile identified by the provided ICCID is not available.
8.2.5	Profile Type	3.9	Unknown	Indicates that the Profile Type identified by this Profile Type is unknown to the SM-DP+.
8.2.5	Profile Type	1.2	Not Allowed (Authorisation)	Indicates that the function caller is not allowed to perform this function on the Profile Type.

8.2.5	Profile Type	3.7	Unavailable	No more Profile available for the requested Profile Type.
8.2.5	Profile Type	3.8	Refused	Indicates that the Profile Type identified by this Profile Type is not aligned with the Profile Type of Profile identified by the ICCID.

Table 24: DownloadOrder Specific Status Codes

5.3.2 Function: ConfirmOrder

Related Procedures: Download initiation

Function Provider Entity: SM-DP+

Description:

This function is used to confirm a previously requested download order.

If the SM-DS or Default SM-DP+ is to be used for the Profile download and the EID has not been provided within the DownloadOrder function, then the EID SHALL be present. If EID is not present, the SM-DP+ SHALL return a 'Function execution status' indicating 'Failed' with a status code '8.1.1 EID - 2.2 Mandatory Element Missing'.

If the EID is present in both the DownloadOrder and ConfirmOrder functions it SHALL be the same value. If EID is different, the SM-DP+ SHALL return a 'Function execution status' indicating 'Failed' with a status code '8.1.1 EID - 3.10 Invalid Association'.

On reception of this function call, the SM-DP+ SHALL:

- Confirm the allocation of an ICCID in its inventory.
- Generate a MatchingID (section 4.1.1) if it is not provided by the Operator.
- Store the MatchingID.
- Store the EID if available.
- If the Confirmation Code is provided by the Operator, calculate the hash of the Confirmation Code and store the hash value together with the MatchingID, where the hash value is SHA256(Confirmation Code).
- If SM-DS address is provided:
    - Verify that the MatchingID is not a zero length value.
    - Store the SM-DS address with the Profile to be used later for Event Registration and Event Deletion.
    - If the releaseFlag is set to true, perform Event Registration to the SM-DS address stored with the Profile as defined in section 3.6.1, where the MatchingID SHALL be used as the EventID. Otherwise the Event Registration at this point in time is optional. If it is not done in this step, it will be done during the ReleaseProfile function.

The SM-DP+ MAY perform additional operations.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the ICCID has been reserved.
- 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
iccid	Identification of the Profile to download and install in the eUICC	ICCID	1	M
eid	Identification of the targeted eUICC	EID	1	O
matchingId	The MatchingID as defined in section (4.1.1), when generated by the Operator	String	1	O
confirmationCode	A code used to authorise the usage of the MatchingID to confirm the download and installation of the Profile	String	1	O
smdsAddress	The SM-DS address to be used for Event Registration	FQDN	1	O
releaseFlag	If 'true', the Profile SHALL be immediately released for Profile download and installation	Boolean	1	M

Table 25: ConfirmOrder Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
eid	Identification of the targeted eUICC. EID SHALL be returned if bound to this order.	EID	1	C
matchingId	The MatchingID as defined in section (4.1.1).	String	1	M
smdpAddress	The SM-DP+ address to be used for this specific download order.	FQDN	1	O

Table 26: ConfirmOrder Additional Output Data

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.2.1	Profile ICCID	3.9	Unknown	Indicates that the Profile, identified by this ICCID is unknown to the SM-DP+.
8.2.1	Profile ICCID	1.2	Not Allowed (Authorisation)	Indicates that the function caller is not allowed to perform this function on the target Profile.
8.2.6	Matching ID	3.3	Already in Use (Uniqueness)	Conflicting MatchingID value.
8.9	SM-DS	5.1	Inaccessible	Indicates that the smdsAddress is invalid or not reachable.

8.9	SM-DS	4.2	Execution Error	The cascade SM-DS registration has failed. SM-DS has raised an error.
8.1.1	EID	2.2	Mandatory Element Missing	Indicates that the EID is missing in the context of this order (SM-DS address provided or MatchingID value is empty).
8.1.1	EID	3.10	Invalid Association	Indicates that a different EID is already associated with this ICCID.

Table 27: ConfirmOrder Specific Status Codes

5.3.3 Function: CancelOrder

Related Procedures: Download initiation

Function Provider Entity: SM-DP+

Description:

This function is used to cancel a pending download order request.

On reception of this function call, the SM-DP+ SHALL:

- Confirm that the Profile identified by the provided ICCID is allocated and not yet downloaded.
- If there is a MatchingID already associated with the ICCID, then check that the provided MatchingID is the one associated with the ICCID.
- If there is an EID already associated with the ICCID, then check that the provided EID is the one associated with the ICCID.

If any of these verifications fail, the SM-DP+ SHALL return a 'Function execution status' indicating 'Failed' with the relevant status code.

Otherwise the SM-DP+ SHALL:

- Cancel the pending order if possible, and return the ICCID to inventory or mark it as not available for future use, based on the provided final Profile status indicator.
- If the order was previously linked to an event registration, the SM-DP+ SHALL subsequently execute the event deletion procedure.

The SM-DP+ MAY perform additional operations.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the ICCID has been released from the MatchingID and the associated profile will not be downloaded.
- 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 table here after.

Additional Input Data:

Input data name	Description	Type	No.	MOC
iccid	Identification of the Profile to be cancelled from previously requested download order.	ICCID	1	M
eid	eUICC Identifier.	EID	1	C
matchingId	The MatchingID as generated in ConfirmOrder.	String	1	C
finalProfileStatusIndicator	An indicator uses to indicate to the SM-DP+ to perform additional operations.	String	1	M

Table 28: CancelOrder Additional Input Data

The eid input data SHALL be provided if an EID has been associated for the download order to cancel.

The matchingId input data SHALL be provided if a MatchingID has been allocated for the download order to cancel.

Final Profile Status Indicator	Description
Available	Indicates that the download order for this Profile, identified by this ICCID will be cancelled; and the ICCID is released back to the inventory and available for future use.
Unavailable	Indicates that the download order for this Profile, identified by this ICCID will be cancelled; and the ICCID is not available for future use.

Table 29: Definition of Final Profile Status Indicator

Additional Output Data:

No additional output data.

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.2.1	Profile ICCID	3.9	Unknown	Indicates that the Profile, identified by this ICCID is unknown to the SM-DP+.
8.2.1	Profile ICCID	1.2	Not Allowed (Authorisation)	Indicates that the function caller is not allowed to perform this function on the target Profile.
8.2.1	Profile ICCID	3.3	Already in Use	The profile, identified by this ICCID, is already downloaded.
8.2.1	Profile ICCID	3.10	Invalid Association	Indicates that a different EID is associated with this ICCID.
8.2.6	Matching ID	3.10	Invalid Association	Indicates that a different MatchingID is associated with this ICCID.

Table 30: CancelOrder Specific Status Codes

5.3.4 Function: ReleaseProfile

Related Procedures: Download initiation

Function Provider Entity: SM-DP+

Description:

This function is used to release the Profile in order to allow the End User to start the download and installation procedure after the Operator performs any relevant operation on its back-end (e.g. provisioning of HLR).

On reception of this function call, the SM-DP+ SHALL:

- Verify that the Profile identified by the provided ICCID has been processed with "ES2+.DownloadOrder" and "ES2+.ConfirmOrder", but not released yet. If this verification fails, the SM-DP+ SHALL return error status "8.2.1 Profile ICCID - 3.5 Invalid transition"
- Set the Profile state as 'Released' to allow the download.
- If SM-DS address was stored with the Profile and if the Event Registration was not already done, perform Event Registration to the SM-DS as defined in section 3.6.1, where the MatchingID SHALL be used as the EventID.

The SM-DP+ 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 Profile identified by the provided ICCID has been released and is ready to download.
- 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 table here after.

Additional Input Data:

Input data name	Description	Type	No.	MOC
iccid	Identification of the Profile to be released from previously requested download order.	ICCID	1	M

Table 31: ReleaseProfile Additional Input Data

Additional Output Data:

No additional output data.

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.2.1	Profile ICCID	3.9	Unknown	Indicates that the Profile, identified by this ICCID, is unknown to the SM-DP+.
8.2.1	Profile ICCID	1.2	Not Allowed (Authorisation)	Indicates that the function caller is not allowed to perform this function on the target Profile.

8.2.1	Profile ICCID	3.5	Invalid transition	Indicates that the target Profile cannot be released.

Table 32: ReleaseProfile Specific Status Codes

NOTE: If a Profile has already been released this function will return 'Executed-Success' and take no other action.

5.3.5 Function: HandleDownloadProgressInfo

Related Procedures: Profile Download and Installation

Notification Handler/Recipient: Operator

Description:

This function SHALL be used by the SM-DP+ to notify the Operator of the progress of a pending Profile download order request. This function MAY be used at several points of the Profile Download and Installation procedure. It is assumed that the ICCID and the EID are enough to identify the pending Profile download order request at the SM-DP+ and the Operator sides. It is also assumed that the ICCID is enough for the SM-DP+ to retrieve the Operator to notify.

What is performed by the Operator receiving this notification is out of scope of this specification.

Additional Input Data:

Input data name	Description	Type	No.	MOC
eid	Identifies the targeted eUICC. This information SHALL be set if available to the SM-DP+.	EID	1	C
iccid	Identifies the Profile to download and install in the eUICC.	ICCID	1	M
profileType	Identifies the Profile Type to download and install in the eUICC.	String	1	M
timestamp	Indicates the date/time when the operation has been performed.	DATETIME	1	M
notificationPointId	Indicates the step reached within the Profile Download and Installation procedure. Defined check points are (1): '1' -> Eligibility and retry limit check '2' -> Confirmation Failure '3' -> BPP download '4' -> BPP installation	INTEGER	1	M
notificationPointStatus	Indicates the status after the execution of the notification point. The ExecutionStatus type is re-used to specify the result of processing of the operation related	ExecutionStatus	1	M

	to the notification point (Executed-Success, Executed-WithWarning, Failed), and optionally to provide information on any encountered problem (status code, data/object that causes the status code, and message to provide textual and human readable explanation of the status code).			
resultData	The finalResult data object as contained in the ProfileInstallationResult, when received from the eUICC.	Binary	1	C
NOTE 1: This specification reserves values from 0 to 99 for future use. The Operator and the SM-DP+, based on agreed behaviour, MAY define additional custom notification points. In that case, values >=100 SHALL be used.				

Table 32a: HandleDownloadProgressInfo Additional Input Data

The following table provides the mapping between the cancel session reason code received within the ES9+.CancelSession and the status code that SHALL be set in the notificationPointStatus input data.

Cancel reason code	Status code
endUserRejection(0)	'8.8.5 Download Order - 3.8 Refused'
pprNotAllowed(3)	'8.2.8 PPR - 1.2 Not Allowed'
metadataMismatch(4)	'8.2.9 Profile Metadata - 3.11 Value has Changed'
loadBppExecutionError(5)	'8.2.10 Bound Profile Package – 4.2 Execution Error'
undefinedReason(127)	'8 eUICC Remote Provisioning - 4.2. Execution Error'

Table 32b: Cancel session reason code mapping to Status code

NOTE: Both `postponed(1)` and `timeout(2)` cancel session reason codes do not terminate the download order, and thus do not lead to a notification to the Operator.

The following table provides additional status codes that can be set in the notificationPointStatus input data by the SM-DP+:

Reason	Status code
Maximum number of attempts of Profile Download has been exceeded	'8.8.5 Download Order - 6.4 Maximum number of retries exceeded'
Maximum number of attempts of Confirmation Code verification has been exceeded	'8.2.7 Confirmation Code – 6.4 Maximum number of retries exceeded'

Table 32c: Additional Status Codes

5.4 ES6 (Operator -- eUICC)

This interface is present between the Operator and their Enabled Profile in eUICC. It allows the Operator to make modifications on their Profile in the eUICC using legacy OTA mechanisms.

The ES6 functions are addressed to the eUICC through a secure channel, as defined in ETSI TS 102 225 [38] and ETSI TS 102 226 [39], established between the Operator and the MNO-SD of the Enabled Profile. This interface is the same as the one used with UICCs.

The initial OTA Key sets are part of the Profile and are loaded by the SM-DP+ during the "Profile Download and Installation" (section 3.1.3), or loaded by the EUM before eUICC issuance.

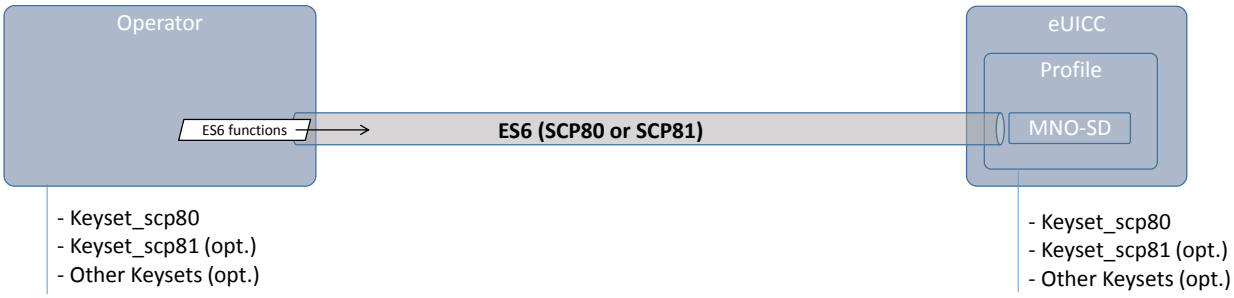


Figure 32: ES6

5.4.1 Function: UpdateMetadata

Related Procedures: N/A

Function Provider Entity: ISD-P

Description:

This function allows to update the following Profile Metadata of the targeted Profile:

- Service Provider name
- Profile Name
- Icon type and Icon
- Profile Policy Rule

As this function is provided by the ISD-P, the STORE DATA command message defined hereunder has to be preceded by an INSTALL [for personalisation] as defined in SGP.02 [2] section 4.1.2.1:

- The reserved AID value for Profile's ISD-P (Annex D) SHALL be used to indicate that the Security Domain targeted by the INSTALL [for personalisation] command is the ISD-P of the Profile containing the MNO-SD.
- According to GlobalPlatform Card Specification [8], INSTALL [for personalisation] command can only be used on applications associated with a Security Domain. As an

exception to this rule, the eUICC SHALL allow the MNO-SD to receive this command sequence with data destined to the ISD-P.

Command Message

This function uses the command message STORE DATA as defined in GlobalPlatform Card Specification [8] with the specific coding defined in this section.

Code	Value	Meaning
CLA	'80'	GPCS [8] section 11.11
INS	'E2'	STORE DATA
P1	'10' or '90'	See below
P2	Xx	Block number
Lc	xx	Length of data field
Data	'xx xx…'	See below
Le	'00'	

Table 33: UpdateMetadata Command Message

Parameter P1

The P1 SHALL be coded as follows:**Error! Reference source not found.**

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0/1	-	-	-	-	-	-	-	More blocks / Last block
-	0	0	-	-	-	-	-	No general encryption information or non-encrypted data
-	-	-	1	0	-	-	-	BER-TLV format of the command data field
-	-	-	-	-	-	-	0	Case 3 command as defined in GlobalPlatform AmdA [9]
-	-	-	-	-	X	X	-	RFU

Table 34: UpdateMetadata P1

The Profile Metadata values to update MAY not fit in a single STORE DATA, in that case the Operator SHALL use several STORE DATA. A transfer of an intermediate block of a TLV SHALL be done by indicating "More blocks". A last block of a TLV SHALL be transferred indicating "Last block".

Data Field

The data field SHALL be coded as defined hereunder.

```
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'
}
```

For profilePolicyRules:

This version of the specification only defines unsetting PPRs. For this operation, the `pprUpdateControl` bit SHALL be set to zero. A value of one SHALL be treated as an error.

If `pprUpdateControl` is set to zero, the following SHALL apply: Each PPR bit of a Profile SHALL be logically ANDed with the corresponding bit of the UpdateMetadataRequest.

'`serviceProviderName`', '`profileName`', '`iconType`', '`icon`' and '`profilePolicyRules`' fields are all optional. But at least one field SHALL be present.

'`iconType`' and '`icon`' SHALL be both present and contain non-empty values, be both present and contain empty values, or both be absent.

If a field is present with a non-empty value, the eUICC SHALL update the Profile Metadata with the provided value.

If a field is present with an empty value, the eUICC SHALL delete the actual value from the Profile Metadata.

If a field is absent, the eUICC SHALL not update the corresponding Profile Metadata value.

If there is an error while processing the UpdateMetadata, the Profile Metadata SHALL remain unchanged.

Response Message

Data Field

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

Processing State Returned in the Response Message

See GlobalPlatform Card Specification [8] section 11.11.3.2.

- The following additional status bytes are defined:
  - '6A 81': `pprUpdateControl` setting not supported

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

The ES8+ is an interface defined between the Profile Package Binding function of the SM-DP+ and the eUICC. This interface is intended to be tunnelled over the ES9+ and ES10b interfaces.

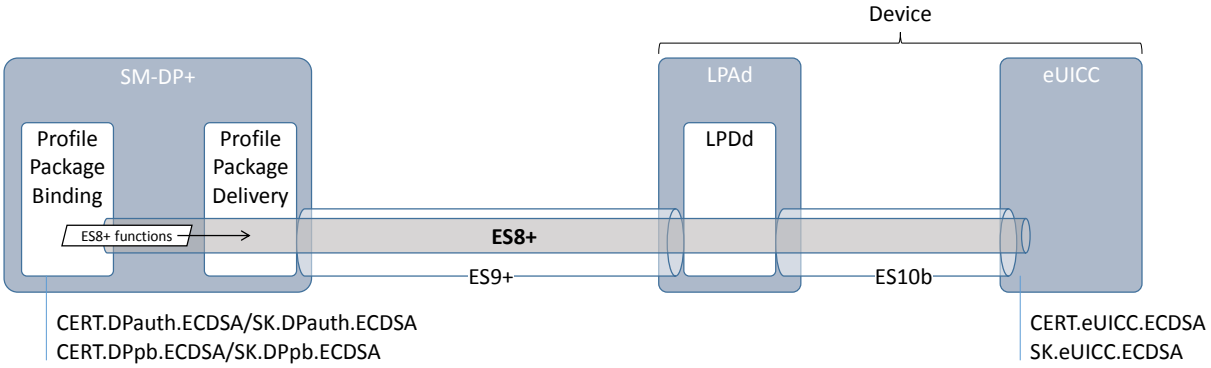


Figure 33: ES8+

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 secure channel is established by:

- Mutual authentication of the eUICC and the SM-DP+ using SK.DPauth.ECDSA / CERT.DPauth.ECDSA and SK.EUICC.ECDSA/CERT.EUICC.ECDSA.
- Session keys agreement based on exchanged one-time public keys of both parties during mutual authentication (Annex G).

The SM-DP+ authenticates the eUICC by:

- Verifying the CERT.EUICC.ECDSA with PK.EUM.ECDSA extracted from CERT.EUM.ECDSA, itself verified with PK.CI.ECDSA extracted from CERT.CI.ECDSA.
- Verifying the signature of the eUICC computed over an SM-DP+ challenge with PK.EUICC.ECDSA extracted from the verified CERT.EUICC.ECDSA.

The eUICC authenticates the SM-DP+ by:

- Verifying the CERT.DPauth.ECDSA with PK.CI.ECDSA.
- Verifying the signature of the SM-DP+ with the PK.DPauth.ECDSA extracted from the verified CERT.DPauth.ECDSA.

The data exchanged after channel establishment are secured using SCP03t as defined in SGP.02 [2]. The eUICC SHALL support the SCP03t with:

- AES in CBC mode with key length of 128 bits, referred as AES-128 (key length as defined in SGP.02 [2]).
- Use of C-MAC and C-DECRYPTION.

As a result the SM-DP+ and eUICC are mutually authenticated, all data sent from the Profile Package Binding function of the SM-DP+ to the eUICC are MACed and encrypted.

Response data generated by the eUICC when processing the BPP received on ES8+ is returned protected by a signature generated by the eUICC.

5.5.1 Function: InitialiseSecureChannel

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used by the SM-DP+ to open a new RSP session with the target eUICC. The function carries the identifier of the remote operation type to be performed by the eUICC (e.g. installation of a new Bound Profile Package) and the necessary material for key agreement with Perfect Forward Secrecy (PFS), allowing a secure end-to-end communication between the SM-DP+ and the eUICC:

- Transaction ID
- Description of the keys to generate
- One-time public key for key agreement generated by SM-DP+ (otPK.DP.ECKA)
- Signature upon material (including the previously generated otPK.EUICC.ECKA, also acting as an eUICC challenge) to ensure its integrity and authenticity.

The Transaction ID SHALL be unique within the scope and lifetime of each SM-DP+.

NOTE: Transaction IDs not being reused protects against attacks which try to replay CancelSession messages.

The level of security is implicitly deduced from the remote operation type to execute.

The reception of the InitialiseSecureChannel function SHALL be rejected if a secure channel session is already ongoing.

On reception of this command the eUICC SHALL:

- Verify the SM-DP+ signature using the PK.DPbp.ECDSA; if the signature is invalid the command SHALL be rejected, an error SHALL be returned, Profile installation SHALL be aborted, and any contextual data associated to its Profile installation (like the SM-DP+ certificate) SHALL be discarded.
- Verify that the requested Remote operation type is one of the defined types.
- Verify that the received transaction ID matches the transaction ID of the on-going RSP session (section 5.7.5 "ES10b.PrepareDownload" function).
- Verify that Control Reference Template describing the keys to generate matches the values defined here under (Command message part).

If any of these verifications fail, the eUICC SHALL return an error.

If these verifications are successful, the eUICC SHALL:

- Generate the session keys (S-ENC and S-MAC) and the initial MAC chaining value from received otPK.DP.ECKA and previously generated otSK.EUICC.ECKA.

Command Data

The command data for this function is encoded in the ASN.1 data object as described below.

```
--Definition of data objects for InitialiseSecureChannel Request
InitialiseSecureChannelRequest ::= [35] SEQUENCE { -- Tag 'BF23'
   remoteOpId RemoteOpId, -- Remote Operation Type Identifier (value SHALL be set
to installBoundProfilePackage)
   transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
   controlRefTemplate[6] IMPLICIT ControlRefTemplate, -- Control Reference Template
(Key Agreement). Current specification considers a subset of CRT specified in
GlobalPlatform Card Specification [8] section 6.4.2.3 for the Mutual Authentication
Data Field
   smdpOtpk [APPLICATION 73] OCTET STRING, ---otPK.DP.ECKA as specified in
GlobalPlatform Card Specification [8] section 6.4.2.3 for ePK.OCE.ECKA, tag '5F49'
   smdpSign [APPLICATION 55] OCTET STRING -- SM-DP's signature, tag '5F37'
}

ControlRefTemplate ::= SEQUENCE {
   keyType[0] Octet1, -- Key type according to GlobalPlatform Card Specification
[8] Table 11-16, AES= '88' , Tag '80'
   keyLen[1] Octet1, --Key length in number of bytes. For current specification key
length SHALL by 0x10 bytes, Tag '81'
   hostId[4] OctetTo16 -- Host ID value , Tag '84'
}
```

NOTE: The tag '90' for ‘SCP identifiers and parameters’ is not used. This specification only uses one SCP type derived from SCP11a defined in GlobalPlatform Card Specification Amendment F [13]. The tag '95' for ‘Key Usage Qualifier’ is also not used. This is determined by the ‘Remote operation type identifier’ (see hereunder). Since only AES key type is specified currently, key values from [8] Table 11-16 are not used.

The eUICC SHALL verify the values provided for key type and key length.

SM-DP+ signature (smdpSign) is computed as described in section 2.6.7.2, using the SM-DP+ private key SK.DPbp.ECDSA across the following concatenated data objects:

- remoteOpId
- transactionId
- controlRefTemplate
- smdpOtpk
- euiccOtpk, as provided earlier in the prepareDownloadResponse data object received in the "ES9+.GetBoundProfilePackage" function.

As the signature includes the otPK.EUICC.ECKA, the eUICC can authenticate the SM-DP+.

When type is 'Install Bound Profile Package', the implicit Key Usage Qualifier SHALL be set to MAC and ENCRYPTION.

The eUICC SHALL return an error '05' for any other Remote operation type identifier value.

If all checking are valid, the eUICC SHALL process the key derivation as described in Annex G.

5.5.2 Function: ConfigureISDP

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used by the SM-DP+ to provide data to the eUICC for configuring the ISD-P. For this version of the specification, this data element only contains the optional SM-DP+ proprietary data.

NOTE: Information like the amount of assigned memory MAY be added in future versions.

On reception of this command the eUICC SHALL:

- Create the ISD-P for the Profile and assign an AID value from the range reserved for ISD-Ps in SGP.02 [2].
- If the length of the SM-DP+ proprietary data exceeds the maximum size, terminate with error 'incorrectInputValues'.
- Store the SM-DP+ proprietary data in the ISD-P.

Command data

The command data for this function is encoded in the ASN.1 data object below.

```
--Definition of data objects for ConfigureISDPRequest
ConfigureISDPRequest ::= [36] SEQUENCE { -- Tag 'BF24'
   dpProprietaryData [24] DpProprietaryData OPTIONAL -- Tag 'B8'
}

DpProprietaryData ::= SEQUENCE { -- maximum size including tag and length field:
128 bytes
   dpOid OBJECT IDENTIFIER -- OID in the tree of the SM-DP+ that created the
Profile
   -- additional data objects defined by the SM-DP+ MAY follow
}
```

5.5.3 Function: StoreMetadata

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used by the SM-DP+ to provide Profile Metadata of the Profile to the eUICC.

On reception of this command the eUICC SHALL verify the following:

- The Profile Class is supported.
- The ICCID is different than that of all other installed profiles.
- The PPRs in the Profile, if any, are allowed for the Profile Owner. This verification SHALL be done as described section 2.9.3.1.

If any verification fails, the eUICC SHALL report an error and stop the profile installation procedure. Otherwise store the data elements for future use.

Command data

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

```
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'
}

NotificationEvent ::= BIT STRING {
   notificationInstall(0),
   notificationEnable(1),
   notificationDisable(2),
   notificationDelete(3)
}

NotificationConfigurationInformation ::= SEQUENCE {
   profileManagementOperation NotificationEvent,
   notificationAddress UTF8String -- FQDN to forward the notification
}
```

Each of the `notificationInstall(0)`, `notificationEnable(1)`, `notificationDisable(2)`, `notificationDelete(3)` MAY appear several times in the sequence of `notificationConfigurationInfo` data object. In that case, it specifies several recipient addresses for the same notification event.

The data object `profileOwner` is optional. It SHALL be present if the `profilePolicyRules` data object is present. In this instance the `mccMnc` field SHALL not specify any wildcard ('E') digits.

The data object `profilePolicyRules` is optional. A Profile that has no PPR SHALL not have this data object set, else the `profilePolicyRules` SHALL identify all the PPRs set in the Profile. The `PrdIds` type is defined in section 2.8.1.1.

5.5.4 Function: ReplaceSessionKeys

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used to replace the SCP03t session keys (S-ENC and S-MAC) during the loading of a Bound Profile Package by a new set of session keys (typically the PPK-ENC and PPK-CMAC (section 2.5). Note that both keys are replaced; this function doesn't allow replacement of only one of the session keys.

On reception of this function the eUICC SHALL:

- Verify that the new keys are of same length as the old keys. If not the eUICC SHALL return an error, and the loading of the BPP SHALL be aborted.
- Replace the current session keys with the new set of keys.

Once the function is successfully executed, the eUICC SHALL use this new set of keys for decryption and MAC verification of subsequent SCP03t blocks of data. The key type of the new set of keys is the same as the session keys they replace.

Command data

The command message for this function is encoded in the ASN.1 data object below.

```
-- Definition of request message for command ReplaceSessionKeys
ReplaceSessionKeysRequest ::= [38] SEQUENCE { -- tag 'BF26'
/*The new initial MAC chaining value*/
   initialMacChainingValue OCTET STRING,
/*New session key value for encryption/decryption (PPK-ENC)*/
   ppkEnc OCTET STRING,
/*New session key value of the session key C-MAC computation/verification (PPK-
MAC)*/
   ppkCmac OCTET STRING
}
```

5.5.5 Function: LoadProfileElements

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used by the SM-DP+ to provide the Profile Elements defined by SIMalliance specification [5] to the eUICC.

Command messages, response messages and the processing on the eUICC are defined in SIMalliance specification [5].

The eUICC SHALL ignore the ICCID value provided in the 'ProfileHeader' PE.

The eUICC SHALL verify that the following values provided in the Profile Metadata via "ES8+.StoreMetadata" are reflected in the content of EFs of the Profile:

- The ICCID provided in the Profile Metadata is identical to the value of EF_{ICCID}.
- The `mccMnc` value provided in the Profile Metadata SHALL match the MCC and MNC values in EF_{IMSI}.
- If `gid1` or `gid2` is provided in the Profile Metadata: The corresponding EF_{GID1} or EF_{GID2} SHALL be present and the related service in EF_{UST} SHALL indicate "available".
- If `gid1` or `gid2` is not provided in the Profile Metadata: The corresponding service in EF_{UST} for EF_{GID1} or EF_{GID2} SHALL indicate "not available".

Any failure SHALL be indicated by an `installFailedDueToDataMismatch` error.

If the Profile is a Test Profile, the eUICC SHALL check if the key(s) for network authentication follow the requirements defined in section 2.4.5.3.

If the Profile contains PPRs, the eUICC SHALL verify that the Profile Owner mcc and mnc values provided in the Profile Metadata are the same as the MCC and MNC values contained in the EF_{IMSI} of the Profile; and, if set, that Profile Owner `gid1` and `gid2` values provided in the Profile Metadata are the same as the EF_{GID1} and EF_{GID2}. A failure SHALL be indicated by an `installFailedDueToDataMismatch` error.

On any error during the processing of a Profile Element, the Profile installation SHALL be stopped and the ISD-P and all the related Profile Components SHALL be deleted.

If the Profile is successfully installed, the eUICC SHALL first generate the Profile Installation Result and then as many Notifications as configured in its metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.

5.6 ES9+ (LPA -- SM-DP+)

ES9+ is the interface between:

- The LPA entity (more specifically the LPD endpoint).
- The SM-DP+ (more specifically the Profile Package Delivery endpoint).

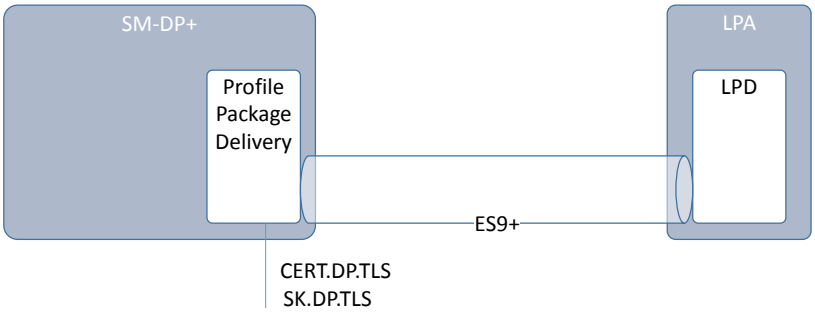


Figure 34: ES9+

The LPA SHALL communicates with the SM-DP+ secured by HTTPS in server authentication mode as described in section 2.6.6.

The format of the TLS Certificates (CERT.DP.TLS) used for TLS connections is described in section 4.5.2.1.

During TLS establishment, the LPA SHALL verify the received CERT.DP.TLS according to section 4.5.2.2. If any of these verifications fail, the TLS connection SHALL be rejected, and the on-going procedure SHALL fail.

5.6.1 Function: InitiateAuthentication

Related Procedures: Profile Download and Installation

Function Provider Entity: SM-DP+

Description:

This function requests the SM-DP+ authentication. This is following the "GetEUICCChallenge" between the eUICC and the LPAd, where the LPAd retrieves material from the eUICC to be provided to the SM-DP+.

On reception of this function call, the SM-DP+ SHALL:

- Verify that it supports the Specification Version Numberindicated by the eUICC.
- Check if the received address matches its own SM-DP+ address.
- Check if it can use one of the GSMA CI Public Keys against which eUICC signatures can be verified, and select the CI as defined in section 2.6.7.1.
- Verify that it can provide a CERT.DPauth.ECDSA signed by one of the GSMA CI Public Keys supported by the eUICC and select a CERT.DPauth.ECDSA preferably according to the priority provided by the eUICC for the CI Public Keys.

If any of these verifications fail, the SM-DP+ SHALL return a 'Function execution status' indicating 'Failed' with the relevant status code.

Otherwise the SM-DP+ SHALL:

- Generate a TransactionID which is used to uniquely identify the ongoing RSP session.
- Generate a serverChallenge for eUICC authentication attached with the ongoing RSP session.
- Generate a serverSigned1 data object as expected by the eUICC and described in section 5.7.13 "ES10b.AuthenticateServer".
- Generate a signature (serverSignature1) as described in section 5.7.13 "ES10b.AuthenticateServer" using the SK related to the selected CERT.DPauth.ECDSA.

The SM-DP+ 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 RSP session has been successfully initiated.
- 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
euiccChallenge	euiccChallenge generated by the eUICC. LPDd can retrieve the euiccChallenge from the eUICC by calling "ES10b.GetEUICCChallenge" (section 5.7.7).	Binary[16]	1	M

euiccInfo1	euiccInfo1 of the eUICC. LPDd can retrieve the euiccInfo1 by calling "ES10b.GetEUICCInfo" (section 5.7.8).	Binary(1)	1	M
smdpAddress	The SM-DP+ Address as known and provided by the LPA.	FQDN	1	M
NOTE 1: euiccInfo1 SHALL be provided as an encoded EuiccInfo1 data object.				

Table 35: InitiateAuthentication Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
transactionId	Transaction ID as generated by the SM-DP+ (section 3.1.1.4).	Binary[1-16]	1	M
serverSigned1	The data object signed by the SM-DP+.	Binary(1)	1	M
serverSignature1	SM-DP+ signature.	Binary(1)	1	M
euiccCiPKIdToBeUsed	CI Public Key Identifier to be used by the eUICC for signature.	Binary(1)	1	M
serverCertificate	SM-DP+ Certificate (CERT.DPauth.ECDSA).	Binary(1)	1	M
NOTE 1: serverSigned1, serverSignature1, euiccCiPKIdToBeUsed and serverCertificate are data objects defined in section 5.7.13 (function "ES10b.AuthenticateServer"). They SHALL be returned as encoded data objects including the tags defined for them in the AuthenticateServerRequest data object.				

Table 36: InitiateAuthentication Additional Output Data

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.8.1	SM-DP+ Address	3.8	Refused	Invalid SM-DP+ Address.
8.8.2	Security configuration	3.1	Unsupported	None of the proposed Public Key Identifiers is supported by the SM-DP+.
8.8.3	Specification Version Number	3.1	Unsupported	The Specification Version Number indicated by the eUICC is not supported by the SM-DP+.
8.8.4	SM-DP+ Certificate	3.7	Unavailable	The SM-DP+ has no CERT.DPAuth.ECDSA signed by one of the GSMA CI Public Key supported by the eUICC.

Table 37: InitiateAuthentication Specific Status codes

5.6.2 Function: GetBoundProfilePackage

Related Procedures: Profile Download and Installation

Function Provider Entity: SM-DP+

Description:

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

End point of this function on the SM-DP+ side is the Profile Package Delivery which is in charge to deliver the input data to the Profile Package Binding.

The Profile Package Binding output data is delivered to the LPA through the Profile Package Delivery.

This function is correlated to a previous normal execution of an "ES9+.AuthenticateClient" function through a TransactionID delivered by the SM-DP+.

On reception of this function call, the SM-DP+ SHALL:

- Verify that the received transactionId is known and relates to an ongoing RSP session.
- Verify the eUICC signature (euiccSignature2) using the PK.EUICC.ECDSA attached to the ongoing RSP session as described in (section 5.7.5 "ES10b.PrepareDownload").
- Verify if this order requires a Confirmation Code verification; if yes, the SM-DP+ SHALL verify that the received Hashed Confirmation Code matches the value known by the SM-DP+.

If any of these verifications fail, the SM-DP+ SHALL return a 'Function execution status' indicating 'Failed' with the relevant status code. If the SM-DP+ determines that the related Profile download order has expired, the relevant status code is "8.8.5 Download order - 4.10. Time to Live Expired".

If the maximum number of retries for Confirmation Code verification has been exceeded, the corresponding Profile download order SHALL be terminated.

If the Bound Profile Package has been previously generated for this eUICC, the SM-DP+ SHALL check if the otPK.EUICC.ECKA provided by the eUICC is the same as the one used to generate this BPP. If so, the BPP can be re-used: only the signature for InitialiseSecureChannel needs to be recalculated.

If the Bound Profile Package has been previously generated for this eUICC, but the otPK.EUICC.ECKA provided by the eUICC is different than the one previously used to generate this BPP, the SM-DP+ SHALL either terminate the procedure with an error or re-bind the PPP as described below.

To bind the PPP, the SM-DP+ SHALL:

- Attach the otPK.EUICC.ECKA to the ongoing RSP session.

- Generate one time ECKA key pair (otPK.DP.ECKA, otSK.DP.ECKA) for key agreement.
- Generate the session keys (S-ENC and S-MAC) and the initial MAC chaining value from received otPK.EUICC.ECKA and previously generated otSK.DP.ECKA.
- Generate the Profile Metadata of the Profile.
- Generate the Bound Profile Package as described in (section 2.5.4).
- Erase otSK.DP.ECKA immediately once BPP is generated.

The SM-DP+ 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 BoundProfilePackage has been successfully built and is part of the output data.
- 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	MOC
transactionId	Transaction ID as generated by the SM-DP+ (section 3.1.1.4).	Binary[1-16]	M
prepareDownloadResponse	PrepareDownloadResponse data object defined in section 5.7.4.	Binary(1)	M
NOTE 1: prepareDownloadResponse SHALL be provided as an encoded PrepareDownloadResponse data object			

Table 38: GetBoundProfilePackage Additional Input Data

Additional Output Data:

Output data name	Description	Type	MOC
transactionID	Transaction ID as generated by the SM-DP+ (section 3.1.1.4).	Binary[1-16]	M
boundProfilePackage	Bound Profile Package data object to be transferred to the eUICC using "ES10b.LoadBoundProfilePackage" (section 5.7.6).	Binary(1)	M
NOTE 1: boundProfilePackage SHALL be provided as an encoded BoundProfilePackage data object			

Table 39: GetBoundProfilePackage Additional Output Data

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.1	eUICC	6.1	Verification Failed	eUICC signature is invalid.

8.2	Profile	3.7	Unavailable	BPP is not available for a new binding.
8.10.1	TransactionId	3.9	Unknown	The RSP session identified by the TransactionID is unknown.
8.2.7	Confirmation Code	2.2	Mandatory Element Missing	Confirmation Code is missing.
8.2.7	Confirmation Code	3.8	Refused	Confirmation Code is refused.
8.2.7	Confirmation Code	6.4	Maximum number of retries exceeded	The maximum number of retries for the Confirmation Code has been exceeded.
8.8.5	Download order	4.10	Time to Live Expired	The Download order has expired.

Table 40: GetBoundProfilePackage Specific status codes

5.6.3 Function: AuthenticateClient

Related Procedures: Common Mutual Authentication

Function Provider Entity: SM-DP+

Description:

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

This function is correlated to a previous normal execution of an "ES9+.InitiateAuthentication" function through a Transaction ID delivered by the SM-DP+.

On reception of this function call, the SM-DP+ SHALL:

- Verify the validity of the CERT.EUM.ECDSA, using the related public key PK.CI.ECDSA.
- Verify the validity of the CERT.EUICC.ECDSA, using the public key PK.EUM.ECDSA.
- Verify the eUICC signature (euiccSignature1) using the PK.EUICC.ECDSA as described in section 5.7.13 "ES10b.AuthenticateServer".
- Verify that the transactionId is known and relates to an ongoing RSP session.
- Verify that the serverChallenge attached to the ongoing RSP session matches the serverChallenge returned by the eUICC.

If any of these verifications fail, the SM-DP+ SHALL return a 'Function execution status' indicating 'Failed' with the relevant status code. If the SM-DP+ determines that the related Profile download order has expired, the relevant status code is "8.8.5 Download order - 4.10. Time to Live Expired".

If the maximum number of retries for Profile download has been exceeded, the corresponding Profile download order SHALL be terminated.

Otherwise the SM-DP+ SHALL perform additional verification depending on the use case where this function is involved and the received ctxParams1.

If `ctxParams1` contains a `ctxParamsForCommonAuthentication` data object, the SM-DP+ SHALL:

- Verify there is a pending Profile download order for the incoming eUICC. For that, the SM-DP+ SHALL:
  - If there is a pending Profile download order associated with this EID:
    - If `ctxParamsForCommonAuthentication` contains a matchingId data object, verify that it matches the MatchingID for this pending Profile download order.
    - If `ctxParamsForCommonAuthentication` does not contain a matchingId data object, any pending Profile download order associated to this EID MAY be selected.
  - If there is no pending Profile download order associated with this EID:
    - If `ctxParamsForCommonAuthentication` contains a matchingId data object, verify there is a pending Profile download order associated with this MatchingID.
    - If `ctxParamsForCommonAuthentication` does not contain a matchingId data object, this SHALL be considered as verification failure.
- Identify the Profile corresponding to the pending Profile download order.
- Verify that the identified Profile has been released (Profile state is Released, see section 3.1.6).
- Perform an eligibility check as described in Annex F.

If any of these verifications fail, the SM-DP+ SHALL return a 'Function execution status' indicating 'Failed' with the relevant status code.

Otherwise the SM-DP+ SHALL:

- Generate the Profile Metadata of the Profile.
- Attach the PK.EUICC.ECDSA to the ongoing RSP session.
- Verify if this order requires a Confirmation Code verification. If yes, the SM-DP+ SHALL set the `ccRequiredFlag` data field of the `smdpSigned2` data object to true.
- Generate an smdpSigned2 data object and compute the signature smdpSignature2 as defined in "ES10b.PrepareDownloadRequest".

The SM-DP+ 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 eUICC has been successfully authenticated.
- 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
transactionId	Transaction ID as generated by the SM-DP+ (section 3.1.1.4).	Binary[1-16]	1	M
authenticateServerResponse	Authenticate Server Response.	Binary(1)	1	M
NOTE 1: AuthenticateServerResponse data object defined in section 5.7.13 (function "ES10b.AuthenticateServer").				

Table 41: AuthenticateClient Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
transactionID	Transaction ID as generated by the SM-DP+ (section 3.1.1.4).	Binary[1-16]	1	M
profileMetadata	Profile Metadata for the purpose of display by the LPA.	Binary(1)	1	C
smdpSigned2	The data to be signed by the SM-DP+.	Binary(1)	1	C
smdpSignature2	SM-DP+ signature.	Binary(1)	1	C
smdpCertificate	SM-DP+ Certificate (CERT.DPpb.ECDSA).	Binary(1)	1	C
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"). They SHALL be returned as encoded data objects including the tags defined for them in the StoreMetadataRequest/PrepareDownloadRequest data object.				

Table 42: AuthenticateClient Additional Output Data

profileMetadata, smdpSigned2, smdpSignature2 and smdpCertificate SHALL be provided when this function is called in the context of the Profile Download and Installation procedure as described in section 3.1.3.

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.1.2	EUM Certificate	6.1	Verification Failed	Certificate is invalid.
8.1.2	EUM Certificate	6.3	Expired	Certificate has expired.
8.1.3	eUICC Certificate	6.1	Verification Failed	Certificate is invalid.
8.1.3	eUICC Certificate	6.3	Expired	Certificate has expired.
8.1	eUICC	6.1	Verification Failed	eUICC signature is invalid or serverChallenge is invalid.

8.1	eUICC	4.8	Insufficient Memory	eUICC does not have sufficient space for this Profile.
8.11.1	CI Public Key	3.9	Unknown	Unknown CI Public Key. The CI used by the EUM Certificate is not a trusted root for the SM-DP+.
8.2	Profile	1.2	Not allowed	Profile has not been released.
8.10.1	TransactionId	3.9	Unknown	The RSP session identified by the TransactionID is unknown.
8.2.6	MatchingID	3.8	Refused	MatchingID (AC_Token or EventID) is refused.
8.1.1	EID	3.8	Refused	EID doesn't match the expected value.
8.2.5	Profile Type	4.3	Stopped on warning	No eligible Profile for this eUICC/Device.
8.8.5	Download order	4.10	Time to Live Expired	The Download order has expired.
8.8.5	Download order	6.4	Maximum number of retries exceeded	The maximum number of retries for the Profile download order has been exceeded.

Table 43: AuthenticateClient Specific Status Codes

5.6.4 Function: HandleNotification

Related Procedures: Notifications

Function Provider Entity: SM-DP+

Description:

This function SHALL be called by the LPA to notify the SM-DP+ that a Profile Management Operation has successfully been performed on the eUICC.

The SM-DP+ SHALL manage the Notification according to section 3.5 and acknowledge the LPA of the processing.

The SM-DP+ MAY perform additional operations which are out of scope of this specification.

Additional Input Data:

Input data name	Description	Type	MOC
pendingNotification	PendingNotification data object as defined in section 5.7.10	Binary(1)	M
NOTE 1: pendingNotification SHALL be provided as an encoded PendingNotification data object			

Table 44: HandleNotification Additional Input Data

Additional Output Data:

No additional output data.

5.6.5 Function: CancelSession

Related Procedures: Download and Installation

Function Provider Entity: SM-DP+

Description:

This function is to request the cancellation of an on-going RSP session upon a decision of the End User. This function MAY be used in different procedures.

This function is correlated to a previous normal execution of an "ES9+.AuthenticateClient" function through a transactionId delivered by the SM-DP+.

On reception of this function call, the SM-DP+ SHALL:

- Verify that the received transactionId is known and relates to an ongoing RSP session.
- Verify the eUICC signature (euiccCancelSessionSignature) using the PK.EUICC.ECDSA attached to the ongoing RSP session as described in (section 5.7.14 "ES10b.CancelSession").
- Verify that the received smdpOid corresponds to the SM-DP+ (i.e. is the same value as the one contained in the CERT.DPauth.ECDSA used during the Common Mutual Authentication Procedure).

If any of these verifications fail, the SM-DP+ SHALL return a 'Function execution status' indicating 'Failed' with the relevant status code.

Otherwise, the SM-DP+ SHALL return a function execution status 'Executed-Success', and perform additional operations depending on the context and the reason received, as described hereunder.

When used within a Profile Download and Installation procedure, and if the cancel session reason contained in euiccCancelSessionSigned indicates a terminal code (see section 5.7.14), the SM-DP+ SHALL:

1. Notify the Operator using the function "ES2+.HandleDownloadProgressInfo" function with the identification of the step reached in the on-going procedure and an operation status indicating 'Failed' with status code according to mapping given in section 5.3.5.
2. Terminate the corresponding pending download process.
3. If required, execute the SM-DS Event Deletion procedure described in section 3.6.3.

NOTE: The operations 1), 2) and 3) are described as performed in the context of this function execution. Alternatively they MAY be done asynchronously by the SM-DP+. Operation 2) and 3) MAY not be performed depending on the agreed SM-DP+ behaviour with the Operator. If the operations are not performed, the Operator has the responsibility to take care of the management of the Download Order, e.g. by calling the "ES2+.CancelOrder" on reception of the notification "ES2+.HandleDownloadProgressInfo".

The SM-DP+ 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 RSP session has been cancelled.
- 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
transactionId	Transaction ID as generated by the SM-DP+ (section 3.1.1.4).	Binary[1-16]	1	M
cancelSessionResponse	Defined in "ES10b.CancelSession" function, section 5.7.14	Binary(1)	1	M
NOTE 1: cancelSessionResponse SHALL be provided as an encoded `CancelSessionResponse` data object				

Table 45: CancelSession Additional Input Data

Additional Output Data:

No output data.

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.10.1	TransactionId	3.9	Unknown	The RSP session identified by the TransactionID is unknown.
8.1	eUICC	6.1	Verification Failed	eUICC signature is invalid.
8.8	SM-DP+	3.10	Invalid Association	The provided SM-DP+ OID is invalid.

Table 46: CancelSession Specific status codes

5.7 ES10x (LPA -- eUICC)

ES10 contains 3 different interfaces described below.

The ES10a is an interface defined between the LDSd and ISD-R (LPA Services).

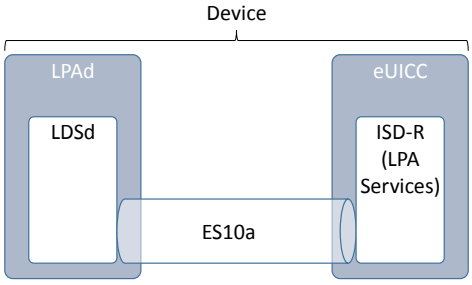


Figure 35: ES10a

The ES10b is an interface defined between the LPDd and ISD-R (LPA Services).

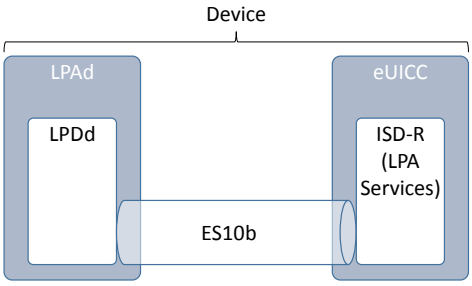


Figure 36: ES10b

The ES10c is an interface defined between the LUI and ISD-R (LPA Services).

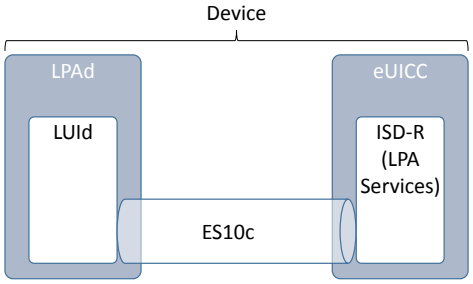


Figure 37: ES10c

5.7.1 ISD-R Selection and LPAe Activation

Before sending any command to the eUICC, the LPA 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 [8]. This MANAGE CHANNEL and SELECT commands can be intrinsically used via a dedicated Device OS API (e.g. OMAPI defined by SIMalliance [OMAPI] if provided).

The Device SHALL ensure that only the LPA, but no other application on the Device, is permitted to select the ISD-R.

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 [8]:

```
ISDRProprietaryApplicationTemplate ::= [PRIVATE 0] SEQUENCE { -- Tag 'E0'
   svn [2] VersionType,     -- GSMA SGP.22 version supported (SVN)
   lpaeSupport BIT STRING {
      lpaeUsingCat(0), -- LPA in the eUICC using Card Application Toolkit
      lpaeUsingScws(1) -- LPA in the eUICC using Smartcard Web Server
   } OPTIONAL
```

```
}
```

NOTE: eUICCs according to version 1.X of this specification will not return this data structure.

If the Device supports the requirements for an option of the LPAe as defined in section 5.11 and the eUICC indicated support for that option in the ISDRProprietaryApplicationTemplate, the Device MAY activate this option by sending an LpaeActivationRequest to the ISD-R.

If the Device indicates support for LUId, LPDd and LDSd and it does not send an LpaeActivationRequest, the eUICC SHALL not activate the LPAe.

In all other cases, the eUICC MAY activate the LPAe.

The LpaeActivationRequest SHALL be sent to the ISD-R using the transport mechanism defined in section 5.7.2.

The command data SHALL be coded as follows:

```
LpaeActivationRequest ::= [66] SEQUENCE { -- Tag 'BF42'
   lpaeOption BIT STRING {
      activateCatBasedLpae(0), -- LPAe with LUIe based on CAT
      activateScwsBasedLpae(1) -- LPAe with LUIe based on SCWS
   }
}
```

The response data SHALL be coded as follows:

```
LpaeActivationResponse ::= [66] SEQUENCE { -- Tag 'BF42'
   lpaeActivationResult INTEGER {ok(0), notSupported(1)}
}
```

5.7.2 Transport Command

One generic APDU is used on the interfaces ES10a, ES10b and ES10c to transport all command request and command response data.

Command Message

All functions use the command message STORE DATA as defined in GlobalPlatform Card Specification [8] with the specific coding defined below.

Code	Value	Meaning
CLA	'80'-'83' or 'C0'-'CF'	See GlobalPlatform Card Specification [8] section 11.1.4
INS	'E2'	STORE DATA
P1	'11' or '91'	See below
P2	'xx'	Block number
Lc	Var.	Length of data field
Data	'xx xx…'	The data field SHALL be one of the data object command DER encoded defined in ES10x

Code	Value	Meaning
Le	'00'	

Table 47: ES10x STORE DATA command APDU

Parameter P1

The P1 SHALL be coded as defined in the following table.

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0/1	-	-	-	-	-	-	-	More blocks/Last block
-	0	0	-	-	-	-	-	No general encryption information or non-encrypted data
-	-	-	1	0	-	-	-	BER-TLV format of the command data field
-	-	-	-	-	-	-	1	Case 4 command as defined in GlobalPlatform AmdA [9]
-	-	-	-	-	X	X	-	RFU

Table 48: ES10x STORE DATA P1

This interface is defined with command functions that are mostly handled with a single APDU command and response pair. When multiple STORE DATA commands are required, it is indicated by the use of the 'more commands' bit in the P1 byte as defined in GlobalPlatform Card Specification [8], and procedure bytes controlling the return of additional data (e.g. '61 XX'). In particular if the size of the response is bigger than 256 bytes, the chaining of the commands SHALL be done as defined in ISO/IEC 7816-4 [14]. The responses SHALL be retrieved by the Device using several GET RESPONSE commands.

Data Field

The command data field contains the command request data for each function.

Response Message

Data Field

The response data field contains the command response data for each function.

Processing State Returned in the Response Message

eUICC SHALL indicate an APDU header coding error as defined in GlobalPlatform Card Specification [8] section 11.11.3.2.

A successful execution of the APDU command SHALL be indicated by the status bytes '90 00' if no proactive command is pending and by '91 XX' if a proactive command (e.g., REFRESH) is pending. All function specific errors SHALL be indicated in the response data field.

An incorrect/invalid data field encoding (i.e. not a DER data object) SHALL be indicated by status bytes '6A 80' (Incorrect values in command data).

An unsupported or unknown command request in the data field SHALL be indicated by status bytes '6A 88' (Reference data not found).

While a Profile state change is ongoing, i.e. a command was sent to the eUICC which mandates a REFRESH, but the REFRESH was not yet successfully executed (i.e. no Terminal Response with result "command performed successfully" received or reset of the eUICC), the eUICC MAY reject any other ES10 command with the status word '69 85' (Conditions of use not satisfied).

5.7.3 Function (ES10a): GetEuiccConfiguredAddresses

Related Procedures: SM-DS – Event Retrieval

Function Provider Entity: ISD-R (LPA Services)

Description:

This function retrieves the Root SM-DS and if configured the Default SM-DP+ address from the eUICC. Both addresses are coded as FQDN.

Command Data

The command data SHALL be coded as follows:

```
EuiccConfiguredAddressesRequest ::= [60] SEQUENCE {  -- Tag 'BF3C'
}
```

Response Data

The response data SHALL be coded as follows:

```
EuiccConfiguredAddressesResponse ::= [60] SEQUENCE {  -- Tag 'BF3C'
   defaultDpAddress UTF8String OPTIONAL,  -- Default SM-DP+ address as an FQDN
   rootDsAddress UTF8String  -- Root SM-DS address as an FQDN
}
```

5.7.4 Function (ES10a): SetDefaultDpAddress

Related Procedures: Set/Edit Default SM-DP+ Address

Function Provider Entity: ISD-R (LPA Services)

Description:

This function is used to update the default SM-DP+ address.

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:

```
SetDefaultDpAddressRequest ::= [63] SEQUENCE { -- Tag 'BF3F'
   defaultDpAddress UTF8String -- Default SM-DP+ address as an FQDN
}
```

Response Data

The response data SHALL be coded as follows:

```
SetDefaultDpAddressResponse ::= [63] SEQUENCE { -- Tag 'BF3F'
   setDefaultDpAddressResult INTEGER { ok (0), undefinedError (127)}
}
```

5.7.5 Function (ES10b): PrepareDownload

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R (LPA Services)

Description:

This function initiates a Bound Profile Package download after a successful authentication of an SM-DP+.

On reception of this command, the eUICC SHALL:

- Verify that the received transactionId contained in the smdpSigned2 matches the one of the ongoing RSP session.
- Verify that the SM-DP+ has been previously authenticated and a CERT.DPauth.ECDSA is attached to the ongoing RSP session.
- Verify the validity of the CERT.DPpb.ECDSA (using the ECASD service) as defined in section 4.5.2.2.
- Verify the signature (smdpSignature2) of the SM-DP+ done upon smdpSigned2 data structure as described hereunder.
- Verify that CERT.DPauth.ECDSA and CERT.DPpb.ECDSA belong to the same entity (i.e. same OID in subjectAltName).
- Verify that CERT.DPauth.ECDSA and CERT.DPpb.ECDSA trust chain lead to the same GSMA CI certificate.

If any of these verifications fail, the eUICC SHALL return an error.

If these verifications are successful, the eUICC SHALL:

- Extract the public key of the CERT.DPpb.ECDSA and attach it to the RSP session context.
- If bppEuiccOtpk is provided in smdpSigned2 and it corresponds to a stored one-time key pair otPK.EUICC.ECKA/ otSK.EUICC.ECKA for this SM-DP+: use this key pair for the RSP session. Otherwise: generate a new one-time key pair otSK.EUICC.ECKA and otPK.EUICC.ECKA using the curve indicated by the Key Parameter Reference Value of CERT.DPpb.ECDSA, and attach otSK.EUICC.ECKA to the RSP session context.
- Generate euiccSigned2 data object as defined hereunder.

- Generate the euiccSignature2, as defined hereunder, with the SK.EUICC.ECDSA related to the CERT.EUICC.ECDSA as requested by SM-DP+.

Command Data

The command data SHALL be coded as follows.

```
PrepareDownloadRequest ::= [33] SEQUENCE { -- Tag 'BF21'
   smdpSigned2 SmdpSigned2,                    -- Signed information
   smdpSignature2 [APPLICATION 55] OCTET STRING,      -- tag '5F37'
   hashCc Octet32 OPTIONAL, -- Hash of confirmation code
   smdpCertificate Certificate    -- CERT.DPpb.ECDSA
}

SmdpSigned2 ::= SEQUENCE {
   transactionId [0] TransactionId,      -- The TransactionID generated by the
SM-DP+
   ccRequiredFlag BOOLEAN, --Indicates if the Confirmation Code is required
   bppEuiccOtpk [APPLICATION 73] OCTET STRING OPTIONAL      -- otPK.EUICC.ECKA
already used for binding the BPP, tag '5F49'
}
```

smdpSignature2 SHALL be created using the SK.DPpb.ECDSA and verified by the eUICC using the PK.DPpb.ECDSA as described in section 2.6.7.2. smdpSignature2 SHALL apply on the concatenated data object smdpSigned2 and euiccSignature1.

Response Data

The response data SHALL be coded as follows.

```
PrepareDownloadResponse ::= [33] CHOICE { -- Tag 'BF21'
   downloadResponseOk PrepareDownloadResponseOk,
   downloadResponseError PrepareDownloadResponseError
}

PrepareDownloadResponseOk ::= SEQUENCE {
   euiccSigned2 EUICCSigned2,           -- Signed information
   euiccSignature2 [APPLICATION 55] OCTET STRING      -- tag '5F37'
}

EUICCSigned2 ::= SEQUENCE {
   transactionId [0] TransactionId,
   euiccOtpk [APPLICATION 73] OCTET STRING,           -- otPK.EUICC.ECKA, tag '5F49'
   hashCc Octet32 OPTIONAL        -- Hash of confirmation code
}

PrepareDownloadResponseError ::= SEQUENCE {
   transactionId [0] TransactionId,
   downloadErrorCode DownloadErrorCode
}

DownloadErrorCode ::= INTEGER {invalidCertificate(1), invalidSignature(2),
unsupportedCurve(3), noSessionContext(4), invalidTransactionId(5),
undefinedError(127)}
```

euiccSignature2 SHALL be created using the SK.EUICC.ECDSA and verified using the PK.EUICC.ECDSA as described in section 2.6.7.2. euiccSignature2 SHALL apply on the concatenated data objects euiccSigned2 and smdpSignature2.

5.7.6 Function (ES10b): LoadBoundProfilePackage

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R (LPA Services)

Description:

This function transfers a Bound Profile Package to the eUICC. The transfer is done by calling repeatedly this function with blocks of 255 bytes or less according to the structure of the Bound Profile Package, i.e. each TLV of the BPP that is up to 255 bytes is transported in one APDU. Larger TLVs are sent in blocks of 255 bytes for the first blocks and a last block that MAY be shorter.

The eUICC SHALL erase the otSK.EUICC.ECKA attached to this RSP session no later than the successful completion of the BPP installation.

If this function is called when there is no RSP session, or if it is called with a TLV that is not expected according to the structure of the Bound Profile Package, the eUICC SHALL return the status words '6A 88' (Reference data not found) or '69 85' (Conditions of use not satisfied) as the response of the transport command defined in section 5.7.2.

Command Data

The command data SHALL contain a block of data of the BPP. The transfer and slicing in blocks of data SHALL follow description given in section 2.5.5.

Response Data

The data presence in the response message depends on the block status:

- For an intermediate block of data of a BPP TLV, the response message SHALL not contain data field.
- For the last block of data of a BPP TLV, a response message containing a Profile Installation Result SHALL be present or absent as specified in section 2.5.6.

5.7.7 Function (ES10b): GetEUICCChallenge

Related Procedures: Common Mutual Authentication

Function Provider Entity: ISD-R (LPA Services)

Description:

This function initiates a RSP session between an RSP Server and the ISD-R. The initiation of the RSP session is materialized on the eUICC by the creation of a context containing an eUICC challenge.

Only one RSP session can be managed by the ISD-R at a time. So an on-going RSP session SHALL be completed before requesting the opening of a new one.

On reception of this function, the eUICC SHALL:

- Determine if a previous session was not completed. If so, then:
  - The eUICC MAY store the unused otPK.eUICC.ECKA and otSK.eUICC.ECKA, together with the SM-DP+ OID, for future retry.
  - The eUICC SHALL discard the previous session context.

- Create a new session context and generate a new random challenge attached to this RSP session.

Command Data

The command data SHALL be coded as follows:

```
GetEuiccChallengeRequest ::= [46] SEQUENCE { -- Tag 'BF2E'
}
```

Response Data

The response data SHALL be coded as follows:

```
GetEuiccChallengeResponse ::= [46] SEQUENCE { -- Tag 'BF2E'
   euiccChallenge Octet16  -- random eUICC challenge
}
```

5.7.8 Function (ES10b): GetEUICCInfo

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R (LPA Services)

Description:

This function gets the eUICC Information as defined in section 0. This function MAY be called at any time.

Command Data

The command data SHALL be coded as follows to retrieve EUICCInfo1:

```
GetEuiccInfo1Request ::= [32] SEQUENCE { -- Tag 'BF20'
}
```

The command data SHALL be coded as follows to retrieve EUICCInfo2:

```
GetEuiccInfo2Request ::= [34] SEQUENCE { -- Tag 'BF22'
}
```

Response Data

The response data SHALL be coded as follows:

```
EUICCInfo1 ::= [32] SEQUENCE { -- Tag 'BF20'
   svn [2] VersionType,    -- GSMA SGP.22 version supported (SVN)
   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
}

EUICCInfo2 ::= [34] SEQUENCE { -- Tag 'BF22'
   profileVersion [1] VersionType,      -- SIMAlliance Profile package version
supported
   svn [2] VersionType,    -- GSMA SGP.22 version supported (SVN)
```

```
   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
   sasAcreditationNumber UTF8String (SIZE(0..64)),
   certificationDataObject [12] CertificationDataObject OPTIONAL
}

-- Definition of RspCapability
RspCapability ::= BIT STRING {
   additionalProfile(0), -- at least one more Profile can be installed
   crlSupport(1), -- CRL
   rpmSupport(2), -- Remote Profile Management
   testProfileSupport (3) -- support for test profile
}

-- Definition of CertificationDataObject
CertificationDataObject ::= SEQUENCE {
   platformLabel UTF8String,      -- Platform_Label as defined in GlobalPlatform
DLOA specification [57]
   discoveryBaseURL UTF8String    -- Discovery Base URL of the SE default DLOA
Registrar as defined in GlobalPlatform DLOA specification [57]
}
```

The "number of installed application" value field of `extCardResource` SHALL be set to '00'.

The `ts102241Version` field indicates the latest version of ETSI TS 102 241 [53] supported by the eUICC. This field SHALL not be present if the eUICC doesn't support Java card™.

The `globalplatformVersion` field indicates the latest version of GlobalPlatform Card Specification [8] supported by the eUICC. This field SHALL be present if the supported version differs from the one required in this specification.

Elements in `euiccCiPKIdListForVerification` and `euiccCiPKIdListForSigning` SHALL be set in decreasing order of priority by the eUICC, where the first element in the list is the most preferred and the last element in the list is the least preferred.

The `forbiddenProfilePolicyRules` data object SHALL contain the list of PPRs that are 'forbidden' to be set in any Profile (the `PprIds` type is defined in section 2.8.1.1). A PPR is 'forbidden' when there is no PPAR related to this PPR. In addition, PPR1 is 'forbidden' if an Operational Profile is currently installed on the eUICC.

The information contained in `forbiddenProfilePolicyRules` data object SHALL be used during the eligibility check performed by the SM-DP+: the SM-DP+ SHALL not deliver a Profile containing a PPR 'forbidden' by the eUICC.

The ppVersion data object indicates the version of the GSMA eUICC Protection Profile for RSP against which the eUICC has been certified. During the interim period until the product can be certified against the GSMA eUICC Protection Profile, the ppVersion SHALL be set to V0.X.Y.

The sasAcreditationNumber data object indicates the SAS for RSP accreditation number obtained by the EUM.

During the interim period until the EUM can be certified against the SAS for RSP, the sasAcreditationNumber SHALL contain the accreditation number obtained by the EUM for production of UICC (called SAS-UP).

The additionalProfile bit SHALL be set to '1' to indicate that at least one more Profile can be installed. Otherwise it SHALL be set to '0'.

The crlSupport bit SHALL be set to '1' to indicate that the eUICC supports the optional CRL management feature (section 4.6). Otherwise it SHALL be set to '0'.

The rpmSupport bit is reserved for future use and SHALL be set to '0'.

The testProfileSupport bit SHALL be set to '1' to indicate that the eUICC supports the optional Test Profile feature. Otherwise it SHALL be set to '0'.

5.7.9 Function: (ES10b): ListNotification

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used by the LPA to list all available pending notifications from an eUICC before retrieving a specific notification.

Command Data

The command data SHALL be coded as follows:

```
ListNotificationRequest ::= [40] SEQUENCE { -- Tag 'BF28'
   profileManagementOperation [1] NotificationEvent OPTIONAL
}
```

The profileManagementOperation data object can be used to filter the list of notifications that the eUICC SHALL return. 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.

If profileManagementOperation data object is omitted, the eUICC SHALL return all stored notifications whatever their type.

If `profileManagementOperation` data object indicates no event (all bits set to 0), the eUICC SHALL return an empty list or `undefinedError(127)`.

Response Data

The response data SHALL contain the 'List Notification Response' data object if available, and filtered according to `profileManagementOperation` data object received in the command data. The eUICC MAY provide the notifications in any order. The list SHALL be empty if there are no pending notification matching the filtering criteria.

```
ListNotificationResponse ::= [40] CHOICE { -- Tag 'BF28'
   notificationMetadataList SEQUENCE OF NotificationMetadata,
   listNotificationsResultError INTEGER {undefinedError(127)}
}

NotificationMetadata ::= [47] SEQUENCE { -- Tag 'BF2F'
   seqNumber [0] INTEGER,
   profileManagementOperation [1] NotificationEvent, /*Only one bit SHALL be set to
1*/
   notificationAddress UTF8String, -- FQDN to forward the notification
   iccid Iccid OPTIONAL
}
```

5.7.10 Function (ES10b): RetrieveNotificationsList

Related Procedures: Notifications

Function Provider Entity: ISD-R (LPA Services)

Description:

This function retrieves the list of Pending notifications for installed Profiles including their confirmation required and the related data.

Command Data

The command data SHALL be coded as follows:

```
RetrieveNotificationsListRequest ::= [43] SEQUENCE { -- Tag 'BF2B'
   searchCriteria CHOICE {
      seqNumber [0] INTEGER,
      profileManagementOperation [1] NotificationEvent
   } OPTIONAL
}
```

The `searchCriteria` data object can be used to filter the list of notifications that the eUICC SHALL return, filtering can be done on sequence number or notification type. 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`.

If `searchCriteria` data object is omitted, the eUICC SHALL return all stored Notifications.

Response Data

The response data SHALL contain the list of PendingNotification data objects. The list SHALL be filtered according to the notification seqNumber or indicated operation type that generates notifications provided in the command data. The eUICC MAY provide the notifications in any order. The list SHALL be empty if there are no pending notifications matching the filtering criteria.

The following is the definition of the RetrieveNotificationsListResponse data object

```
RetrieveNotificationsListResponse ::= [43] CHOICE { -- Tag 'BF2B'
   notificationList SEQUENCE OF PendingNotification,
   notificationsListResultError INTEGER { undefinedError(127)}
}

PendingNotification ::= CHOICE {
   profileInstallationResult [55] ProfileInstallationResult, -- tag 'BF37'
   otherSignedNotification OtherSignedNotification
}

OtherSignedNotification ::= SEQUENCE {
   tbsOtherNotification NotificationMetadata,
   euiccNotificationSignature [APPLICATION 55] OCTET STRING, -- eUICC signature of
tbsOtherNotification, 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
}
```

euiccNotificationSignature SHALL be created using the SK.EUICC.ECDSA and verified using the PK.EUICC.ECDSA as described in section 2.6.7.2. euiccNotificationSignature SHALL apply on the tbsOtherNotification data object.

When generating the euiccNotificationSignature, the eUICC SHALL use credentials related to the euiccCiPKIdToBeUsed parameter received from the SM-DP+ during the Profile Download and Installation Procedure.

5.7.11 Function (ES10b): RemoveNotificationFromList

Related Procedures: Notifications

Function Provider Entity: ISD-R (LPA Services)

Description:

This function informs the eUICC that:

1. A specific Notification has been sent to the recipient address; and
2. The eUICC SHALL remove such Notification from the Pending Notifications List.

Command Data

The sent Notification TLV SHALL be the DER encoding of the NotificationSent defined as follows:

```
NotificationSentRequest ::= [48] SEQUENCE { -- Tag 'BF30'
   seqNumber [0] INTEGER
```

```
}
```

Response Data

The response data SHALL be coded as follows:

```
NotificationSentResponse ::= [48] SEQUENCE { -- Tag 'BF30'
   deleteNotificationStatus INTEGER {ok(0), nothingToDelete(1),
undefinedError(127)}
}
```

5.7.12 Function (ES10b): LoadCRL

Related Procedures: None

Function Provider Entity: ISD-R (LPA Services)

Description:

This function is used to transfer a CRL-A to the eUICC. It is optionally supported by the Device and the eUICC, as indicated by their corresponding capabilities.

On reception of this command the eUICC SHALL:

- Verify the CRL signature using the PK.CI.ECDSA identified by the extension field 'Authority Key Identifier'; if the key is unknown or if signature is invalid, the command execution SHALL be stopped with error code `verificationKeyNotFound` or `invalidSignature`.
- Verify CRL format is valid and is a CRL-A. If not, the command execution SHALL be stopped with error code `invalidCRLFormat`.
- Compare the 'CRL Number' field value contained in the provided CRL-A with that of the last successfully processed CRL-A on the eUICC:
  - If the 'CRL Number' field value is lower: stop the command execution and return a response with error code `fresherCrlAlreadyLoaded`.
  - If the 'CRL Number' field value is equal:
    - If the CRL-A is a complete CRL (i.e. the Extension for Total Partial CRL Number and the Extension for Partial CRL Number are missing): stop the command execution and return a response `loadCRLResponseOk`, with an empty missing part indication.
    - If the CRL-A is a partial CRL and this part has already been loaded, stop the command execution and return a response `loadCRLResponseOk` with missing parts indication.
- If the provided CRL-A is a Delta CRL (as defined in RFC 5280 [17]): verify that the CRL-A identified by the 'Base CRL Number' field has already been processed. If not, the command execution SHALL be stopped with error code `baseCrlMissing`.

If these verifications are successful, the eUICC SHALL:

1. Update its internal time reference with the value indicated in the field 'thisUpdate' and apply the process described in section 4.6.3.
2. Remove Certificate entries from its internal storage for each revoked Certificate that has expired.
4. If the CRL-A is not empty, add a new Certificate entry in its internal storage for each newly revoked Certificate where an entry comprises the Certificate serial number and expiration date.
5. Store the new CRL number with the value indicated in the extension field 'CRL Number'. Optionally, update the list of processed part numbers of this CRL-A.

The eUICC SHALL ensure the atomicity of steps 3 and 4, either both steps SHALL be performed or none of them. In case of error during one of the two steps, the eUICC SHALL not add any new Certificate entry in its internal storage, leave the stored CRL number unchanged, leave the list of processed part numbers unchanged and return the error code `notEnoughMemorySpace`, if relevant, or `undefinedError`.

The eUICC MAY apply the changes provided by each CRL-A part before it has received all parts.

Command Data

The command data SHALL be coded as follows.

The 'crl' field SHALL contain a complete CRL-A, or a partial CRL-A as described in section 4.6.1.

```
-- Definition of data structure command for loading a CRL
LoadCRLRequest ::= [53] SEQUENCE {  -- Tag 'BF35'
   -- A CRL-A
   crl CertificateList
}
```

Response Data

The response data SHALL be coded as follows:

```
-- Definition of data structure response for loading a CRL
LoadCRLResponse ::= [53] CHOICE {  -- Tag 'BF35'
loadCRLResponseOk LoadCRLResponseOk,
loadCRLResponseError LoadCRLResponseError
}

LoadCRLResponseOk ::= SEQUENCE {
   missingParts SEQUENCE OF INTEGER OPTIONAL
}
LoadCRLResponseError ::= INTEGER {invalidSignature(1), invalidCRLFormat(2),
notEnoughMemorySpace(3), verificationKeyNotFound(4), fresherCrlAlreadyLoaded(5),
baseCrlMissing(6), undefinedError(127)}
```

The optional field '`missingParts`' SHALL be present when the provided CRL-A is a partial CRL and not all the parts have been processed. In that case the '`missingParts`' field SHALL contain the list of all numbers (starting from 1 and up to the value contained in the certificate extension 'Total Partial CRL Number') identifying parts that have not been already processed.

Example:

The provided CRL is a valid (no error condition) part of a CRL-A split in 3 parts. The certificate extension 'Total Partial CRL Number' contains the value '3', and the certificate extension 'Partial CRL Number' contains the value '2'. And no parts have been previously processed.

In that case, the eUICC SHALL process the part with success and return a '`missingParts`' data object containing the values '1' and '3'.

5.7.13 Function (ES10b): AuthenticateServer

Related Procedures: Common Mutual Authentication

Function Provider Entity: ISD-R (LPA Services)

Description:

This function performs the authentication of the RSP Server by the eUICC.

On reception of this command, the eUICC SHALL:

- Verify that a RSP session context exists (i.e. "ES10b.GetUICCChallenge" function has been previously called).
- Verify the validity of the RSP Server Certificate for authentication (using the ECASD service), as described in section 4.5.2.2 using the PK.CI.ECDSA identified by the Authority Key Identifier contained in the RSP Server Certificate.
- Verify that the RSP Server Certificate is either CERT.DPauth.ECDSA or CERT.DSauth.ECDSA, as described in section 4.5.2.2.
- Verify the signature (serverSignature1) of the RSP Server done upon serverSigned1 as described hereunder.
- Verify that the euiccChallenge attached to the ongoing RSP session matches the euiccChallenge returned by the RSP Server inside the serverSigned1.
- Verify that euiccCiPKIdToBeUsed is supported and related credentials are available for signing.

If any of these verifications fail, the eUICC SHALL return an `authenticateResponseError` with the relevant error code.

Otherwise the eUICC SHALL:

- Attach the received `transactionId` in the RSP session context.
- Attach the received RSP Server Certificate in the RSP session context.
- Generate `euiccSigned1` data object as defined hereunder.
- Generate the `euiccSignature1` as defined hereunder, with the SK.EUICC.ECDSA related to the CERT.EUICC.ECDSA as requested by RSP Server.

Command Data

The command data SHALL be coded as follows.

```
AuthenticateServerRequest ::= [56] SEQUENCE { -- Tag 'BF38'
```

```
   serverSigned1 ServerSigned1,                        -- Signed information
   serverSignature1 [APPLICATION 55] OCTET STRING,     -- tag '5F37'
   euiccCiPKIdToBeUsed SubjectKeyIdentifier,           -- CI Public Key Identifier to
be used
   serverCertificate Certificate, -- RSP Server Certificate CERT.XXauth.ECDSA
   ctxParams1 CtxParams1
}

ServerSigned1 ::= SEQUENCE {
   transactionId [0] TransactionId,            -- The Transaction ID generated by
the RSP Server
   euiccChallenge [1] Octet16,           -- The eUICC Challenge
   serverAddress [3] UTF8String,  -- The RSP Server address as an FQDN
   serverChallenge [4] Octet16           -- The RSP Server Challenge
}

CtxParams1 ::= CHOICE {
   ctxParamsForCommonAuthentication CtxParamsForCommonAuthentication-- New
contextual data objects MAY be defined for extensibility.
}

CtxParamsForCommonAuthentication ::= SEQUENCE {
   matchingId UTF8String OPTIONAL, -- The MatchingId could be the Activation code
token or EventID or empty
   deviceInfo DeviceInfo -- The Device information
}
```

`serverSignature1` SHALL be created using the private key associated to the RSP Server Certificate for authentication, and verified by the eUICC using the contained public key as described in section 2.6.7.2. `serverSignature1` SHALL apply on `serverSigned1` data object.

Response Data

The response data SHALL be coded as follows.

```
AuthenticateServerResponse ::= [56] CHOICE { -- Tag 'BF38'
   authenticateResponseOk AuthenticateResponseOk,
   authenticateResponseError AuthenticateResponseError
}

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
}

EuiccSigned1 ::= SEQUENCE {
   transactionId [0] TransactionId,
   serverAddress [3] UTF8String, -- The RSP Server address as an FQDN
   serverChallenge [4] Octet16,   -- The RSP Server Challenge
   euiccInfo2 [34] EUICCInfo2,
   ctxParams1 CtxParams1
}

AuthenticateResponseError ::= SEQUENCE {
   transactionId [0] TransactionId,
   authenticateErrorCode AuthenticateErrorCode
}
```

```
AuthenticateErrorCode ::= INTEGER {invalidCertificate(1), invalidSignature(2),
unsupportedCurve(3), noSessionContext(4), invalidOid(5), euiccChallengeMismatch(6),
ciPKUnknown(7), undefinedError(127)}
```

`euiccSignature1` SHALL be created using the SK.EUICC.ECDSA and verified using the PK.EUICC.ECDSA as described in in section 2.6.7.2. `euiccSignature1` SHALL apply on `euiccSigned1` data object.

5.7.14 Function (ES10b): CancelSession

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R (LPA Services)

Description:

This function allows to cancel an on-going RSP session on the eUICC and to get a signed data structure that is necessary for the LPAd to request the same session cancellation to the RSP Server.

On reception of this command the eUICC SHALL:

- Verify that the Transaction ID provided as input data is known and matches the one retained in the context of the ongoing RSP session. If not the eUICC SHALL return an error code `invalidTransactionId`.

If these verifications are successful, the eUICC SHALL:

- Generate an `euiccCancelSessionSigned` data object as defined hereunder.
- Generate the `euiccCancelSessionSignature`, as defined hereunder, with the SK.EUICC.ECDSA.
- If the CancelSessionReason is '`postponed`' or '`timeout`', the eUICC MAY store the unused otPK.eUICC.ECKA and otSK.eUICC.ECKA, together with the SM-DP+ identity, for future retry.
- Discard the on-going RSP session context.

The eUICC MAY perform additional internal operations, which are out of scope of this specification.

Command Data

The command data SHALL be coded as follows.

```
CancelSessionRequest ::= [65] SEQUENCE { -- Tag 'BF41'
   transactionId TransactionId,    -- The TransactionID generated by the RSP Server
   reason CancelSessionReason
}

CancelSessionReason ::= INTEGER {endUserRejection(0), postponed(1), timeout(2),
pprNotAllowed(3), metadataMismatch(4), loadBppExecutionError(5),
undefinedReason(127)}
```

The cancel session reasons `endUserRejection(0)`, `pprNotAllowed(3)`, `metadataMismatch(4)`, `loadBppExecutionError(5)`, `undefinedReason(127)` terminate the download order.

Response Data

The response data SHALL be coded as follows.

```
CancelSessionResponse ::= [65] CHOICE { -- Tag 'BF41'
   cancelSessionResponseOk CancelSessionResponseOk,
   cancelSessionResponseError INTEGER {invalidTransactionId(5),
undefinedError(127)}
}

CancelSessionResponseOk ::= SEQUENCE {
   euiccCancelSessionSigned EuiccCancelSessionSigned,       -- Signed information
   euiccCancelSessionSignature [APPLICATION 55] OCTET STRING -- tag '5F37'
}

EuiccCancelSessionSigned ::= SEQUENCE {
   transactionId TransactionId,
   smdpOid OBJECT IDENTIFIER, -- SM-DP+ OID as contained in CERT.DPauth.ECDSA
   reason CancelSessionReason
}
```

`euiccCancelSessionSignature` SHALL be created using the SK.EUICC.ECDSA and verified using the PK.EUICC.ECDSA as described in section 2.6.7.2. `euiccCancelSessionSignature` SHALL apply on the data object `euiccCancelSessionSigned`.

5.7.15 Function (ES10c): GetProfilesInfo

Related Procedures: Local Profile Management – List Profiles

Function Provider Entity: ISD-R (LPA Services)

Description:

This function retrieves the list of Profile information for installed Profiles including their current state (Enabled/Disabled) and their associated Profile Metadata. This function MAY also be used to retrieve this information for a particular Profile.

If there is a Profile state change ongoing (i.e. during an enable, disable or eUICCMemoryReset command requiring a REFRESH) and the Profile state is requested, the eUICC SHALL terminate the command with the status word '69 85' (Conditions of use not satisfied).

Command Data

The command data consists of the search criteria and the tag list. It SHALL be coded as follows:

```
ProfileInfoListRequest ::= [45] SEQUENCE { -- Tag 'BF2D'
   searchCriteria [0] CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID of the ISD-P, tag '4F'
      iccid Iccid, -- ICCID, tag '5A'
      profileClass [21] ProfileClass -- Tag '95'
```

```
   } OPTIONAL,
   tagList [APPLICATION 28] OCTET STRING OPTIONAL -- tag '5C'
}
```

It SHALL be possible to search for all the Profiles using no search criterion.

The value field of the tag list (tag ‘5C’) contains a concatenation of tags (without delimitation) indicating the data objects to include in the response for each Profile matching all given search criteria.

If a requested data object is not present for a matching Profile, the data object SHALL simply be omitted in the response for that Profile.

The eUICC SHALL support the following tags in the tag list:

- ICCID, tag '5A' (*)
- ISD-P AID, tag '4F' (*)
- Profile state, tag '9F70' (*)
- Profile Nickname, tag '90' (*)
- Service provider name, tag '91' (*)
- Profile name, tag '92' (*)
- Icon type, tag '93' (*)
- Icon, tag '94' (*)
- Profile Class, tag '95' (*)
- Notification Configuration Info, tag 'B6'
- Profile Owner, tag 'B7'
- SM-DP+ proprietary data, tag 'B8'
- Profile Policy Rules, tag '99'

If no tag list is present, the eUICC SHALL return the default ProfileInfo: the ProfileInfo data objects marked with (*) for each Profile matching the selection criterion.

If a tag list is present, the eUICC SHALL return all data objects requested in the tag list for each Profile matching the selection criterion.

Example of use:

- Retrieve the ProfileInfo for all installed Profiles. The command data field SHALL be coded as 'BF2D 00'.
- Retrieve all information of a particular Profile/ISD-P having the following AID: A0 00 00 05 59 10 10 FF FF FF FF 89 00 00 10 00. The command data field SHALL be coded as 'BF2D 24 A0 12 4F 10 A0 00 00 05 59 10 10 FF FF FF FF 89 00 00 10 00 5C 0E 5A 4F 9F70 90 91 92 93 94 95 B6 B7 B8 99'.
- Retrieve ICCID and Profile state for all installed Profiles. The command data field SHALL be coded as 'BF 2D 05 5C 03 5A 9F70'.

Response Data

The response data SHALL contain the list of data objects as required by the tag list. The list SHALL be empty if no Profile is installed or if no Profile matches the given search criteria.

The following is the definition of the ProfileInfoListResponse data object:

```
-- Definition of ProfileInfoListResponse
ProfileInfoListResponse ::= [45] CHOICE { -- Tag 'BF2D'
   profileInfoListOk SEQUENCE OF ProfileInfo,
   profileInfoListError ProfileInfoListError
}

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
ES10c:GetProfilesInfo
   profileClass [21] ProfileClass DEFAULT operational, -- 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'}

IconType ::= INTEGER {jpg(0), png(1)}
ProfileState ::= INTEGER {disabled(0), enabled(1)}
ProfileClass ::= INTEGER {test(0), provisioning(1), operational(2)}
ProfileInfoListError ::= INTEGER {incorrectInputValues(1), undefinedError(127)}
```

The `profileOwner` data object can only be returned if Profile Owner has been provided in Profile Metadata or if files EF_{IMSI}, EF_{GID1} or EF_{GID2} are not PIN protected.

The `profilePolicyRules` data object SHALL contain the identifiers of all Profile Policy Rules of the Profile.

5.7.16 Function (ES10c): EnableProfile

Related Procedures: Local Profile Management – Enable Profile

Function Provider Entity: LPA Services

Description:

This function is used to enable a Profile on the eUICC. The function makes the target Profile enabled, and disables implicitly the currently Enabled Profile, if any. This SHALL be performed in an atomic way, meaning that in case of any error during the command execution, the command SHALL stop and SHALL leave the involved Profiles in their original states prior to command execution.

NOTE: Before calling the EnableProfile function with the refreshFlag not being set, the Device has the responsibility to ensure that the relevant conditions for use are met, as indicated in section 3.2.1.

Upon reception of the EnableProfile function, if the refreshFlag is not set, the eUICC SHALL:

- Check if there is a proactive session ongoing (which the Device did not terminate). If so, the eUICC SHALL terminate the EnableProfile command with error "catBusy".
- 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.

Regardless of the value of refreshFlag, the eUICC SHALL:

- Verify that the target Profile is in the Disabled state.
- If the target Profile is not a Test Profile: check if the Profile Policy Rules of the currently Enabled Profile allow its disabling.
- If the currently Enabled Profile is a Test Profile: verify
    - that the target Profile is (another) Test Profile, or
    - if an Operational Profile was in Enabled state before the (first) Test Profile was enabled, that the target Profile is this Operational Profile.
- If any of these verifications fail, terminate the command with an error.

If the refreshFlag is not set:

- the eUICC SHALL reset the PIN status
- the eUICC SHALL disable the currently Enabled Profile (if any), Enable the target Profile and implicitly select the MF on the basic logical channel.
- The eUICC SHALL return OK to the LPAd

NOTE: for subsequent actions by the Device see section 3.2.1.

If the refreshFlag is set:

- If there is a proactive session ongoing, the eUICC SHALL terminate the EnableProfile command with error "catBusy".

NOTE: the Device MAY take implementation-dependent action to terminate the proactive command session, and MAY send the enable command again without any further End User interaction.

- Otherwise
    - The eUICC SHALL mark the currently enabled Profile (if any) as "to be disabled", mark the target Profile as "to be enabled" and return OK to the LPAd
    - 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 [31].
    - Upon reception of the Terminal Response with result "command performed successfully" or upon reset of the eUICC, the eUICC SHALL disable the currently Enabled Profile (if any) and Enable the target 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 target Profile. Subsequent actions are implementation specific.

NOTE: In the case the Device supports the Single Wire Protocol interface to the eUICC, the Device SHOULD take the appropriate actions regarding this interface when Profile state is changed.

The eUICC MAY start a new proactive UICC session only after a new TERMINAL PROFILE command is executed.

- If a previously 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:

```
EnableProfileRequest ::= [49] SEQUENCE { -- Tag 'BF31'
   profileIdentifier CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
      iccid Iccid -- ICCID, tag '5A'
   },
   refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
```

Response Data

The response data SHALL be coded as follows:

```
EnableProfileResponse ::= [49] SEQUENCE { -- Tag 'BF31'
   enableResult INTEGER {ok(0), iccidOrAidNotFound (1),
profileNotInDisabledState(2), disallowedByPolicy(3), wrongProfileReenabling(4),
catBusy(5), undefinedError(127)}
}
```

Alternative Case 3 Command

In addition to the command data exchange described above, the eUICC SHALL support the following alternative command for the function:

The command data is sent to the eUICC in a Case 3 STORE DATA command:

- The STORE DATA APDU SHALL be coded as defined in section 5.7.2, with the exception of P1 which SHALL be set to '90'.
- The command data SHALL be coded as defined above, the response data SHALL not be present.

- The following additional status bytes are defined:
  '6A 82': Profile not found
  '69 85': Profile not in disabled state or command not allowed by Profile Policy Rules or re-enabling wrong Profile
  '93 00': CAT is busy. Command cannot be executed at present, further normal commands are allowed

5.7.17 Function (ES10c): DisableProfile

Related Procedures: Local Profile Management – Disable Profile

Function Provider Entity: LPA Services

Description:

This function is used to disable a Profile on the eUICC.

NOTE: Before calling the DisableProfile function with the refreshFlag not being set, the Device has the responsibility to ensure that the relevant conditions for use are met, as indicated in section 3.2.2.

Upon reception of the DisableProfile function, if the refreshFlag is not set, the eUICC SHALL:

- Check if there is a proactive session ongoing (which the Device did not terminate). If so, the eUICC SHALL terminate the DisableProfile function with error "catBusy".
- 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.

Regardless of the value of refreshFlag, the eUICC SHALL:

- Verify that the target Profile is in the Enabled state.
- If the currently Enabled Profile is not a Test Profile: check if the Profile Policy Rules of the target Profile allows its disabling.
- If any of these verifications fail, terminate the command with an error.
- If the refreshFlag is not set:
  - the eUICC SHALL reset the PIN status.
  - the eUICC SHALL disable the currently Enabled Profile.
  - If the target Profile is a Test Profile and an Operational Profile was in Enabled state before the (first) Test Profile was enabled: enable this Operational Profile. If this Operational Profile was deleted while the Test Profile was enabled, no error SHALL be generated and the function execution SHALL continue.
  - The eUICC SHALL implicitly select the MF on the basic logical channel.
  - The eUICC SHALL return OK to the LPAd.

NOTE: for subsequent actions by the Device see section 3.2.2.

- If the refreshFlag is set:
  - If there is a proactive session ongoing, the eUICC SHALL terminate the DisableProfile function with error "catBusy"

NOTE: the Device MAY take implementation-dependent actions to terminate the proactive command session, and MAY send the Disable command again without any further End User interaction.

  - Otherwise:

    - The eUICC SHALL mark the currently enabled Profile as "to be disabled" and return OK to the LPAd.
    - 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 [31].
    - Upon reception of the Terminal Response with result "command performed successfully" or upon reset of the eUICC, the eUICC SHALL disable the currently Enabled Profile. If the target Profile is a Test Profile and an Operational Profile was in Enabled state before the (first) Test Profile was enabled: the eUICC SHALL enable this Operational Profile. If this Operational Profile was deleted while the Test Profile was enabled, no error SHALL be generated and the function execution SHALL continue.
    - 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. Subsequent actions are implementation specific.

NOTE: In the case the Device supports the Single Wire Protocol interface to the eUICC, the Device SHOULD take the appropriate actions regarding this interface when Profile state is changed.

- If the target Profile is successfully disabled, 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:

```
DisableProfileRequest ::= [50] SEQUENCE { -- Tag 'BF32'
   profileIdentifier CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
      iccid Iccid -- ICCID, tag '5A'
   }
   refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
```

Response Data

The response data SHALL be coded as follows:

```
DisableProfileResponse ::= [50] SEQUENCE { -- Tag 'BF32'
```

```
   disableResult INTEGER {ok(0), iccidOrAidNotFound (1),
profileNotInEnabledState(2), disallowedByPolicy(3), catBusy(5),
undefinedError(127)}
}
```

Alternative Case 3 Command

In addition to the command data exchange described above, the eUICC SHALL support the following alternative command for the function:

The command data is sent to the eUICC in a Case 3 STORE DATA command:

- The STORE DATA APDU SHALL be coded as defined in section 5.7.2, with the exception of P1 which SHALL be set to '90'.
- The command data SHALL be coded as defined above, the response data SHALL not be present.
- The following additional status bytes are defined:
  '6A 82': Profile not found.
  '69 85': Profile not in enabled state or command not allowed by Profile Policy Rules.
  '93 00': CAT is busy. Command cannot be executed at present, further normal commands are allowed.

5.7.18 Function (ES10c): DeleteProfile

Related Procedures: Delete Profile

Function Provider Entity: ISD-R (LPA Services)

Description:

This function deletes a Profile from eUICC. This function can be used at any time by the LPAd. The Profile to be deleted can be identified by ISD-P AID or ICCID.

On reception of this command the eUICC SHALL check the state and the Profile Policy Rules of the target Profile. If Profile is in Enabled state or the Profile Policy Rules does not allow deletion of the Profile, the eUICC SHALL return an error; otherwise the eUICC SHALL delete the ISD-P containing the target Profile and its related Profile Metadata.

If the target Profile is successfully deleted, 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:

```
DeleteProfileRequest ::= [51] CHOICE { -- Tag 'BF33'
   isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
   iccid Iccid -- ICCID, tag '5A'
}
```

Response Data

The response data SHALL be coded as follows:

```
DeleteProfileResponse ::= [51] SEQUENCE { -- Tag 'BF33'
   deleteResult INTEGER {ok(0), iccidOrAidNotFound (1),
profileNotInDisabledState(2), disallowedByPolicy(3), undefinedError(127)}
}
```

Alternative Case 3 Command

In addition to the command data exchange described above, the eUICC SHALL support the following alternative command for the function:

The command data is sent to the eUICC in a Case 3 STORE DATA command:

- The STORE DATA APDU SHALL be coded as defined in section 5.7.2, with the exception of P1 which SHALL be set to '90'.
- The command data SHALL be coded as defined above, the response data SHALL not be present.
- The following additional status bytes are defined:
  '6A 82': Profile not found.
  '69 85': Profile not in disabled state or command not allowed by Profile Policy Rules.

5.7.19 Function (ES10c): eUICCMemoryReset

Related Procedures: eUICC Memory Reset

Function Provider Entity: LPA Services

Description:

This function deletes selected subsets of the Profiles stored on an eUICC regardless of their enabled status or any Profile Policy Rules. Two subsets are defined:

- Operational Profiles.
- Test Profiles that were not pre-installed (i.e., Test Profiles that were "field loaded").

NOTE: The identification of pre-installed Test Profiles is out of the scope of this specification.

The eUICC returns a status indicating whether any Profiles were deleted.

This function can also be used to reset the Default SM-DP+ address to its initial value.

Any combination MAY be specified.

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 command execution, the command SHALL stop and SHALL leave the eUICC and the involved Profiles in their original states prior to command execution.

Upon reception of the eUICCMemoryReset function and dependent on the parameters set, the eUICC SHALL do the following:

- In case there is an Enabled Profile and a proactive session is ongoing, the function SHALL return error code '`catBusy`'.

NOTE: 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.

- Otherwise the function SHALL:
  - 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. If there was an Enabled Profile, the ISD-R SHALL send a REFRESH proactive command to the Device.
  - Reset the Default SM-DP+ address to its initial value.
  - For each deleted Profile, the eUICC SHALL generate as many Notifications as configured in its Profile Metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.

Command Data

The command data SHALL be coded as follows:

```
EuiccMemoryResetRequest ::= [52] SEQUENCE { -- Tag 'BF34'
   resetOptions [2] BIT STRING {
      deleteOperationalProfiles(0),
      deleteFieldLoadedTestProfiles(1),
      resetDefaultSmdpAddress(2)}
}
```

Response Data

The response data SHALL be coded as follows:

```
EuiccMemoryResetResponse ::= [52] SEQUENCE { -- Tag 'BF34'
   resetResult INTEGER {ok(0), nothingToDelete(1), catBusy(5), undefinedError(127)}
}
```

5.7.20 Function (ES10c): GetEID

Related Procedures: Profile Download Initiation

Function Provider Entity: ISD-R (LPA Services)

Description:

This function gets the EID from the eUICC. This function can be used at any time by the LPA, and for instance during the Profile Download Initiation when the End user MAY have to provide the EID to the contracting Service Provider/Operator, and when the EID is not available by another mean, e.g. the End User MAY have lost the physical material where it was printed on.

Command Data

The data field SHALL indicate EID data object '5C 01 5A' (tag '5A' identifies the EID).

The command data SHALL be coded as follows:

```
GetEuiccDataRequest ::= [62] SEQUENCE { -- Tag 'BF3E'
   tagList [APPLICATION 28] Octet1  -- tag '5C', the value SHALL be set to '5A'
}
```

Response Data

The response data SHALL be coded as follows:

```
GetEuiccDataResponse ::= [62] SEQUENCE { -- Tag 'BF3E'
   eidValue [APPLICATION 26] Octet16  -- tag '5A'
}
```

In case the provided `tagList` is invalid or unsupported, the eUICC SHALL return an error status word.

5.7.21 Function (ES10c): SetNickname

Related Procedures: Set/Edit Nickname

Function Provider Entity: LPA Services

Description:

This function is used to add or update a Profile Nickname associated to one Profile present on-card.

Upon reception of the SetNickname function, the eUICC SHALL:

- Verify that the target Profile is present on the eUICC
- Update the target Profile nickname with the provided data

In case a Profile Nickname already exists for the indicated Profile, the Profile Nickname SHALL be updated with the new value. In case the new value is an empty string, the Profile Nickname SHALL be removed. Removing a non-existing Profile Nickname SHALL not be considered an error.

Command Data

The command data SHALL be coded as follows:

```
-- Definition of Profile Nickname Information
SetNicknameRequest ::= [41] SEQUENCE { -- Tag 'BF29'
   iccid Iccid,
   profileNickname [16] UTF8String (SIZE(0..64))
}
```

Response Data

The response data SHALL be coded as follows:

```
SetNicknameResponse ::= [41] SEQUENCE { -- Tag 'BF29'
   setNicknameResult INTEGER {ok(0), iccidNotFound (1), undefinedError(127)}
}
```

5.7.22 Function (ES10b): GetRAT

Related Procedures: Profile Download and Installation

Function Provider entity: ISD-R (LPA Services)

Description:

This function retrieves the Rules Authorisation Table (RAT) from the eUICC. It can be called at any time. The RAT is used by the LPA to determine if a Profile containing PPRs can be installed, conditionally with End User Consent, on the eUICC as defined in section 2.9.2.4.

Command data

The command data SHALL be coded as follows:

```
GetRatRequest ::= [67] SEQUENCE { -- Tag ' BF43'
   -- No input data
}
```

Response data

The response data SHALL be coded as follows:

```
GetRatResponse ::= [67] SEQUENCE { -- Tag 'BF43'
   rat RulesAuthorisationTable
}

RulesAuthorisationTable ::= SEQUENCE OF ProfilePolicyAuthorisationRule
ProfilePolicyAuthorisationRule ::= SEQUENCE {
   pprIds PprIds,
   allowedOperators SEQUENCE OF OperatorId,
   pprFlags BIT STRING {consentRequired(0)}
}
```

The list of `ProfilePolicyAuthorisationRule` data objects SHALL be returned in the same order as stored in the eUICC. This list MAY be empty.

The `pprIds` data object SHALL identify at least one PPR. The LPA SHALL ignore the `pprUpdateControl` bit.

The `allowedOperators` data object SHALL follow the description given in section 2.9.2.1.

The `consentRequired` bit set to 1 indicates that the End User consent is required.

5.8 ES11 (LPA -- SM-DS)

ES11 is the interface between:

- The LPA entity (more specifically the LDS endpoint).

- The SM-DS.

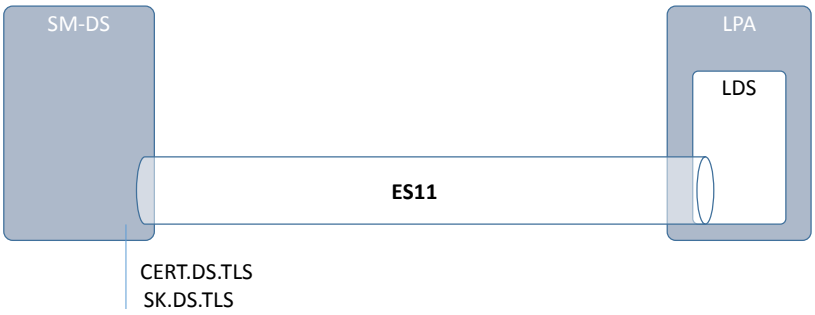


Figure 38: ES11

The LPA SHALL communicate with the SM-DS secured by TLS in server authentication mode as described in section 2.6.6.

The format of the TLS Certificates (CERT.DS.TLS) used for TLS connections is described in section 4.5.2.1.

During TLS establishment, LPA SHALL verify the received CERT.DS.TLS according to section 4.5.2.2. If any of these verifications fail, the TLS connection SHALL be rejected, and the on-going procedure SHALL fail.

5.8.1 Function: InitiateAuthentication

Related Procedures: Common Mutual Authentication for Event Retrieval

Function Provider Entity: SM-DS

Description:

This function is identical to "ES9+.InitiateAuthentication" where the SM-DS plays the role of SM-DP+ and where:

- The SM-DP+ SHALL be replaced by the SM-DS.
- CERT.DPauth.ECDSA SHALL be replaced by CERT.DSauth.ECDSA.
- smdpAddress SHALL be replaced by smdsAddress.

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.9.1	SM-DS Address	3.8	Refused	Invalid SM-DS Address.
8.9.2	Security configuration	3.1	Unsupported	None of the proposed Public Key Identifiers is supported by the SM-DS.

8.9.3	Specification Version Number	3.1	Unsupported	The Specification Version Number indicated by the eUICC is not supported by the SM-DS.
8.9.4	SM-DS Certificate	3.7	Unavailable	The SM-DS has no CERT.DS.ECDSA signed by one of the GSMA CI Public Key supported by the eUICC.

Table 48a: InitiateAuthentication Specific Status codes

5.8.2 Function: AuthenticateClient

Related Procedures: Common mutual authentication for Event Retrieval

Function Provider Entity: SM-DS

Description:

This function SHALL be called by the LPA to request the authentication of the eUICC by the SM-DS.

This function is correlated to a previous normal execution of an "ES11.InitiateAuthentication" function through a Transaction ID delivered by the SM-DS.

On reception of this function call, the SM-DS SHALL:

- Verify the validity of the CERT.EUM.ECDSA, using the public key PK.CI.ECDSA.
- Verify the validity of the CERT.EUICC.ECDSA, using the public key PK.EUM.ECDSA.
- Verify the eUICC signature (euiccSignature1) using the PK.EUICC.ECDSA as described in section 5.7.13 "ES10b.AuthenticateServer".
- Verify that the transactionId is known and relates to an ongoing RSP session.
- Verify that the serverChallenge attached to the ongoing RSP session matches the serverChallenge returned by the eUICC.

If any of these verifications fail, the SM-DS SHALL return a 'Function execution status' indicating 'Failed' with the relevant status code.

Otherwise the SM-DS SHALL retrieve and return the Event ID and RSP Server address pair(s) for the requesting eUICC depending on the content of the received ctxParams1:

- If `ctxParamsForCommonAuthentication` contains no matchingId data object or an empty value for matchingId, the SM-DS SHALL retrieve and return a list of Event ID and RSP Server address pairs corresponding to the EID. This list MAY be empty.
- If `ctxParamsForCommonAuthentication` contains a matchingId data object with an eventId value, the SM-DS SHALL retrieve and return the pair of the Event ID (corresponding to EventID1 described in section 3.6.1.2) and RSP Server address corresponding to the EID and eventId value. If no Event Record identified by the EID and eventId exist, the specific error (8.9.5-Event Record, 3.9-Unknown) SHALL be returned.

The SM-DS 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 Event Retrieval has been successfully executed.
- 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
transactionId	Transaction ID as generated by the SM-DS (section 3.1.2).	Binary[1-16]	1	M
authenticateServerResponse	Defined in section 5.7.13.	Authenticate ServerResponse	1	M

Table 49: AuthenticateClient Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
transactionID	Transaction ID as generated by the SM-DS (section 3.1.2).	Binary[1-16]	1	M
eventEntries	The list of Event ID and RSP Server address pair(s) for the EID.	EVENT_ENTRY	0..N	M

Table 50: AuthenticateClient Additional Output Data

The **EVENT_ENTRY** type is defined by the following data structure:

Data name	Description	Type	No.	MOC
eventId	Identification of the Event.	String	1	M
rspServerAddress	RSP Server address where the operation corresponding to the Event can be processed.	FQDN	1	M

Table 51: EVENT_ENTRY

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.1.2	EUM Certificate	6.1	Verification Failed	Certificate is invalid.
8.1.2	EUM Certificate	6.3	Expired	Certificate has expired.
8.1.3	eUICC Certificate	6.1	Verification Failed	Certificate is invalid.
8.1.3	eUICC Certificate	6.3	Expired	Certificate has expired.
8.1	eUICC	6.1	Verification Failed	eUICC signature is invalid or serverChallenge is invalid.
8.10.1	TransactionId	3.9	Unknown	The RSP session identified by the TransactionID is unknown.

8.9.5	Event Record	3.9	Unknown	No Event identified by the Event ID for the EID exists.

Table 52: AuthenticateClient specific Status Codes

5.9 ES12 (SM-DS -- SM-DP+)

The ES12 is used by the SM-DP+ to manage Event registration.

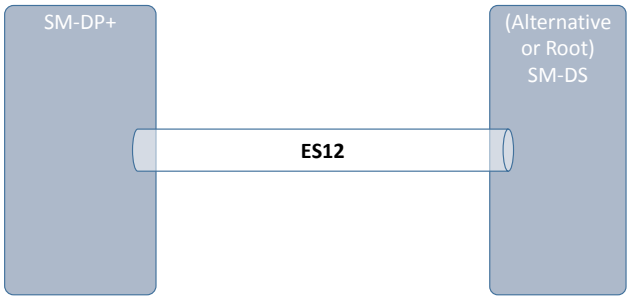


Figure 39: ES12

The SM-DP+ communicates with the SM-DS through a secure connection, by establishing a TLS connection with mutual authentication using CERT.DP.TLS and CERT.DS.TLS. Additional details about security requested on this interface and the level of data encryption are defined in section 2.6 and GSMA SAS SM specification [23].

5.9.1 Function: RegisterEvent

Related Procedures: Event Registration

Function Provider Entity: SM-DS (Alternative SM-DS or Root SM-DS)

Description:

This function registers an Event Record in the SM-DS (an Alternative SM-DS or a Root SM-DS).

The function caller MAY require that the Event registration is cascaded to the Root SM-DS using the input data '`forwardingIndicator`'. The Root SM-DS SHALL ignore this input data.

On reception of this function call, the SM-DS SHALL:

- Verify that the EventID is not already used by the function caller

If any of these verifications fail, the SM-DS SHALL return a 'Function execution status' indicating 'Failed' with the relevant status code.

Otherwise the SM-DS SHALL:

- Store the received Event Record, consisting of the EID, the RSP Server address and the EventID, all provided as function input data, together with the function caller identity (either an SM-DP+ or an SM-DS OID) received in the TLS Certificate.

- If required (`forwardingIndicator` input data set to 'true'), and if the function provider is an Alternative SM-DS, cascade the Event registration to the Root-SM-DS by calling the "ES15.RegisterEvent" with the relevant input data:
    - Same EID value.
    - Its own SM-DS address to be used for Event retrieval.
    - Its own generated Event ID corresponding to this incoming registration Event.

If registration cascading fails the SM-DS SHALL delete the locally stored Event Record and return a function execution status 'Failed' with the relevant status code.

The SM-DS MAY perform additional verifications and 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 Event has been registered (and cascaded if required).
- 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	Identification of the targeted eUICC.	EID	1	M
rspServerAddress	RSP Server address where the operation corresponding to the registered Event can be executed.	FQDN	1	M
eventId	Identification of the Event. The EventID SHALL be unique in the context of the function caller.	String	1	M
forwardingIndicator	Indicates if the registration has to be made to the Root SM-DS. The Root SM-SD SHALL ignore this input data.	Boolean	1	M

Table 53: RegisterEvent Additional Input Data

Additional Output Data:

No additional output data.

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.9.5	Event record	3.3	Already in use	The Event Record already exist in the SM-DS (EventID duplicated).
8.9	SM-DS	5.1	Inaccessible	The cascade SM-DS registration has failed. Root SM-DS was unavailable.

8.9	SM-DS	4.2	Execution error	The cascade SM-DS registration has failed. Root SM-DS has raised an error.

Table 54: RegisterEvent specific status codes

5.9.2 Function: DeleteEvent

Related Procedures: Event Deletion

Function Provider Entity: SM-DS

Description:

This function deletes an Event Record in the SM-DS (an Alternative SM-DS or a Root SM-DS).

On reception of this function call, the SM-DS SHALL:

- Retrieve the Event Record corresponding to the provided Event ID and the function caller identity (SM-DP+ or SM-DS OID) received in the TLS Certificate.

If any of these verifications fail, the SM-DS SHALL return a 'Function execution status' indicating 'Failed' with the relevant status code.

Otherwise the SM-DS SHALL:

- Determine if the Event Record registration has been cascaded to the Root SM-DS.
- If the Event Record registration was not cascaded, then the SM-DS SHALL delete the retrieved Event Record.
- If the Event Record registration was cascaded, then
    - the SM-DS SHALL cascade the deletion of the Event Record to the Root SM-DS by calling the "ES15.DeleteEvent" with the Event ID value generated by the SM-DS during Event Record registration.
    - If deletion cascading fails:
        - because the Event record was not found, the SM-DS SHALL ignore this error case and continue.
        - for any reason, the SM-DS SHALL return a function execution status 'Failed' with the relevant status code.
- If deletion cascading has succeeded, the SM-DS SHALL delete the retrieved Event Record.

The SM-DS MAY perform additional verifications and 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 Event has been deleted (and cascaded if required).
- 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	Identification of the targeted eUICC.	EID	1	M
eventId	Identification of the Event. The EventID SHALL be unique in the context of the function caller.	String	1	M

Table 55: DeleteEvent Additional Input Data

Additional Output Data:

No additional output data

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.9.4	Event Record	3.9	Unknown	The targeted Event Record doesn't exist.
8.9	SM-DS	5.1	Inaccessible	The cascade SM-DS deletion has failed. Root SM-DS was unavailable.
8.9	SM-DS	4.2	Execution error	The cascade SM-DS deletion has failed. Root SM-DS has raised an error.

Table 56: DeleteEvent specific Status Codes

5.10 ES15 (SM-DS -- SM-DS)

This interface is a particular case of the ES12 interface where an Alternative SM-DS is communicating to the Root SM-DS to manage the cascading of Events. This interface is functionally identical to ES12.

Figure 40: ES15

The Alternative SM-DS and the Root SM-DS communicate through a secure connection, by establishing a TLS connection with mutual authentication using their CERT.DS.TLS. Additional details about security requested on this interface and the level of data encryption are defined in section 2.6 and GSMA SAS SM specification [23].

5.10.1 Function: RegisterEvent

Related Procedures: Event Registration

Function Provider Entity: (Root) SM-DS

Description:

This function is identical to "ES12.RegisterEvent".

5.10.2 Function: DeleteEvent

Related Procedures: Event Deletion

Function Provider Entity: (Root) SM-DS

Description:

This function is identical to "ES12.DeleteEvent".

5.11 LUI in the eUICC (LUIe)

The implementation of the LUIe does not require an RSP-specific interface between the eUICC and the device.

A Device supporting LUIe SHALL support one of the generic mechanisms defined in this section.

NOTE: DeviceCapabilities MAY be empty if provided by the LPAe.

5.11.1 LUIe using CAT

In order to support this option, the Device SHALL support at least the CAT mechanisms defined in Annex C.4.

5.11.2 LUIe using SCWS

In order to support this option, the Device SHALL support at least the CAT mechanisms defined in Annex C.4.

In addition, the eUICC and the Device SHALL support the Smartcard Web Server as defined in [7].

6 Interface binding over HTTP

This section defines how to use HTTP/1.1, defined in RFC 2616 [48], and TLS, defined in RFC 5246 [16], as the transport layer to exchange ES2+, ES9+, ES11, ES12, and ES15 function requests and responses.

On ES9+ and ES11, the LPA always acts as an HTTP client and is in charge of managing the connection establishment to the RSP Server. The LPA SHALL use either JSON binding defined in section 6.5 or ASN.1 binding defined in section 6.6.

On ES2+, ES12, and ES15 any RSP Server MAY act as an HTTP client or an HTTP server. JSON binding defined in section 6.5 SHALL be used.

In case of communication failure, the HTTP client is responsible for retry and reconnection management.

6.1 TLS Security

Transport Layer Security (TLS) secures the messages exchanged between a function requester and function provider. TLS SHALL be used with mutual authentication on ES2+, ES12, and ES15. TLS SHALL be used with server authentication on ES9+ and ES11.

This specification mandates usage of TLS v1.2 defined in RFC 5246 [16] to allow appropriate algorithm and key length.

6.1.1 Identification/Authentication/Authorisation

If applicable on the interface, authentication of the sending party of a JSON message SHALL rely on the Transport layer security (using TLS certificate of the sending party).

6.1.2 Integrity

The integrity of the message SHALL exclusively rely on the Transport Layer Security (TLS).

6.1.3 Confidentiality

The confidentiality of the message SHALL exclusively rely on the Transport Layer Security (TLS).

6.2 HTTP request and response

An HTTP POST request SHALL be used to transport a single function execution request. The corresponding function execution response SHALL be returned as defined in SGP.02 [02] depending on the used Message Exchange Pattern (MEP).

This specification uses the following MEPs:

- Synchronous Request-Response: the request payload SHALL be sent in the HTTP POST request, and the function execution response SHALL be returned in the HTTP POST response.
- Notification: the notification payload SHALL be sent in the HTTP POST request and the HTTP POST response body SHALL be empty.

HTTP POST request for ES9+ SHALL contain a "`User-Agent`" header field as defined hereunder:

```
User-Agent: gsma-rsp-lpad or gsma-rsp-lpae
```

The "`User-Agent`" field MAY contain additional information after a semicolon.

HTTP POST request and response SHALL contain an "`X-Admin-Protocol`" header field as defined hereunder:

```
Admin-Protocol: gsma/rsp/v<x.y.z>
```

Where:
<x.y.z> indicates the highest version of SGP.22 [This document] supported by the sender. When the sender is the Device, this indicates the highest version supported by the LPA.

HTTP POST request and response SHALL contain a "`Content-type`" header field to indicate the nature of the binding. A JSON binding SHALL be indicated by the value

"`application/json`". An ASN-1 binding SHALL be indicated by the value "`application/x-gsma-rsp-asn1`". The "`Content-type`" header field of an HTTP response SHOULD NOT be set when the body is empty (e.g. case of notification function response). If present, it SHALL be ignored.

HTTP POST request and response MAY contain additional header fields. Their use is out of scope of this specification.

6.3 HTTP response status codes

Standard HTTP status codes SHALL apply to this section.

Status codes '1xx' (Information), '3xx' (Redirection), '4xx' (HTTP client error) and '5xx' (HTTP server error) MAY be used by the RSP Server (i.e. the HTTP server).

The retry policy for HTTP request answered with status codes '4xx' and '5xx' is out of scope of this specification.

A normal request-response function execution status (MEP Synchronous request-response) SHALL be indicated by the HTTP status code '200' (OK) in the HTTP response, regardless whether the function response is an error or a success, as defined in SGP.02 [02].

A normal notification function execution status (MEP Notification) SHALL be indicated by the HTTP status code '204' (No Content) with an empty HTTP response body as defined in SGP.02 [02].

Other status codes '2xx' SHALL not be used by the RSP Server.

6.4 Secure Channel Set-Up on ES2+

The process of setting up secure channel is out of scope of this document. This process includes the exchange of the following information:

- Function requester and Function provider OIDs and identity SHALL be registered to GSMA Policy Authority and respective values have been communicated to each party.
- Function requester and Function provider URL SHALL have been communicated to each party.
- Function requester and Function provider parties' trust SHALL have been established on an X-509 certificate chain basis.

6.5 Function Binding in JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is based on a subset of the JavaScript Programming Language. JSON is a text format that is completely language independent.

6.5.1 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.

```
HTTP POST <HTTP Path> HTTP/1.1
Host: <Server Address>
User-Agent: gsma-rsp-lpad
X-Admin-Protocol: gsma/rsp/v<x.y.z>
Content-Type: application/json
Content-Length: <Length of the JSON requestMessage>

<JSON requestMessage>
```

The <HTTP Path> is used to indicate which function execution is requested by the HTTP client. The list of defined <HTTP Path> are described in section 6.5.2.

- 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
Content-Length: <Length of the JSON responseMessage>

<JSON responseMessage>
```

6.5.1.1 Definition of <JSON requestMessage>

<JSON requestMessage> is the combination of:

- <JSON requestHeader>
- <JSON body> which depends on the function called

HTTP messages for ES9+ and ES11 SHALL not contain the <JSON requestHeader>.

6.5.1.2 Definition of <JSON responseMessage>

<JSON responseMessage> is the combination of:

- <JSON responseHeader>
- <JSON body> which depends on the function called

The HTTP POST response body SHALL be empty for MEP notification message (see section 6.3).

6.5.1.3 Definition of <JSON requestHeader>

The <JSON requestHeader> maps the function input header.

```
{
   "header" : {
      "type" : "object",
      "properties" : {
         "functionRequesterIdentifier" : {
            "type" : "string",
            "description" : "identification of the function requester"
         },
         "functionCallIdentifier" : {
            "type" : "string",
            "description" : "identification of the function call"
         }
```

```
      },
      "required" : ["functionRequesterIdentifier", "functionCallIdentifier"]
   }
}
```

6.5.1.4 Definition of <JSON responseHeader>

The <JSON responseHeader> maps the function output header.

```
{
   "header" : {
      "type" : "object",
      "properties" : {
         "functionExecutionStatus" : {
            "type" : "object",
            "description" : "Whether the function has been processed correctly or
not"
            "properties" : {
               "status" : {
                  "type" : "string",
                  "description" : " Executed-Success, Executed-WithWarning, Failed,
Expired"
               },
               "statusCodeData" : {
                  "type" : "object",
                  "properties" : {
                     "subjectCode" : {
                        "type" : "string",
                        "description" : "OID of the subject code"
                     },
                     "reasonCode" : {
                        "type" : "string",
                        "description" : "OID of the reason code"
                     },
                     "subjectIdentifier" : {
                        "type" : "string",
                        "description" : "Identifier of the subject "
                     },
                     "message" : {
                        "type" : "string",
                        "description" : "Textual and human readable explanation"
                     }
                  },
                  "required" : ["subjectCode", "reasonCode"]
               },
            },
            "required" : ["status"]
         }
      },
      "required" : ["functionExecutionStatus"]
   }
}
```

6.5.2 List of functions

	Function	Path	MEP
ES2+	DownloadOrder	/gsma/rsp2/es2plus/downloadOrder	Synchronous
	ConfirmOrder	/gsma/rsp2/es2plus/confirmOrder	Synchronous
	CancelOrder	/gsma/rsp2/es2plus/cancelOrder	Synchronous
	ReleaseProfile	/gsma/rsp2/es2plus/releaseProfile	Synchronous
	HandleDownloadProgressInfo	/gsma/rsp2/es2plus/handleDownloadProgressInfo	Notification
ES9+	InitiateAuthentication	/gsma/rsp2/es9plus/initiateAuthentication	Synchronous
	AuthenticateClient	/gsma/rsp2/es9plus/authenticateClient	Synchronous
	GetBoundProfilePackage	/gsma/rsp2/es9plus/getBoundProfilePackage	Synchronous
	HandleNotification	/gsma/rsp2/es9plus/handleNotification	Notification
	CancelSession	/gsma/rsp2/es9plus/cancelSession	Synchronous
ES11	InitiateAuthentication	[As ES9+]	[As ES9+]
	AuthenticateClient	[As ES9+]	[As ES9+]
ES12	RegisterEvent	/gsma/rsp2/es12/registerEvent	Synchronous
	DeleteEvent	/gsma/rsp2/es12/deleteEvent	Synchronous
ES15	RegisterEvent	[As ES12]	[As ES12]
	DeleteEvent	[As ES12]	[As ES12]

Table 57: List of Functions

6.5.2.1 "ES2+.DownloadOrder" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.DownloadOrder" function:

```
{
   "type" : "object",
   "properties" : {
      "eid" : {
         "type" : "string",
         "pattern" : "^[0-9]{32}$",
         "description" : "EID as desc in SGP.02"
      },
      "iccid" : {
         "type" : "string",
         "pattern" : "^[0-9]{19}[0-9F]?$",
         "description" : "ICCID as described in section 5.2.1"
      },
      "profileType" : {
         "type" : "string",
         "description" : "content free information defined by the Operator"
      }
   }
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES2+.DownloadOrder" function:

```
{
   "type" : "object",
```

```
   "properties" : {
     "iccid" : {
       "type" : "string",
       "pattern" : "^[0-9]{19}[0-9F]?$",
       "description" : "ICCID as described in section 5.2.1"
     }
   },
   "required" : ["iccid"]
}
```

6.5.2.2 "ES2+.ConfirmOrder" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.ConfirmOrder" function:

```
{
   "type" : "object",
   "properties" : {
     "iccid" : {
       "type" : "string",
       "pattern" : "^[0-9]{19}[0-9F]?$",
       "description" : "ICCID as described in section 5.2.1"
     },
     "eid" : {
       "type" : "string",
       "pattern" : "^[0-9]{32}$",
       "description" : "EID as desc in SGP.02"
     },
     "matchingId" : {
       "type" : "string",
       "description" : "as defined in section {5.3.2}"
     },
     "confirmationCode" : {
       "type" : "string",
       "description" : "as defined in section {5.3.2}"
     },
     "smdsAddress" : {
       "type" : "string",
       "description" : "as defined in section {5.3.2}"
     },
     "releaseFlag" : {
       "type" : "boolean",
       "description" : "as defined in section {5.3.2}"
     }
   },
   "required" : ["iccid", "releaseFlag"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES2+.ConfirmOrder" function:

```
{
   "type" : "object",
   "properties" : {
     "eid" : {
       "type" : "string",
       "pattern" : "^[0-9]{32}$",
       "description" : "EID as desc in SGP.02"
```

```
    },
    "matchingId" : {
      "type" : "string",
      "description" : "as defined in section {5.3.2}"
    },
    "smdpAddress" : {
      "type" : "string",
      "description" : "as defined in section {5.3.2}"
    }
  },
  "required" : []
}
```

6.5.2.3 "ES2+.CancelOrder" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.CancelOrder" function:

```
{
  "type" : "object",
  "properties" : {
    "iccid" : {
      "type" : "string",
      "pattern" : "^[0-9]{19}[0-9F]?$",
      "description" : "ICCID as described in section 5.2.1"
    },
    "eid" : {
      "type" : "string",
      "pattern" : "^[0-9]{32}$",
      "description" : "EID as desc in SGP.02"
    },
    "matchingId" : {
      "type" : "string",
      "description" : "as defined in section {5.3.2}"
    },
    "finalProfileStatusIndicator" : {
      "type" : "string",
      "description" : "as defined in section {5.3.4}"
    }
  },
  "required" : ["iccid"]
}
```

6.5.2.4 "ES2+.ReleaseProfile" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.ReleaseProfile" function:

```
{
  "type" : "object",
  "properties" : {
    "iccid" : {
      "type" : "string",
      "pattern" : "^[0-9]{19}[0-9F]?$",
      "description" : "ICCID as described in section 5.2.1"
      },
  "required" : ["iccid"]
  }
}
```

6.5.2.5 "ES2+.HandleDownloadProgressInfo" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.HandleDownloadProgressInfo" function:

```
{
   "type" : "object",
   "properties" : {
     "eid" : {
        "type" : "string",
        "pattern" : "^[0-9]{32}$",
        "description" : "EID as described in SGP.02"
     },
     "iccid" : {
        "type" : "string",
        "pattern" : "^[0-9]{19}[0-9F]?$",
        "description" : "ICCID as described in section 5.2.1"
     },
     "profileType" : {
        "type" : "string",
        "description" : "Content free information defined by the Operator
(e.g.'P9054-2')"
     },
     "timestamp" : {
        "type" : "string",
        "pattern" : "$[0-9]{4}-[0-9]{2}-[0-9]{2}T[0-9]{2}:[0-9]{2}:[0-
9]{2}[T,D,Z]{1}$",
        "description" : "String format as specified by W3C: YYYY-MM-DDThh:mm:ssTZD
(E.g. 2001-12-17T09:30:47Z)"
     },
     "notificationPointId" : {
        "type" : "integer",
        "description" : "Identification of the step reached in the procedure"
     },
     "notificationPointStatus" : {
        "type" : "object",
        "description" : "ExecutionStatus Common Data Type"
        "properties" : {
          "status" : {
            "type" : "string",
            "description" : "Executed-Success, Executed-WithWarning, Failed or
Expired"
          },
          "statusCodeData" : {
            "type" : "object",
            "properties" : {
              "subjectCode" : {
                "type" : "string",
                "description" : "OID of the subject code"
              },
              "reasonCode" : {
                "type" : "string",
                "description" : "OID of the reason code"
              },
              "subjectIdentifier" : {
                "type" : "string",
                "description" : "Identifier of the subject"
              },
              "message" : {
```

```
                  "type" : "string",
                  "description" : "Textual and human readable explanation"
               }
            },
            "required" : ["subjectCode", "reasonCode"]
         },
      },
      "required" : ["status"]
   },
   "resultData" : {
      "type" : "string",
      "format" : "base64",
      "description" : "base64 encoded binary data containing the Result data
contained in the ProfileInstallationResult"
   }
 },
 "required" : ["iccid", "profileType", "timestamp", "notificationPointId",
"notificationPointStatus"]
}
```

6.5.2.6 "ES9+.InitiateAuthentication" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.InitiateAuthentication" function:

```
{
  "type" : "object",
  "properties" : {
    "euiccChallenge" : {
      "type" : "string",
      "format" : "base64",
      "description" : "base64 encoded binary data containing eUICC Challenge
defined in Section 5.6.1"
    },
    "euiccInfo1" : {
      "type" : "string",
      "format" : "base64",
      "description" : "base64 encoded binary data containing euiccinfo1 defined
in Section 5.6.1"
    },
    "smdpAddress" : {
      "type" : "string",
      "description" : "SM-DP+ Address as defined in Section 5.6.1"
    }
  },
  "required" : ["euiccChallenge", "euiccInfo1", "smdpAddress"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES9+.InitiateAuthentication" function:

```
{
  "type" : "object",
  "properties" : {
    "transactionId" : {
      "type" : "string",
      "pattern" : "^[0-9,A-F]{2,32}$",
```

```
        "description" : "Hexadecimal representation of the TransactionID defined
in Section 5.6.1"
     },
     "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"
     },
     "euiccCiPKIdToBeUsed" : {
        "type" : "string",
        "format" : "base64",
        "description" : "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"
     }
   },
   "required" : ["transactionId", "serverSigned1", "serverSignature1",
"euiccCiPKIdToBeUsed", "serverCertificate"]
}
```

NOTE: LPA is in charge of transcoding the transactionId.

6.5.2.7 "ES9+.GetBoundProfilePackage" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.GetBoundProfilePackage" function:

```
{
   "type" : "object",
   "properties" : {
     "transactionId" : {
        "type" : "string",
        "pattern" : "^[0-9,A-F]{2,32}$",
        "description" : "Hexadecimal representation of the TransactionID defined
in Section 5.6.2"
     },
     "prepareDownloadResponse" : {
        "type" : "string",
        "format" : "base64",
        "description" : "base64 encoded binary data containing
PrepareDownloadResponse defined in Section 5.6.2"
     }
   },
   "required" : ["transactionId", "prepareDownloadResponse"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES9+.GetBoundProfilePackage" function:

```
{
   "type" : "object",
   "properties" : {
     "transactionId" : {
        "type" : "string",
        "pattern" : "^[0-9,A-F]{2,32}$",
        "description" : "Hexadecimal representation of the TransactionID defined
in Section 5.6.2"
     },
     "boundProfilePackage" : {
        "type" : "string",
        "format" : "base64",
        "description" : "base64 encoded binary data containing Bound Profile
Package defined in Section 5.6.2"
     }
   },
   "required" : ["transactionId", "boundProfilePackage"]
}
```

6.5.2.8 "ES9+.AuthenticateClient" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.AuthenticateClient" function:

```
{
   "type" : "object",
   "properties" : {
     "transactionId" : {
        "type" : "string",
        "pattern" : "^[0-9,A-F]{2,32}$",
        "description" : "Hexadecimal representation of the TransactionID defined
in Section 5.6.3"
     },
     "authenticateServerResponse" : {
        "type" : "string",
        "format" : "base64",
        "description" : "base64 encoded binary data containing
AuthenticateServerResponse defined in Section 5.6.3"
     }
   },
   "required" : ["transactionId", "authenticateServerResponse"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES9+.AuthenticateClient" function:

```
{
   "type" : "object",
   "properties" : {
     "transactionId" : {
        "type" : "string",
        "pattern" : "^[0-9,A-F]{2,32}$",
        "description" : "Hexadecimal representation of the TransactionID defined
in Section 5.6.3"
```

```
     },
     "profileMetadata" : {
       "type" : "string",
       "format" : "base64",
       "description" : " base64 encoded binary data containing
StoreMetadataRequest defined in section 5.5.3",
     },
     "smdpSigned2" : {
       "type" : "string",
       "format" : "base64",
       "description" : "SmdpSigned2 encoded data object"
     },
     "smdpSignature2" : {
       "type" : "string",
       "format" : "base64",
       "description" : "SM-DP+ signature as defined in ES10b.PrepareDownload"
     },
     "smdpCertificate" : {
       "type" : "string",
       "format" : "base64",
       "description" : "The Certificate as required by ES10b.PrepareDownload"
     } },
   "required" : ["transactionId", "profileMetadata", " smdpSigned2",
"smdpSignature2", "smdpCertificate"]
}
```

Depending on the targeted RSP Server (SM-DP+ or SM-DS) the response MAY be a <JSON body> corresponding to "ES9+.AuthenticateClient" or "ES11.AuthenticateClient" function.

6.5.2.9 "ES9+.HandleNotification" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.HandleNotification" function.

```
{
   "type" : "object",
   "properties" : {
     "pendingNotification" : {
       "type" : "string",
       "format" : "base64",
       "description" : "base64 encoded binary data containing the
PendingNotification defined in section 5.7.10"
     }
   },
   "required" : ["pendingNotification"]
}
```

6.5.2.10 "ES9+.CancelSession" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.CancelSession" function:

```
{
   "type" : "object",
   "properties" : {
```

```
     "transactionId" : {
        "type" : "string",
        "pattern" : "^[0-9,A-F]{2,32}$",
        "description" : "Hexadecimal representation of the TransactionID defined
in Section 5.6.5"
     },
     "cancelSessionResponse" : {
        "type" : "string",
        "format" : "base64",
        "description" : "base64 encoded binary data containing
CancelSessionResponse data object defined in Section 5.7.14"
     }
   },
   "required" : ["transactionId", "cancelSessionResponse"]
}
```

The "ES9+.CancelSession" function has no <JSON body> part of the <JSON responseMessage>.

6.5.2.11 "ES11.InitiateAuthentication" Function

The <JSON body> part of the <JSON requestMessage> and <JSON body> part of the <JSON responseMessage> corresponding to the "ES11.InitiateAuthentication" function is identical to the one defined for the "ES9+.InitiateAuthentication" function.

6.5.2.12 "ES11.AuthenticateClient" Function

The <JSON body> part of the <JSON requestMessage> corresponding to the "ES11.AuthenticateClient" function is identical to the one defined for the "ES9+.AuthenticateClient" function. Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES11.AuthenticateClient" function:

```
{
   "type" : "object",
   "properties" : {
     "transactionId" : {
        "type" : "string",
        "pattern" : "^[0-9,A-F]{2,32}$",
        "description" : "Hexadecimal representation of the TransactionID, see
Section 5.8.2"
     },
     "eventEntries" : {
        "type" : array,
        "items" : {
          "type" : "object",
          "description" : "data containing Event-Entry, see section 5.8.2",
          "properties" : {
            "eventId" : {
              "type" : "string",
              "description" : "Identification of the event"
            },
            "rspServerAddress" : {
              "type" : "string",
              "description" : "RSP Server address where the event can be found"
            }
          },
```

```
          "required" : ["eventId", "rspServerAddress"]
        }
      }
   },
   "required" : ["transactionId", "eventEntries"]
}
```

Depending on the targeted RSP Server (SM-DP+ or SM-DS) the response MAY be a <JSON body> corresponding to "ES9+.AuthenticateClient" or "ES11.AuthenticateClient" function.

6.5.2.13 "ES12.RegisterEvent" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES12.RegisterEvent" function:

```
{
   "type" : "object",
   "properties" : {
     "eid" : {
        "type" : "string",
        "pattern" : "^[0-9]{32}$",
        "description" : "EID as desc in SGP.02"
     },
     "rspServerAddress" : {
        "type" : "string",
        "description" : "as defined in section 5.9.1"
     },
     "eventId" : {
        "type" : "string",
        "description" : "as defined in section 5.9.1"
     },
     "forwardingIndicator" : {
        "type" : "boolean",
        "description" : "as defined in section 5.9.1"
     }
   },
   "required" : ["eid", "rspServerAddress", "eventId", "forwardingIndicator"]
}
```

This function has no <JSON body> part of the <JSON responseMessage>.

6.5.2.14 "ES12.DeleteEvent" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES12.DeleteEvent" function:

```
{
   "type" : "object",
   "properties" : {
     "eid" : {
        "type" : "string",
        "pattern" : "^[0-9]{32}$",
        "description" : "EID as desc in SGP.02"
```

```
    },

    "eventId" : {
       "type" : "string",

       "description" : "as defined in section 5.9.2"
    }
  },
  "required" : ["eid", "eventId"]
}
```

This function has no <JSON body> part of the <JSON responseMessage>.

6.6 Function Binding in ASN.1

The LPAe SHALL exchange the HTTPS POST requests and responses defined in this section using BIP over TCP.

6.6.1 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.

```
HTTP POST gsma/rsp2/asn1 HTTP/1.1
Host: <Server Address>
User-Agent: gsma-rsp-lpae
X-Admin-Protocol: gsma/rsp/v<x.y.z>
Content-Type: application/x-gsma-rsp-asn1
Content-Length: <Length of the ASN.1 RemoteProfileProvisioningRequest>

<ASN.1 RemoteProfileProvisioningRequest>
```

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 Remote Profile Provisioning Request objects defined as follows:

```
RemoteProfileProvisioningRequest ::= [2] CHOICE {  -- Tag 'A2'
   initiateAuthenticationRequest [57] InitiateAuthenticationRequest,  -- Tag 'BF39'
   authenticateClientRequest [59] AuthenticateClientRequest, -- Tag 'BF3B'
   getBoundProfilePackageRequest [58] GetBoundProfilePackageRequest,  -- Tag 'BF3A'
   cancelSessionRequestEs9 [65] CancelSessionRequestEs9, -- Tag 'BF41'
   handleNotification [61] HandleNotification -- tag 'BF3D'
}
```

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 RemoteProfileProvisioningResponse>

<ASN.1 RemoteProfileProvisioningResponse>
```

The body part of the HTTP POST response SHALL contain one Remote Profile Provisioning Response object defined as follows:

```
RemoteProfileProvisioningResponse ::= [2] CHOICE { -- Tag 'A2'
   initiateAuthenticationResponse [57] InitiateAuthenticationResponse, -- Tag
'BF39'
   authenticateClientResponseEs9 [59] AuthenticateClientResponseEs9, -- Tag 'BF3B'
   getBoundProfilePackageResponse [58] GetBoundProfilePackageResponse, -- Tag
'BF3A'
   cancelSessionResponseEs9 [65] CancelSessionResponseEs9, -- Tag 'BF41'
   authenticateClientResponseEs11 [64] AuthenticateClientResponseEs11 -- Tag 'BF40'
}
```

6.6.2 List of functions

6.6.2.1 "ES9+.InitiateAuthentication" Function

The "ES9+.InitiateAuthentication" request function is defined as follows:

```
InitiateAuthenticationRequest ::= [57] SEQUENCE { -- Tag 'BF39'
   euiccChallenge [1] Octet16, -- random eUICC challenge
   smdpAddress [3] UTF8String,
   euiccInfo1 EUICCInfo1
}
```

The "ES9+.InitiateAuthentication" response function is defined as follows:

```
InitiateAuthenticationResponse ::= [57] CHOICE { -- Tag 'BF39'
   initiateAuthenticationOk InitiateAuthenticationOkEs9,
   initiateAuthenticationError INTEGER {
      invalidDpAddress(1),
      euiccVersionNotSupportedByDp(2),
      ciPKIdNotSupported(3)
   }
}

InitiateAuthenticationOkEs9 ::= SEQUENCE {
   transactionId [0]TransactionId, -- The TransactionID generated by the SM-DP+
   serverSigned1 ServerSigned1, -- Signed information
   serverSignature1 [APPLICATION 55] OCTET STRING, -- Server Sign1, tag '5F37'
   euiccCiPKIdToBeUsed SubjectKeyIdentifier, -- The CI Public Key to be used as
required by ES10b.AuthenticateServer
   serverCertificate Certificate
}
```

6.6.2.2 "ES9+.AuthenticateClient" Function

The "ES9+.AuthenticateClient" request function is defined as follows:

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

The "ES9+.AuthenticateClient" response function is defined as follows:

```
AuthenticateClientResponseEs9 ::= [59] CHOICE {  -- Tag 'BF3B'
   authenticateClientOk AuthenticateClientOk,
   authenticateClientError INTEGER {
      eumCertificateInvalid(1),
```

```
      eumCertificateExpired(2),
      euiccCertificateInvalid(3),
      euiccCertificateExpired(4),
      euiccSignatureInvalid(5),
      matchingIdRefused(6),
      eidMismatch(7),
      noEligibleProfile(8),
      ciPKUnknown(9),
      invalidTransactionId(10),
      insufficientMemory(11),
      undefinedError(127)
   }
}

AuthenticateClientOk ::= SEQUENCE {
   transactionId [0] TransactionId,
   profileMetaData [37] StoreMetadataRequest,
   smdpSigned2 SmdpSigned2, -- Signed information
   smdpSignature2 [APPLICATION 55] OCTET STRING,      -- tag '5F37'
   smdpCertificate Certificate    -- CERT.DPpb.ECDSA
}
```

6.6.2.3 "ES9+.GetBoundProfilePackage" Function

The "ES9+.GetBoundProfilePackage" request function is defined as follows:

```
GetBoundProfilePackageRequest ::= [58] SEQUENCE {  -- Tag 'BF3A'
   transactionId [0] TransactionId,
   prepareDownloadResponse [33] PrepareDownloadResponse
}
```

The "ES9+.GetBoundProfilePackage" response function is defined as follows:

```
GetBoundProfilePackageResponse ::= [58] CHOICE {  -- Tag 'BF3A'
   getBoundProfilePackageOk GetBoundProfilePackageOk,
   getBoundProfilePackageError INTEGER {
      euiccSignatureInvalid(1),
      confirmationCodeMissing(2),
      confirmationCodeRefused(3),
      confirmationCodeRetriesExceeded(4),
      invalidTransactionId(95),
      undefinedError(127)
   }
}

GetBoundProfilePackageOk ::= SEQUENCE {
   transactionId [0] TransactionId,
   boundProfilePackage [54] BoundProfilePackage
}
```

NOTE: The eUICC MAY start processing of the BPP before having received the full package and having been able to check for a correct TLV structure.

6.6.2.4 "ES9+.HandleNotification" Function

The "ES9+.HandleNotification" request function SHALL consist of the data structure defined for PendingNotification in section 5.7.10. The function is defined as follows:

```
HandleNotification ::= [61] SEQUENCE { -- Tag 'BF3D'
```

```
   pendingNotification PendingNotification
}
```

The function has no response.

6.6.2.5 "ES9+.CancelSession" Function

The "ES9+.CancelSession" request function is defined as follows:

```
CancelSessionRequestEs9 ::= [65] SEQUENCE { -- Tag 'BF41'
   transactionId TransactionId,
   cancelSessionResponse  CancelSessionResponse -- data structure defined for
ES10b.CancelSession function
}
```

The "ES9+.CancelSession" response function is defined as follows:

```
CancelSessionResponseEs9 ::= [65] CHOICE { -- Tag 'BF41'
   cancelSessionOk CancelSessionOk,
   cancelSessionError INTEGER {
      invalidTransactionId(1),
      euiccSignatureInvalid(2),
      undefinedError(127)
   }
}

CancelSessionOk ::= SEQUENCE { -- This function has no output data
}
```

6.6.2.6 Void"ES11.InitiateAuthentication" Function

The `InitiateAuthenticationRequest` and `InitiateAuthenticationResponse` for the binding of the "ES11.InitiateAuthentication" function are identical to the ones defined for the "ES9+.InitiateAuthentication" function.

6.6.2.7 "ES11.AuthenticateClient" Function

The `AuthenticateClientRequest` for the binding of the "ES11.AuthenticateClient" function is identical to the one defined for the "ES9+.AuthenticateClient" function.

The "ES11.AuthenticateClient" response data object is defined as follows:

```
AuthenticateClientResponseEs11 ::= [64] CHOICE {  -- Tag 'BF40'
   authenticateClientOk AuthenticateClientOkEs11,
   authenticateClientError INTEGER {
      eumCertificateInvalid(1),
      eumCertificateExpired(2),
      euiccCertificateInvalid(3),
      euiccCertificateExpired(4),
      euiccSignatureInvalid(5),
      eventIdUnknown(6),
      invalidTransactionId(7),
      undefinedError(127)
   }
}

AuthenticateClientOkEs11 ::= SEQUENCE {
   transactionId TransactionId,
   eventEntries SEQUENCE OF EventEntries
}
```

```
EventEntries ::= SEQUENCE {
   eventId UTF8String,
   rspServerAddress UTF8String
}
```

Annex A Use of GlobalPlatform Privileges (Normative)

The eUICC architecture defined in this specification relies on the ISD-R, ISD-P, MNO-SD and ECASD Security Domains defined in SGP.02 [2].

The GlobalPlatform privileges allocation defined in SGP.02 [2] SHALL be applicable for the ISD-R, ISD-P, MNO-SD and ECASD Security Domains as well as Applications inside a Profile.

Annex B Data Definitions (Normative)

- Coding of the IMEI

The value of the IMEI SHALL be coded as defined in section 4.2.

Annex C Device Requirements (Normative)

C.1 Functional Device Requirements

Functional Device Requirements No.	Requirement
DEV1	For connectivity the Device SHALL support at least one of the network access technologies defined by 3GPP or 3GPP2: • UDP over IP as defined in RFC 768 [34] (subject to the right support of access network technology) • TCP over IP as defined in RFC 793 [19].
DEV2	For Network connection control the Device SHALL support: • RPLMN details (LAC/TAC, NMR). • QoS (failures, duration, power, location). • New network selection after SIM/USIM update.
DEV3	The Device SHALL contain a unique IMEI (International Mobile Equipment Identity) value compliant with the format defined in 3GPP TS 23.003 [35] and/or a unique MEID as defined in 3GPP2 S.R0048-A [36].
DEV4	The Device SHALL support, as a minimum, the following set of proactive commands: • PROVIDE LOCAL INFORMATION (location information, IMEI, NMR, date and time, access technology, at least). • POLL INTERVAL, POLLING OFF, TIMER MANAGEMENT [at least one timer], ENVELOPE (TIMER EXPIRATION). • SET UP EVENT LIST and ENVELOPE (EVENT DOWNLOAD). • REFRESH Command (At least mode 4 - "UICC reset")
DEV5	The Device SHALL comply with the IMEI security requirements defined in the GSMA-EICTA document "Security Principles Related to Handset Theft" [22].
DEV6	A Device SHALL be able to handle an eUICC without any installed Profiles.
DEV7	If a Companion Device does not have the capability itself to communicate directly with the SM-DP+, it SHALL use a Primary Device as a conduit, allowing it to communicate with the SM-DP+.
DEV8	At least one of the Primary or Companion Device SHALL have a UI that allows the secure capture of User Intent.
DEV9	At least one of the Primary or Companion Device SHALL have a UI that allows the user to initiate a Profile Download or Local Profile Management.
DEV10	The Device SHALL conform to the terminal requirements within ETSI TS 102 221 [6].
DEV11	A Device implementation of personalisation ("SIM lock") as defined by 3GPP TS 22.022 SHALL operate the same with an enabled eUICC Profile as with a legacy UICC.
DEV12	An NFC Device SHALL retrieve and enforce access control rules as specified in the GlobalPlatform SEAC specification [56].
DEV13	An NFC Device SHALL at least have a non-removable eUICC or have the capability to support a removable eUICC that is compliant with the Contactless eUICC category as defined in section 4.3 in either instance.

Table 58: Device requirements

C.2 Requirements for Companion Device Scenarios

Secure interaction between the Primary Device and the Companion Device
The LPAd of the Companion Device SHALL support secure capture of the End User intent for the purpose of Local Profile Management through the Primary Device when the Companion Device has to rely on the Primary Device for UI function. This SHALL further include a secure pairing and secured communication between the Primary and Companion Device. The End User MAY perform local Profile download and management towards the eUICC in the Companion Device using the Primary Device.

A secure point to point proximity link at transport level between the Primary Device and the Companion Device SHALL either be implemented by the Primary and Companion Devices' OEM(s), or it SHALL be established as follows:

1. The End User connects the Companion Device to a router (e.g. the Primary Device which shares the network) which will assign an address for the Companion Device.

NOTE: This step MAY be performed at any time prior to step 2.

2. The End User uses the LPAd on the Companion Device to generate a HTTPS URL which includes the Companion Device address (e.g. private local IP address) and security information (i.e. 128-bit random secret key).

   In order to achieve the interoperability between different OEM Devices, the HTTPS URL SHALL be specified as https://hpath/LPA_access_token, where "hpath" is the Companion Device address and "LPA_access_token" is the security information.

3. The LPAd on the Companion Device indicates the HTTPS URL including the Companion Device address and the security information to the Primary Device using one of following non-exhaustive example means:

   - The Companion Device transfers the HTTPS URL to the Primary Device, e.g. using NFC.
   - The Companion Device displays the HTTPS URL which could be input by the End User to the Primary Device.
   - The Companion Device transforms the HTTPS URL into a QR code or bar code so that the Primary Device can scan the code to obtain the HTTPS URL.
   - The Companion Device transfers the HTTPS URL through a wired connection, such as a USB link to the Primary Device.

4. Using the Companion Device address and the security information obtained from the HTTPS URL, a software component (e.g. LPAd) on the Primary Device establishes a HTTPS session with the LPAd on the Companion Device:

   - Firstly, the software component uses the LPA_access_token in step 2 as the PSK to initiate the PSK-TLS connection as defined in RFC 4279 [47] with the LPAd on the Companion Device. During the TLS handshake, the software component in the Primary Device performs mutual authentication with the LPAd on the Companion Device and negotiates the session key.
   - After the TLS connection is established, the software component on the Primary Device sends an HTTP request over the TLS session to the

Companion Device to retrieve the UI presentation of the LUId. Upon receiving the HTTP request, the LPAd on the Companion Device sends the HTTP response containing the UI presentation to the Primary Device.

5. The End User uses the UI provided by the software component on the Primary Device to access the LUI on the Companion Device via the HTTPS session to perform the Local Profile Management Operations towards the eUICC in the Companion Device. The LUI on the Companion Device MAY restrict the actions that can be performed from the Primary Device. For example:

    - It MAY not offer the eUICC Memory Reset.
    - It MAY only expose the 'enable' and 'disable' operations.
    - It MAY expose a Profile for enabling only if no Profile is already enabled on the Companion Device.

C.3 General LPA Requirements

LPA functions

There SHALL be at most one instance of the LPAd per active eUICC.

The LPA SHALL support all the functions related to Profile download and Installation via the LPA's Local Profile Download (LPD) functions as defined in section 3.1.3.

The LPA SHALL support Notifications as defined in sections 3.1.3, 3.5, 5.6.4, 5.7.9, 5.7.10, 5.7.11.

All Activation Code procedures SHALL be implemented natively as part of the LPA, where the Device capabilities permit.

The LPA SHALL support the Local Profile Management Operations via LPA's Local User Interface (LUI) function as defined in sections 3.2 and 3.3:

- Initiate a Profile download RSP session with SM-DP+ as defined in section 3.1.2.
- Query for pending Profile Download Event Record as defined in section 3.6.2.
- Enabling a Disabled Profile as defined in section 3.2.1.
- Disabling an Enabled Profile as defined in section 3.2.2.
- Delete a Profile as defined in section 3.2.3.
- Query the Profile Metadata and states of Profiles installed on the eUICC as defined in section 3.2.4.
- Perform eUICC Memory Reset, as defined in section 3.3.2.
- Perform eUICC Test Memory Reset, as defined in section 3.3.3.
- Set/Edit Profile Nicknames associated with installed Profiles as defined in section 3.2.6.

The LPA SHALL support retrieval of eUICC Information as defined in section 4.3.

The LPAd SHOULD advise the End User when it determines that a Profile Management Operation or Event Retrieval operation would fail (or has failed) because connectivity to the SM-DP+ or SM-DS is not available, or an error occurs. The LPA MAY retry for a period of time as appropriate. The specific means by which the connectivity failure is detected, and

the manner in which it is communicated to the End User, are out of scope of this specification.

LPAd Functions and Security Protection

The specific mechanisms for securing the operation of the LPAd, ensuring its integrity, and ensuring the privacy and integrity of the data it handles are out of scope of this specification. As appropriate for the class of Device, the proper security level associated with LPAd functions SHOULD be ensured based on industry-proven implementations of:

- A secure boot OS.
- An implementation-dependent software/hardware secure execution environment for capturing, storing and verifying the passcode or biometric input.
- Verification of proper OEM signature of LPAd related software components.
- Application-level secure pairing and un-pairing methods between Primary and Companion Devices. This MAY be independent of pairing technologies and associated link layer security (e.g. Bluetooth or Wi-Fi).

The OEM-specific security implementation SHALL:

- Verify the integrity of the LPAd and authorise it to be used.
- Provide access to the trusted LUId user interface only for the authorised LPAd.
- Provide access to the ISD-R of the eUICC only for the authorised LPAd.
- Restrict access to the LPAd to only those applications and services that are provided by the OEM to enable the services and functions of the LPAd.
- Protect the LPAd and the data it handles from unauthorised access and modification. Such data includes, but is not limited to, the EID, Activation Code, Confirmation Code, End User credentials for Authenticated Confirmation, Profile Metadata, Profile Download and Notification payloads, and Event Records.

Depending on the device class, Devices SHALL implement protection mechanisms as shown in the table below.

Device class	Description	Example of Devices
Advanced	Devices with an open operating system where mechanisms such as secure boot and platform signing of applications are available and used to protect the LPA.	Smartphones, Tablets, Laptops, Advanced Wearables
Basic	Devices without possibility to install applications. The attack surface of the LPA is minimal due to the locked down nature of these Devices. Simple mechanisms to ensure that the LPA is not compromised SHALL be taken.	Connected sensors, Simple Wearables, Single use case devices

Table 58a: Device Classes

The Device SHALL provide mechanisms to obtain Authenticated Confirmation and Simple Confirmation. There SHALL be some means by which the End User can configure a personalised credential for Authenticated Confirmation.

The mechanism for User Intent verification is out of scope for this release.

As examples, the recommended User Intent verification could include:

- Biometric (e.g. fingerprint) verification, or
- Device passcode verification, or
- Hard-wired End User input that bypasses Device application processor.

Device Test Mode

The Device and LPAd MAY support Device Test Mode. The method of entering Device Test Mode, exiting Device Test Mode, and Device testing functionality that is not related to Remote SIM Provisioning are implementation-specific and out of the scope of this specification.

The LPAd SHALL only provide access to Test Profiles when the Device is operating in Device Test Mode.

When the Device exits Device Test Mode, the LPAd SHALL disable any enabled Test Profile as defined in section 3.2.2.

C.4 Support for CAT Mechanisms

Dependent on the deployment, the Devices SHALL support at least the CAT mechanisms (ETSI TS 102 223 [31]) indicated in the table below.

CAT mechanism	LPAd	LPAe with LUIe based on CAT	LPAe with LUIe based on SCWS
TERMINAL PROFILE	X	X	X
SETUP MENU ENVELOPE (MENU SELECTION) DISPLAY TEXT GET INKEY GET INPUT PLAY TONE SELECT ITEM EVENT DOWNLOAD - User activity EVENT DOWNLOAD - Idle screen available		X	
SET UP EVENT LIST	X	X	X
REFRESH with UICC Reset or eUICC Profile Switch mode	X	X	X

PROVIDE LOCAL INFORMATION (IMEI)		X	X
SEND SHORT MESSAGE ENVELOPE (SMS-PP DOWNLOAD)	X	X	X
TIMER MANAGEMENT ENVELOPE (TIMER EXPIRATION)		X	X
OPEN CHANNEL related to packet data service bearer	X	X	X (1)
OPEN CHANNEL related to UICC Server Mode			X (1)
CLOSE CHANNEL RECEIVE DATA SEND DATA GET CHANNEL STATUS EVENT DOWNLOAD - Data available EVENT DOWNLOAD - Channel status	X	X	X
NOTE 1: The Device SHALL support running these 2 BIP channels in parallel.			

Table 59: CAT Mechanisms

NOTE: The table also includes requirements for ES6.

Annex D Coding of the AIDs for 'Remote SIM Provisioning' (Normative)

The Coding of the AID for ISD-R, ISD-P and ECASD SHALL be as defined in SGP.02 [2].

Annex E List of Identifiers (Informative)

OIDs

The following identifiers for remote provisioning are created under a dedicated OID tree under ISO branch:

- ASN.1 notation: {ISO(1) identified-organization(3) dod(6) internet(1) private(4) enterprise(1)}
- dot notation: 1.3.6.1.4.1
- IOD-IRI notation: /ISO/Identified-Organization/6/1/4/1

The private enterprise numbers may be found under the Internet Assigned Numbers Authority: http://www.iana.org/assignments/enterprise-numbers/enterprise-numbers

EUM Identifiers

Identifier	Uniqueness	Registration Entity
EUM OID	within the ecosystem	ISO 1.3.6.1.4.1
SIN	within the ecosystem	ISO 7812 [37]

Table 60: EUM Identifiers

eUICC Identifiers

Identifier	Uniqueness	Registration Entity
EID	within the ecosystem	GSMA ESIM Technical Specification SGP.02 [2]
ECASD AID	within the eUICC	GSMA ESIM Technical Specification SGP.02 [2]
ISD-R AID	within the eUICC	GSMA ESIM Technical Specification SGP.02 [2]
ISD-P AID	within the eUICC	eUICC within a range defined in GSMA ESIM Technical Specification SGP.02 [2]
ICCID	Global	ITU-T E.118 [21]
ISD-R TAR	within the eUICC	GSMA ESIM Technical Specification SGP.02 [2]
MNO-SD AID	Within the Profile	ETSI TS 101 220 [33]
MNO-SD TAR	Within the Profile	ETSI TS 101 220 (ISD TAR) []

Table 61: eUICC Identifiers

SM-DP+ Identifier

Identifier	Uniqueness	Registration Entity
SM-DP+ OID	within the ecosystem	ISO 1.3.6.1.4.1

Table 62: SM-DP+ Identifier

SM-DS Identifier

Identifier	Uniqueness	Registration Entity
SM-DS OID	within the ecosystem	ISO 1.3.6.1.4.1

Table 63: SM-DS Identifier

MNO Identifiers

Identifier	Uniqueness	Registration Entity
MNO OID	within the ecosystem	ISO 1.3.6.1.4.1
MCC+MNC (IMSI)	Global	ITU-T for MCC and National Regulators for MNC

Table 64: MNO Identifiers

Annex F Profile Eligibility Check (Informative)

Prior to any Profile download, the Operator or the SM-DP+ verifies if the selected Profile Type is compatible with the targeted Device.

Two types of checking are possible:

- Static eligibility check (SEC): a check based on the static capabilities of the Device and / or the eUICC. These capabilities could be retrieved based on the knowledge of the EID and the TAC. These eUICC capabilities MAY be acquired by various means: information contained in the EID itself, additional tables locally handled by the Operator or communication with an external entity like the EUM. Device capabilities can be retrieved by the Operator based on the TAC. This Static eligibility check is under the responsibility of the Operator; it MAY be done by the SM-DP+ on behalf of the Operator. The means to establish the compatibility of the Profile Type with a Device type and eUICC type is out of scope of this specification.
- Dynamic eligibility check (DEC): a check based on the eUICC Info and / or the Device capabilities signed by the eUICC during Profile Download and Installation procedure. This Dynamic eligibility check is under the responsibility of the SM-DP+ on behalf of the Operator.

The following figure : Eligibility Check" describes the global eligibility process depending on the knowledge of the target Device.

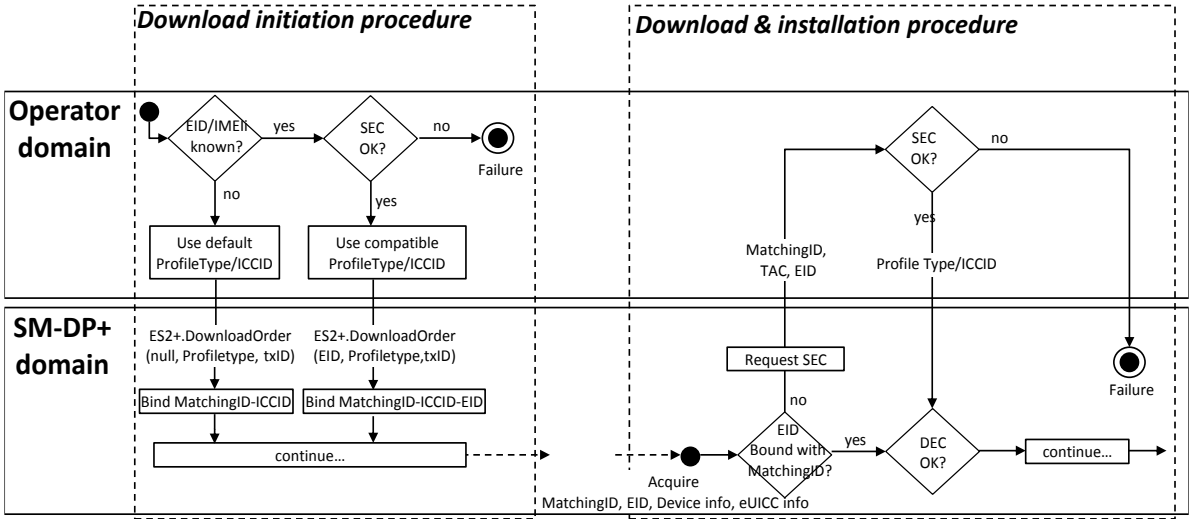


Figure 41: Eligibility Check

Annex G Key Derivation Process (Normative)

The key derivation process described in this section SHALL be executed by both the off-card entity (SM-DP+) and the eUICC in a symmetric way.

- Use otPK.eUICC.ECKA and otPK.DP.ECKA (with their respective one time private keys) to generate the shared secret ShS as described in GlobalPlatform Card Specification Amendment F [13] section 3.1.1 (but limited to ephemeral keys) which constitutes the input for the Key Derivation process.
- Concatenate the following values as SharedInfo as input for the Key Derivation process (this data is the one given as input data in the function "ES8+.InitialiseSecureChannel"):
  - Key type (1 byte)
  - Key length (1 byte)
  - HostID-LV and EID-LV. HostID-LV comprises the length and the value field of the HostID given in the input data; EID-LV comprises the length and value field of the EID.
- Initial MAC Chaining value, S-ENC and S-MAC are taken from KeyData derived from the ShS as defined in BSI TR-03111 [41] for the "X9.63 Key Derivation Function" (SHA-256 SHALL be used for the key derivation to calculate KeyData of sufficient length). This key derivation includes additional information, the 'SharedInfo' of the key derivation algorithm. Keys are assigned as defined in the following table:

KeyData	Key
1 to L	Initial MAC chaining value
L+1 to 2L	S-ENC
2L+1 to 3L	S-MAC

Table 65: Mac Chaining

The initial MAC chaining value is used for the computation of the MAC of the first SCP03t block following the "ES8+.InitialiseSecureChannel" command.

Annex H ASN.1 Definitions (Normative)

```
RSPDefinitions {joint-iso-itu-t(2) international-organizations(23) gsma(146) rsp(1)
spec-version(1) version-two(2)}
DEFINITIONS
AUTOMATIC TAGS
EXTENSIBILITY IMPLIED ::=
BEGIN

IMPORTS Certificate, CertificateList, Time 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)};

id-rsp OBJECT IDENTIFIER ::= {joint-iso-itu-t(2) international-organizations(23)
gsma(146) rsp(1)}

-- Basic types, for size constraints
Octet8 ::= OCTET STRING (SIZE(8))
Octet4 ::= OCTET STRING (SIZE(4))
Octet16 ::= OCTET STRING (SIZE(16))
OctetTo16 ::= OCTET STRING (SIZE(1..16))
Octet32 ::= OCTET STRING (SIZE(32))
Octet1 ::= OCTET STRING(SIZE(1))
Octet2 ::= OCTET STRING (SIZE(2))
VersionType ::= OCTET STRING(SIZE(3)) -- major/minor/revision version are coded as
binary value on byte 1/2/3, e.g. '02 00 0C' for v2.0.12.
-- If revision is not used (e.g. v2.1), byte 3 SHALL be set to '00'.
Iccid ::= [APPLICATION 26] OCTET STRING (SIZE(10)) -- ICCID as coded in EFiccid,
corresponding tag is '5A'
RemoteOpId ::= [2] INTEGER {installBoundProfilePackage(1)}
TransactionId ::= OCTET STRING (SIZE(1..16))

-- Definition of EUICCInfo1 ---------------------------
GetEuiccInfo1Request ::= [32] SEQUENCE { -- Tag 'BF20'
}

EUICCInfo1 ::= [32] SEQUENCE { -- Tag 'BF20'
   svn [2] VersionType,    -- GSMA SGP.22 version supported (SVN)
   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
}

-- Definition of EUICCInfo2 ---------------------------
GetEuiccInfo2Request ::= [34] SEQUENCE { -- Tag 'BF22'
}

EUICCInfo2 ::= [34] SEQUENCE { -- Tag 'BF22'
   profileVersion [1] VersionType,      -- SIMAlliance Profile package version
supported
   svn [2] VersionType,    -- GSMA SGP.22 version supported (SVN)
   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
   sasAcreditationNumber UTF8String (SIZE(0..64)),
   certificationDataObject [12] CertificationDataObject OPTIONAL
}

-- Definition of RspCapability
RspCapability ::= BIT STRING {
   additionalProfile(0), -- at least one more Profile can be installed
   crlSupport(1), -- CRL
   rpmSupport(2), -- Remote Profile Management
   testProfileSupport (3) -- support for test profile
}

-- Definition of CertificationDataObject
CertificationDataObject ::= SEQUENCE {
   platformLabel UTF8String,      -- Platform_Label as defined in GlobalPlatform
DLOA specification [57]
   discoveryBaseURL UTF8String    -- Discovery Base URL of the SE default DLOA
Registrar as defined in GlobalPlatform DLOA specification [57]
}

CertificateInfo ::= BIT STRING {

   reserved(0), -- eUICC has a CERT.EUICC.ECDSA in GlobalPlatform format. The use
of this bit is deprecated.
   certSigningX509(1),     -- eUICC has a CERT.EUICC.ECDSA in X.509 format
   rfu2(2),
   rfu3(3),
   reserved2(4), -- Handling of Certificate in GlobalPlatform format. The use of
this bit is deprecated.
   certVerificationX509(5)-- Handling of Certificate in X.509 format
}

-- Definition of UICCCapability
UICCCapability ::= BIT STRING {
/* Sequence is derived from ServicesList[] defined in SIMalliance PEDefinitions*/
   contactlessSupport(0),  -- Contactless (SWP, HCI and associated APIs)
   usimSupport(1),         -- USIM as defined by 3GPP
   isimSupport(2),         -- ISIM as defined by 3GPP
   csimSupport(3),         -- CSIM as defined by 3GPP2

   akaMilenage(4),         -- Milenage as AKA algorithm
   akaCave(5),             -- CAVE as authentication algorithm
   akaTuak128(6),          -- TUAK as AKA algorithm with 128 bit key length
   akaTuak256(7),          -- TUAK as AKA algorithm with 256 bit key length
   rfu1(8),                -- reserved for further algorithms
   rfu2(9),                -- reserved for further algorithms

   gbaAuthenUsim(10),      -- GBA authentication in the context of USIM
   gbaAuthenISim(11),      -- GBA authentication in the context of ISIM
   mbmsAuthenUsim(12),     -- MBMS authentication in the context of USIM
   eapClient(13),          -- EAP client

   javacard(14),           -- Javacard support
   multos(15),             -- Multos support

   multipleUsimSupport(16), -- Multiple USIM applications are supported within the
same Profile
   multipleIsimSupport(17),  -- Multiple ISIM applications are supported within the
same Profile
   multipleCsimSupport(18)   -- Multiple CSIM applications are supported within the
same Profile
}
```

```
-- Definition of DeviceInfo
DeviceInfo ::= SEQUENCE {
   tac Octet4,
   deviceCapabilities DeviceCapabilities,
   imei Octet8 OPTIONAL
}

DeviceCapabilities ::= SEQUENCE { -- Highest fully supported release for each
definition
  -- The device SHALL set all the capabilities it supports
   gsmSupportedRelease VersionType OPTIONAL,
   utranSupportedRelease VersionType OPTIONAL,
   cdma2000onexSupportedRelease VersionType OPTIONAL,
   cdma2000hrpdSupportedRelease VersionType OPTIONAL,
   cdma2000ehrpdSupportedRelease VersionType OPTIONAL,
   eutranSupportedRelease VersionType OPTIONAL,
   contactlessSupportedRelease VersionType OPTIONAL,
   rspCrlSupportedVersion VersionType OPTIONAL
}

ProfileInfoListRequest ::= [45] SEQUENCE { -- Tag 'BF2D'
   searchCriteria [0] CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID of the ISD-P, tag '4F'
      iccid Iccid, -- ICCID, tag '5A'
      profileClass [21] ProfileClass -- Tag '95'
   } OPTIONAL,
   tagList [APPLICATION 28] OCTET STRING OPTIONAL -- tag '5C'
}

-- Definition of ProfileInfoList
ProfileInfoListResponse ::= [45] CHOICE { -- Tag 'BF2D'
   profileInfoListOk SEQUENCE OF ProfileInfo,
   profileInfoListError ProfileInfoListError
}

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
ES10c:GetProfilesInfo
   profileClass [21] ProfileClass DEFAULT operational, -- 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'
}

PprIds ::= BIT STRING {-- Definition of Profile Policy Rules identifiers
   pprUpdateControl(0), -- defines how to update PPRs via ES6
   ppr1(1), -- Indicator for PPR1 'Disabling of this Profile is not allowed'
   ppr2(2) -- Indicator for PPR2 'Deletion of this Profile is not allowed'
}

OperatorId ::= SEQUENCE {
   mccMnc OCTET STRING (SIZE(3)), -- MCC and MNC coded as defined in 3GPP TS 24.008
[32]
   gid1 OCTET STRING OPTIONAL, -- referring to content of EF GID1 (file identifier
'6F3E') as defined in 3GPP TS 31.102 [54]
   gid2 OCTET STRING OPTIONAL -- referring to content of EF GID2 (file identifier
'6F3F') as defined in 3GPP TS 31.102 [54]
}
```

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

-- Definition of StoreMetadata request

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'
}

NotificationEvent ::= BIT STRING {
   notificationInstall(0),
   notificationEnable(1),
   notificationDisable(2),
   notificationDelete(3)
}

NotificationConfigurationInformation ::= SEQUENCE {
   profileManagementOperation NotificationEvent,
   notificationAddress UTF8String -- FQDN to forward the notification
}

IconType ::= INTEGER {jpg(0), png(1)}
ProfileState ::= INTEGER {disabled(0), enabled(1)}
ProfileClass ::= INTEGER {test(0), provisioning(1), operational(2)}

-- Definition of UpdateMetadata request
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'
}

-- Definition of data objects for command PrepareDownload -------------------------
PrepareDownloadRequest ::= [33] SEQUENCE {  -- Tag 'BF21'
   smdpSigned2 SmdpSigned2,                    -- Signed information
   smdpSignature2 [APPLICATION 55] OCTET STRING,      -- DP_Sign1, tag '5F37'
   hashCc Octet32 OPTIONAL, -- Hash of confirmation code
   smdpCertificate Certificate    -- CERT.DPpb.ECDSA
}

SmdpSigned2 ::= SEQUENCE {
   transactionId [0] TransactionId,     -- The TransactionID generated by the
SM-DP+
   ccRequiredFlag BOOLEAN, --Indicates if the Confirmation Code is required
   bppEuiccOtpk [APPLICATION 73] OCTET STRING OPTIONAL      -- otPK.EUICC.ECKA
already used for binding the BPP, tag '5F49'
}

PrepareDownloadResponse ::= [33] CHOICE {  -- Tag 'BF21'
   downloadResponseOk PrepareDownloadResponseOk,
   downloadResponseError PrepareDownloadResponseError
}

PrepareDownloadResponseOk ::= SEQUENCE {
   euiccSigned2 EUICCSigned2,     -- Signed information
```

```
   euiccSignature2 [APPLICATION 55] OCTET STRING      -- tag '5F37'
}

EUICCSigned2 ::= SEQUENCE {
   transactionId [0] TransactionId,
   euiccOtpk [APPLICATION 73] OCTET STRING,           -- otPK.EUICC.ECKA, tag '5F49'
   hashCc Octet32 OPTIONAL        -- Hash of confirmation code
}

PrepareDownloadResponseError ::= SEQUENCE {
   transactionId [0] TransactionId,
   downloadErrorCode DownloadErrorCode
}

DownloadErrorCode ::= INTEGER {invalidCertificate(1), invalidSignature(2),
unsupportedCurve(3), noSessionContext(4), invalidTransactionId(5),
undefinedError(127)}

-- Definition of data objects for command AuthenticateServer--------------------
AuthenticateServerRequest ::= [56] SEQUENCE { -- Tag 'BF38'
   serverSigned1 ServerSigned1,                       -- Signed information
   serverSignature1 [APPLICATION 55] OCTET STRING,    -- tag '5F37'
   euiccCiPKIdToBeUsed SubjectKeyIdentifier,          -- CI Public Key Identifier to
be used
   serverCertificate Certificate, -- RSP Server Certificate CERT.XXauth.ECDSA
   ctxParams1 CtxParams1
}

ServerSigned1 ::= SEQUENCE {
   transactionId [0] TransactionId,            -- The Transaction ID generated by
the RSP Server
   euiccChallenge [1] Octet16,           -- The eUICC Challenge
   serverAddress [3] UTF8String,  -- The RSP Server address
   serverChallenge [4] Octet16           -- The RSP Server Challenge
}

CtxParams1 ::= CHOICE {
   ctxParamsForCommonAuthentication CtxParamsForCommonAuthentication -- New
contextual data objects MAY be defined for extensibility
}

CtxParamsForCommonAuthentication ::= SEQUENCE {
   matchingId UTF8String OPTIONAL,-- The MatchingId could be the Activation code
token or EventID or empty
   deviceInfo DeviceInfo -- The Device information
}

AuthenticateServerResponse ::= [56] CHOICE { -- Tag 'BF38'
   authenticateResponseOk AuthenticateResponseOk,
   authenticateResponseError AuthenticateResponseError
}

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
}

EuiccSigned1 ::= SEQUENCE {
   transactionId [0] TransactionId,
   serverAddress [3] UTF8String,
   serverChallenge [4] Octet16,   -- The RSP Server Challenge
   euiccInfo2 [34] EUICCInfo2,
   ctxParams1 CtxParams1
}
```

```
AuthenticateResponseError ::= SEQUENCE {
   transactionId [0] TransactionId,
   authenticateErrorCode AuthenticateErrorCode
}

AuthenticateErrorCode ::= INTEGER {invalidCertificate(1), invalidSignature(2),
unsupportedCurve(3), noSessionContext(4), invalidOid(5), euiccChallengeMismatch(6),
ciPKUnknown(7), undefinedError(127)}

-- Definition of Cancel Session------------------------------
CancelSessionRequest ::= [65] SEQUENCE { -- Tag 'BF41'
   transactionId TransactionId,    -- The TransactionID generated by the RSP Server
   reason CancelSessionReason
}

CancelSessionReason ::= INTEGER {endUserRejection(0), postponed(1), timeout(2),
pprNotAllowed(3), metadataMismatch(4), loadBppExecutionError(5),
undefinedReason(127)}

CancelSessionResponse ::= [65] CHOICE { -- Tag 'BF41'
   cancelSessionResponseOk CancelSessionResponseOk,
   cancelSessionResponseError INTEGER {invalidTransactionId(5),
undefinedError(127)}
}

CancelSessionResponseOk ::= SEQUENCE {
   euiccCancelSessionSigned EuiccCancelSessionSigned,       -- Signed information
   euiccCancelSessionSignature [APPLICATION 55] OCTET STRING-- tag '5F37
}

EuiccCancelSessionSigned ::= SEQUENCE {
   transactionId TransactionId,
   smdpOid OBJECT IDENTIFIER, -- SM-DP+ OID as contained in CERT.DPauth.ECDSA
   reason CancelSessionReason
}

-- Definition of Bound Profile Package --------------------------
BoundProfilePackage ::= [54] SEQUENCE { -- Tag 'BF36'
   initialiseSecureChannelRequest [35] InitialiseSecureChannelRequest, -- Tag
'BF23'
   firstSequenceOf87 [0] SEQUENCE OF [7] OCTET STRING, -- sequence of '87' TLVs
   sequenceOf88 [1] SEQUENCE OF [8] OCTET STRING, -- sequence of '88' TLVs
   secondSequenceOf87 [2] SEQUENCE OF [7] OCTET STRING OPTIONAL, -- sequence of
'87' TLVs
   sequenceOf86 [3] SEQUENCE OF [6] OCTET STRING -- sequence of '86' TLVs
}

-- Definition of Get eUICC Challenge ---------------------------
GetEuiccChallengeRequest ::= [46] SEQUENCE { -- Tag 'BF2E'
}

GetEuiccChallengeResponse ::= [46] SEQUENCE { -- Tag 'BF2E'
   euiccChallenge Octet16  -- random eUICC challenge
}

-- Definition of Profile Installation Result
ProfileInstallationResult ::= [55] SEQUENCE { -- Tag 'BF37'
   profileInstallationResultData [39] ProfileInstallationResultData,
   euiccSignPIR EuiccSignPIR
}

ProfileInstallationResultData ::= [39] SEQUENCE { -- Tag 'BF27'
   transactionId[0] TransactionId, -- The TransactionID generated by the SM-DP+
   notificationMetadata[47] NotificationMetadata,
   smdpOid OBJECT IDENTIFIER, -- SM-DP+ OID (same value as in CERT.DPpb.ECDSA)
   finalResult [2] CHOICE {
      successResult SuccessResult,
```

```
      errorResult ErrorResult
   }
}

EuiccSignPIR ::= [APPLICATION 55] OCTET STRING -- Tag '5F37', eUICC's signature

SuccessResult ::= SEQUENCE {
   aid [APPLICATION 15] OCTET STRING (SIZE (5..16)), -- AID of ISD-P
   simaResponse OCTET STRING -- contains (multiple) 'EUICCResponse' as defined in
[5]
}

ErrorResult ::= SEQUENCE {
   bppCommandId BppCommandId,
   errorReason ErrorReason,
   simaResponse OCTET STRING OPTIONAL -- contains (multiple) 'EUICCResponse' as
defined in [5]
}

BppCommandId ::= INTEGER {initialiseSecureChannel(0), configureISDP(1),
storeMetadata(2), storeMetadata2(3), replaceSessionKeys(4), loadProfileElements(5)}

ErrorReason ::= INTEGER {
   incorrectInputValues(1),
   invalidSignature(2),
   invalidTransactionId(3),
   unsupportedCrtValues(4),
   unsupportedRemoteOperationType(5),
   unsupportedProfileClass(6),
   scp03tStructureError(7),
   scp03tSecurityError(8),
   installFailedDueToIccidAlreadyExistsOnEuicc(9),
   installFailedDueToInsufficientMemoryForProfile(10),
   installFailedDueToInterruption(11),
   installFailedDueToPEProcessingError (12),
   installFailedDueToDataMismatch(13),
   testProfileInstallFailedDueToInvalidNaaKey(14),
   pprNotAllowed(15),
   installFailedDueToUnknownError(127)
}

ListNotificationRequest ::= [40] SEQUENCE { -- Tag 'BF28'
   profileManagementOperation [1] NotificationEvent OPTIONAL
}

ListNotificationResponse ::= [40] CHOICE { -- Tag 'BF28'
   notificationMetadataList SEQUENCE OF NotificationMetadata,
   listNotificationsResultError INTEGER {undefinedError(127)}
}

NotificationMetadata ::= [47] SEQUENCE { -- Tag 'BF2F'
   seqNumber [0] INTEGER,
   profileManagementOperation [1] NotificationEvent, /*Only one bit SHALL be set to
1*/
   notificationAddress UTF8String, -- FQDN to forward the notification
   iccid Iccid OPTIONAL
}

-- Definition of Profile Nickname Information
SetNicknameRequest ::= [41] SEQUENCE {  -- Tag 'BF29'
   iccid Iccid,
   profileNickname [16] UTF8String (SIZE(0..64))
}

SetNicknameResponse ::= [41] SEQUENCE { -- Tag 'BF29'
   setNicknameResult INTEGER {ok(0), iccidNotFound (1), undefinedError(127)}
}
```

```
id-rsp-cert-objects OBJECT IDENTIFIER ::= {  id-rsp cert-objects(2)}

id-rspExt OBJECT IDENTIFIER ::= {id-rsp-cert-objects 0}

id-rspRole OBJECT IDENTIFIER ::= {id-rsp-cert-objects 1}

-- Definition of OIDs for role identification
id-rspRole-ci OBJECT IDENTIFIER ::= {id-rspRole 0}
id-rspRole-euicc OBJECT IDENTIFIER ::= {id-rspRole 1}
id-rspRole-eum OBJECT IDENTIFIER ::= {id-rspRole 2}
id-rspRole-dp-tls OBJECT IDENTIFIER ::= {id-rspRole 3}
id-rspRole-dp-auth OBJECT IDENTIFIER ::= {id-rspRole 4}
id-rspRole-dp-pb OBJECT IDENTIFIER ::= {id-rspRole 5}
id-rspRole-ds-tls OBJECT IDENTIFIER ::= {id-rspRole 6}
id-rspRole-ds-auth OBJECT IDENTIFIER ::= {id-rspRole 7}

--Definition of data objects for InitialiseSecureChannel Request
InitialiseSecureChannelRequest ::= [35] SEQUENCE { -- Tag 'BF23'
   remoteOpId RemoteOpId, -- Remote Operation Type Identifier (value SHALL be set
to installBoundProfilePackage)
   transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
   controlRefTemplate[6] IMPLICIT ControlRefTemplate, -- Control Reference Template
(Key Agreement). Current specification considers a subset of CRT specified in
GlobalPlatform Card Specification [8], section 6.4.2.3 for the Mutual
Authentication Data Field
   smdpOtpk [APPLICATION 73] OCTET STRING, ---otPK.DP.ECKA as specified in
GlobalPlatform Card Specification [8] section 6.4.2.3 for ePK.OCE.ECKA, tag '5F49'
   smdpSign [APPLICATION 55] OCTET STRING -- SM-DP's signature, tag '5F37'
}

ControlRefTemplate ::= SEQUENCE {
   keyType[0] Octet1, -- Key type according to GlobalPlatform Card Specification
[8] Table 11-16, AES= '88', Tag '80'
   keyLen[1] Octet1, --Key length in number of bytes. For current specification key
length SHALL by 0x10 bytes, Tag '81'
   hostId[4] OctetTo16 -- Host ID value , Tag '84'
}

--Definition of data objects for ConfigureISDPRequest
ConfigureISDPRequest ::= [36] SEQUENCE { -- Tag 'BF24'
   dpProprietaryData [24] DpProprietaryData OPTIONAL -- Tag 'B8'
}

DpProprietaryData ::= SEQUENCE { --  maximum size including tag and length field:
128 bytes
   dpOid OBJECT IDENTIFIER -- OID in the tree of the SM-DP+ that created the
Profile
   -- additional data objects defined by the SM-DP+ MAY follow
}

-- Definition of request message for command ReplaceSessionKeys
ReplaceSessionKeysRequest ::= [38] SEQUENCE { -- tag 'BF26'
/*The new initial MAC chaining value*/
   initialMacChainingValue OCTET STRING,
/*New session key value for encryption/decryption (PPK-ENC)*/
   ppkEnc OCTET STRING,
/*New session key value of the session key C-MAC computation/verification (PPK-
MAC)*/
   ppkCmac OCTET STRING
}

-- Definition of data objects for RetrieveNotificationsList
RetrieveNotificationsListRequest ::= [43] SEQUENCE { -- Tag 'BF2B'
   searchCriteria CHOICE {
      seqNumber [0] INTEGER,
      profileManagementOperation [1] NotificationEvent
   } OPTIONAL
}
```

```
RetrieveNotificationsListResponse ::= [43] CHOICE { -- Tag 'BF2B'
   notificationList SEQUENCE OF PendingNotification,
   notificationsListResultError INTEGER { undefinedError(127)}
}

PendingNotification ::= CHOICE {
   profileInstallationResult [55] ProfileInstallationResult, -- tag 'BF37'
   otherSignedNotification OtherSignedNotification
}

OtherSignedNotification ::= SEQUENCE {
   tbsOtherNotification NotificationMetadata,
   euiccNotificationSignature [APPLICATION 55] OCTET STRING, -- eUICC signature of
tbsOtherNotification, 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
}

-- Definition of notificationSent
NotificationSentRequest ::= [48] SEQUENCE { -- Tag 'BF30'
   seqNumber [0] INTEGER
}

NotificationSentResponse ::= [48] SEQUENCE { -- Tag 'BF30'
   deleteNotificationStatus INTEGER {ok(0), nothingToDelete(1),
undefinedError(127)}
}

-- Definition of Enable Profile -------------------------
EnableProfileRequest ::= [49] SEQUENCE { -- Tag 'BF31'
   profileIdentifier CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
      iccid Iccid -- ICCID, tag '5A'
   },
   refreshFlag BOOLEAN -- indicating whether REFRESH is required
}

EnableProfileResponse ::= [49] SEQUENCE { -- Tag 'BF31'
   enableResult INTEGER {ok(0), iccidOrAidNotFound (1),
profileNotInDisabledState(2), disallowedByPolicy(3), wrongProfileReenabling(4),
catBusy(5), undefinedError(127)}
}

-- Definition of Disable Profile -------------------------
DisableProfileRequest ::= [50] SEQUENCE { -- Tag 'BF32'
   profileIdentifier CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
      iccid Iccid -- ICCID, tag '5A'
   },
   refreshFlag BOOLEAN -- indicating whether REFRESH is required
}

DisableProfileResponse ::= [50] SEQUENCE { -- Tag 'BF32'
   disableResult INTEGER {ok(0), iccidOrAidNotFound (1),
profileNotInEnabledState(2), disallowedByPolicy(3), catBusy(5),
undefinedError(127)}
}

-- Definition of Delete Profile -------------------------
DeleteProfileRequest ::= [51] CHOICE { -- Tag 'BF33'
   isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
   iccid Iccid -- ICCID, tag '5A'
}

DeleteProfileResponse ::= [51] SEQUENCE { -- Tag 'BF33'
```

```
   deleteResult INTEGER {ok(0), iccidOrAidNotFound (1),
profileNotInDisabledState(2), disallowedByPolicy(3), undefinedError(127)}
}

-- Definition of Memory Reset -------------------------
EuiccMemoryResetRequest ::= [52] SEQUENCE { -- Tag 'BF34'
   resetOptions [2] BIT STRING {
      deleteOperationalProfiles(0),
      deleteFieldLoadedTestProfiles(1),
      resetDefaultSmdpAddress(2)}
}

EuiccMemoryResetResponse ::= [52] SEQUENCE { -- Tag 'BF34'
   resetResult INTEGER {ok(0), nothingToDelete(1), catBusy(5), undefinedError(127)}
}

-- Definition of Get EID -------------------------
GetEuiccDataRequest ::= [62] SEQUENCE { -- Tag 'BF3E'
   tagList [APPLICATION 28] Octet1  -- tag '5C', the value SHALL be set to '5A'
}

GetEuiccDataResponse ::= [62] SEQUENCE { -- Tag 'BF3E'
   eidValue [APPLICATION 26] Octet16  -- tag '5A'
}

-- Definition of Get Rat

GetRatRequest ::= [67] SEQUENCE { -- Tag ' BF43'
   -- No input data
}


GetRatResponse ::= [67] SEQUENCE { -- Tag 'BF43'
   rat RulesAuthorisationTable
}

RulesAuthorisationTable ::= SEQUENCE OF ProfilePolicyAuthorisationRule
ProfilePolicyAuthorisationRule ::= SEQUENCE {
   pprIds PprIds,
   allowedOperators SEQUENCE OF OperatorId,
   pprFlags BIT STRING {consentRequired(0)}
}

-- Definition of data structure command for loading a CRL
LoadCRLRequest ::= [53] SEQUENCE { -- Tag 'BF35'
   -- A CRL-A
   crl CertificateList
}

-- Definition of data structure response for loading a CRL
LoadCRLResponse ::= [53] CHOICE {  -- Tag 'BF35'
   loadCRLResponseOk LoadCRLResponseOk,
   loadCRLResponseError LoadCRLResponseError
}

LoadCRLResponseOk ::= SEQUENCE {
   missingParts SEQUENCE OF INTEGER OPTIONAL
}
LoadCRLResponseError ::= INTEGER {invalidSignature(1), invalidCRLFormat(2),
notEnoughMemorySpace(3), verificationKeyNotFound(4), fresherCrlAlreadyLoaded(5),
baseCrlMissing(6), undefinedError(127)}

-- Definition of the extension for Certificate Expiration Date
id-rsp-expDate OBJECT IDENTIFIER ::= {id-rspExt 1}
ExpirationDate ::= Time

-- Definition of the extension id for total partial-CRL number
id-rsp-totalPartialCrlNumber OBJECT IDENTIFIER ::= {id-rspExt 2}
```

```
TotalPartialCrlNumber ::= INTEGER


-- Definition of the extension id for the partial-CRL number
id-rsp-partialCrlNumber OBJECT IDENTIFIER ::= {id-rspExt 3}
PartialCrlNumber ::= INTEGER

-- Definition for ES9+ ASN.1 Binding --------------------------
RemoteProfileProvisioningRequest ::= [2] CHOICE { -- Tag 'A2'
   initiateAuthenticationRequest [57] InitiateAuthenticationRequest,  -- Tag 'BF39'
   authenticateClientRequest [59] AuthenticateClientRequest, -- Tag 'BF3B'
   getBoundProfilePackageRequest [58] GetBoundProfilePackageRequest,  -- Tag 'BF3A'
   cancelSessionRequestEs9 [65] CancelSessionRequestEs9, -- Tag 'BF41'
   handleNotification [61] HandleNotification -- tag 'BF3D'
}

RemoteProfileProvisioningResponse ::= [2] CHOICE { -- Tag 'A2'
   initiateAuthenticationResponse [57] InitiateAuthenticationResponse, -- Tag
'BF39'
   authenticateClientResponseEs9 [59] AuthenticateClientResponseEs9, -- Tag 'BF3B'
   getBoundProfilePackageResponse [58] GetBoundProfilePackageResponse, -- Tag
'BF3A'
   cancelSessionResponseEs9 [65] CancelSessionResponseEs9, -- Tag 'BF41'
   authenticateClientResponseEs11 [64] AuthenticateClientResponseEs11 -- Tag 'BF40'
}

InitiateAuthenticationRequest ::= [57] SEQUENCE { -- Tag 'BF39'
   euiccChallenge [1] Octet16, -- random eUICC challenge
   smdpAddress [3] UTF8String,
   euiccInfo1 EUICCInfo1
}

InitiateAuthenticationResponse ::= [57] CHOICE { -- Tag 'BF39'
   initiateAuthenticationOk InitiateAuthenticationOkEs9,
   initiateAuthenticationError INTEGER {
      invalidDpAddress(1),
      euiccVersionNotSupportedByDp(2),
      ciPKNotSupported(3)
   }
}

InitiateAuthenticationOkEs9 ::= SEQUENCE {
   transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
   serverSigned1 ServerSigned1, -- Signed information
   serverSignature1 [APPLICATION 55] OCTET STRING, -- Server_Sign1, tag '5F37'
   euiccCiPKIdToBeUsed SubjectKeyIdentifier, -- The curve CI Public Key to be used
as required by ES10b.AuthenticateServer
   serverCertificate Certificate
}

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

AuthenticateClientResponseEs9 ::= [59] CHOICE { -- Tag 'BF3B'
   authenticateClientOk AuthenticateClientOk,
   authenticateClientError INTEGER {
      eumCertificateInvalid(1),
      eumCertificateExpired(2),
      euiccCertificateInvalid(3),
      euiccCertificateExpired(4),
      euiccSignatureInvalid(5),
      matchingIdRefused(6),
      eidMismatch(7),
      noEligibleProfile(8),
      ciPKUnknown(9),
```

```
      invalidTransactionId(10),
      insufficientMemory(11),
      undefinedError(127)
   }
}

AuthenticateClientOk ::= SEQUENCE {
   transactionId [0] TransactionId,
   profileMetaData [37] StoreMetadataRequest,
   smdpSigned2 SmdpSigned2, -- Signed information
   smdpSignature2 [APPLICATION 55] OCTET STRING,      -- tag '5F37'
   smdpCertificate Certificate    -- CERT.DPpb.ECDSA
}

GetBoundProfilePackageRequest ::= [58] SEQUENCE { -- Tag 'BF3A'
   transactionId [0] TransactionId,
   prepareDownloadResponse [33] PrepareDownloadResponse
}

GetBoundProfilePackageResponse ::= [58] CHOICE { -- Tag 'BF3A'
   getBoundProfilePackageOk GetBoundProfilePackageOk,
   getBoundProfilePackageError INTEGER {
      euiccSignatureInvalid(1),
      confirmationCodeMissing(2),
      confirmationCodeRefused(3),
      confirmationCodeRetriesExceeded(4),
      invalidTransactionId(95),
      undefinedError(127)
   }
}

GetBoundProfilePackageOk ::= SEQUENCE {
   transactionId [0] TransactionId,
   boundProfilePackage [54] BoundProfilePackage
}

HandleNotification ::= [61] SEQUENCE {  -- Tag 'BF3D'
   pendingNotification PendingNotification
}

CancelSessionRequestEs9 ::= [65] SEQUENCE { -- Tag 'BF41'
   transactionId TransactionId,
   cancelSessionResponse  CancelSessionResponse -- data structure defined for
ES10b.CancelSession function
}

CancelSessionResponseEs9 ::= [65] CHOICE { -- Tag 'BF41'
   cancelSessionOk CancelSessionOk,
   cancelSessionError INTEGER {
      invalidTransactionId(1),
      euiccSignatureInvalid(2),
      undefinedError(127)
   }
}

CancelSessionOk ::= SEQUENCE { -- This function has no output data
}

EuiccConfiguredAddressesRequest ::= [60] SEQUENCE { -- Tag 'BF3C'
}

EuiccConfiguredAddressesResponse ::= [60] SEQUENCE {  -- Tag 'BF3C'
   defaultDpAddress UTF8String OPTIONAL,  -- Default SM-DP+ address as an FQDN
   rootDsAddress UTF8String  -- Root SM-DS address as an FQDN
}

ISDRProprietaryApplicationTemplate ::= [PRIVATE 0] SEQUENCE { -- Tag 'E0'
   svn [2] VersionType,    -- GSMA SGP.22 version supported (SVN)
```

```
   lpaeSupport BIT STRING {
      lpaeUsingCat(0), -- LPA in the eUICC using Card Application Toolkit
      lpaeUsingScws(1) -- LPA in the eUICC using Smartcard Web Server
   } OPTIONAL
}

LpaeActivationRequest ::= [66] SEQUENCE { -- Tag 'BF42'
   lpaeOption BIT STRING {
      activateCatBasedLpae(0), -- LPAe with LUIe based on CAT
      activateScwsBasedLpae(1) -- LPAe with LUIe based on SCWS
   }
}

LpaeActivationResponse ::= [66] SEQUENCE { -- Tag 'BF42'
   lpaeActivationResult INTEGER {ok(0), notSupported(1)}
}

SetDefaultDpAddressRequest ::= [63] SEQUENCE { -- Tag 'BF3F'
   defaultDpAddress UTF8String -- Default SM-DP+ address as an FQDN
}

SetDefaultDpAddressResponse ::= [63] SEQUENCE { -- Tag 'BF3F'
   setDefaultDpAddressResult INTEGER { ok (0), undefinedError (127)}
}

AuthenticateClientResponseEs11 ::= [64] CHOICE {  -- Tag 'BF40'
   authenticateClientOk AuthenticateClientOkEs11,
   authenticateClientError INTEGER {
      eumCertificateInvalid(1),
      eumCertificateExpired(2),
      euiccCertificateInvalid(3),
      euiccCertificateExpired(4),
      euiccSignatureInvalid(5),
      eventIdUnknown(6),
      invalidTransactionId(7),
      undefinedError(127)
   }
}

AuthenticateClientOkEs11 ::= SEQUENCE {
   transactionId TransactionId,
   eventEntries SEQUENCE OF EventEntries
}

EventEntries ::= SEQUENCE {
   eventId UTF8String,
   rspServerAddress UTF8String
}

END
```

Annex I JSON Request Response Examples (Informative)

An example for the "ES9+.InitiateAuthentication" function is shown below:

- HTTP Request (from LPA to SM-DP+):

```
HTTP POST /gsma/rsp2/es9plus/initiateAuthentication HTTP/1.1
Host: smdp.gsma.com
User-Agent: gsma-rsp-lpad
X-Admin-Protocol: gsma/rsp/v2.0.0
Content-Type: application/json
Content-Length: XXX

{
   "euiccChallenge" : "ZVVpY2NDaGFsbGVuZ2VFeGFtcGxlQmFzZTY0oUFZuQnNZVE5D",
   "euiccInfo1" : "RmVHRnRjR3hsUW1GelpUWTBvVUZadVFuTlpWRTU",
   "smdpAddress" : "smdp.gsma.com"
}
```

- HTTP Response (from LPA to SM-DP+ to)

```
HTTP/1.1 200 OK
X-Admin-Protocol: gsma/rsp/v2.0.0
Content-Type: application/json
Content-Length: XXX

{
   "header" : {
     "functionExecutionStatus" : {
        "status" : "Executed-Success"
     }
   },
   "transactionId" : "0123456789ABCDEF",
   "serverSigned1" : "RKNFZsbFVUa05qUm14e",
   "serverSignature1" : "RKNFZsbFVUa05qUm14e",
   "euiccCiPKIdTobeUsed" : "MDM=",
   "serverCertificate" : "RUU2NTQ0ODQ5NDA0RlpSRUZERA=="
}
```

An example for the "ES2+.DownloadOrder" function is shown as follows.

- HTTP Request (from Operator to SM-DP+):

```
HTTP POST /gsma/rsp2/es2plus/downloadOrder HTTP/1.1
Host: smdp.gsma.com
X-Admin-Protocol: gsma/rsp/v2.0.0
Content-Type: application/json
Content-Length: XXX

{
   "header" : {
     "functionRequesterIdentifier" : "RequesterID",
     "functionCallIdentifier" : "TX-567"
   }
   "eid" : "01020300405060708090A0B0C0D0EOF",
   "iccid" : "01234567890123456789",
   "profileType" : "myProfileType"
```

```
}
```

- HTTP Response for a successful execution:

```
HTTP/1.1 200 OK
X-Admin-Protocol: gsma/rsp/v2.0.0
Content-Type: application/json
Content-Length: XXX

{
   "header" : {
      "functionExecutionStatus" : {
         "status" : "Executed-Success"
      }
   },
   "iccid" : "01234567890123456789"
}
```

- HTTP Response for a failed execution:

```
HTTP/1.1 200 OK
X-Admin-Protocol: gsma/rsp/v2.0.0
Content-Type: application/json
Content-Length: XXX

{
   "header" : {
      "functionExecutionStatus" : {
         "status" : "Failed",
         "statusCodeData" : {
            "subjectCode" : "8.2.5",
            "reasonCode" : "3.7",
            "message" : "No more Profile"
         }
      }
   }
}
```

An example for the "ES2+.HandleDownloadProgressInfo" function is shown as follows:

- HTTP Request:

```
HTTP POST /gsma/rsp2/es2plus/handleDownloadProgressInfo HTTP/1.1
Host: smdp.gsma.com
X-Admin-Protocol: gsma/rsp/v2.0.0
Content-Type: application/json
Content-Length: XXX

{
   "header" : {
      "functionRequesterIdentifier" : "RequesterID",
      "functionCallIdentifier" : "TX-567",
   },
   "eid" : "01020300405060708090A0B0C0D0EOF",
   "iccid" : "01234567890123456789",
```

```
   "profileType" : "myProfileType",
   "timeStamp" : "2015-12-16T09:30:47Z",
   "notificationPointId" : "4"
   "notificationPointStatus" : {
      "status" : "Executed-Success"
   }
}
```

- HTTP Response for a successful execution:

```
HTTP/1.1 204 No Content
X-Admin-Protocol: gsma/rsp/v2.0.0
```

Annex J 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
'BF20'	GetEuiccInfo1Request or EUICCInfo1
'BF21'	PrepareDownloadRequest or PrepareDownloadResponse
'BF22'	GetEuiccInfo2Request or EUICCInfo2
'BF23'	InitialiseSecureChannelRequest
'BF24'	ConfigureISDPRequest
'BF25'	StoreMetadataRequest
'BF26'	ReplaceSessionKeysRequest
'BF27'	Reserved
'BF28'	ListNotificationRequest or ListNotificationResponse
'BF29'	SetNicknameRequest or SetNicknameResponse
'BF2A'	UpdateMetadataRequest
'BF2B'	PendingNotificationsListRequest or PendingNotificationsListResponse
'BF2D'	ProfileInfoListRequest or ProfileInfoListResponse
'BF2E'	GetEuiccChallengeRequest or GetEuiccChallengeResponse
'BF2F'	NotificationMetadata
'BF30'	NotificationSentRequest or NotificationSentResponse
'BF31'	EnableProfileRequest or EnableProfileResponse
'BF32'	DisableProfileRequest or DisableProfileResponse
'BF33'	DeleteProfileRequest or DeleteProfileResponse
'BF34'	EuiccMemoryResetRequest or EuiccMemoryResetResponse
'BF35'	LoadCRLRequest and LoadCRLResponse
'BF36'	BoundProfilePackage
'BF37'	ProfileInstallationResult
'BF38'	AuthenticateServerRequest or AuthenticateServerResponse
'BF39'	InitiateAuthenticationRequest or InitiateAuthenticationResponse
'BF3A'	GetBoundProfilePackageRequest or GetBoundProfilePackageResponse
'BF3B'	AuthenticateClientRequest or AuthenticateClientResponse
'BF3C'	EuiccConfiguredAddressesRequest or EuiccConfiguredAddressesResponse
'BF3D'	handleNotification
'BF3E'	GetEuiccDataRequest or GetEuiccDataResponse
'BF3F'	SetDefaultDpAddressRequest or SetDefaultDpAddressResponse
'BF40'	AuthenticateClientResponseEs11
'BF41'	CancelSessionRequest or CancelSessionResponse or cancelSessionRequestEs9 or cancelSessionResponseEs9
'BF42'	LpaeActivationRequest or LpaeActivationResponse

Tag	Data name
'BF43'	GetRatRequest or GetRatResponse
'E3'	ProfileInfo

Table 66: Tag Allocation

Annex K OID allocation (Informative)

This annex provides some background on the schema of the OID allocation used in this document.

For the purpose of assigning OIDs, a root OID for GSMA was registered within the RSP project.

The value of this root OID is:

joint-iso-itu-t(2) international-organizations(23) gsma(146)

For the purpose of this project, a first node was allocated under this node:

rsp(1)

All OIDs allocated in this version and in version 1.X of this specification belong to the rsp node.

At the time of writing of this specification, Mr. Gary Waite (GSMA) was responsible for the allocation of additional OIDs under the gsma node, i.e. acting as Registration Authority.

Other GSMA projects should use a similar approach: register a project specific node under the gsma node and then define sub-nodes in the project specific documentation.

Within the rsp node, the following schema is used:

rsp(1) – root for the RSP project

spec-version(1) – root for identifying the ASN.1 module of the different versions

version-one(1) – ASN.1 module of version 1.X

version-two(2) – ASN.1 module of version 2.X

… - future ASN.1 modules SHOULD use additional sub-nodes here

cert-objects(2) – root for nodes identifying objects and roles used in certificates

id-rspExt(0) – root for certificate extensions defined in this specification (see Annex H for further details)

id-rspRole(1) – root for roles used in certificates (see Annex H for further details)

Annex L Document Management (Informative)

L.1 Document History

Version	Date	Brief Description of Change	Approval Authority	Editor / Company
V1.0	13 January 2016	New PRD Publication	PSG	Duncan Macadam GSMA
V.1.1	14 April 2016	Minor Change to fix bugs Phase one	PSG	Yolanda Sanz GSMA
V2.0	July 2016	Major Change to include new features	PSG	Yolanda Sanz GSMA
V2.1	Feb 2017	Minor Change to fix bugs and maintenance changes	PSG	Yolanda Sanz GSMA

Other Information

Type	Description
Document Owner	Yolanda Sanz / GSMA
Editor / Company	Denis Praca / Gemalto

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

Your comments or suggestions & questions are always welcome.