

SGP.33-1 eSIM IoT Test Specification for the eUICC

Version 1.0

26 January 2024

Security Classification: Non-confidential

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

Copyright Notice

Copyright © 2024 GSM Association

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.

Compliance Notice

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

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

Table of Contents

[1 Introduction 5](#_Toc157170366)

[1.1 Overview 5](#_Toc157170367)

[1.2 Scope 5](#_Toc157170368)

[1.3 Definition of Terms 5](#_Toc157170369)

[1.4 Abbreviations 6](#_Toc157170370)

[1.5 Document Cross-references 6](#_Toc157170371)

[1.6 Conventions 8](#_Toc157170372)

[2 Testing Rules 8](#_Toc157170373)

[2.1 Applicability 8](#_Toc157170374)

[2.1.1 Format of the Optional Features Table 8](#_Toc157170375)

[2.1.2 Format of the Applicability Table 8](#_Toc157170376)

[2.1.3 Applicability and Notations 9](#_Toc157170377)

[2.1.4 Optional Features Table 9](#_Toc157170378)

[2.1.5 Applicability Table 9](#_Toc157170379)

[2.2 General Consideration 14](#_Toc157170380)

[2.2.1 Test Case Definition 14](#_Toc157170381)

[2.2.2 Test Cases Format 14](#_Toc157170382)

[2.2.3 General Rules for eUICC Testing 19](#_Toc157170383)

[3 Testing Architecture 20](#_Toc157170384)

[3.1 Testing Scope 20](#_Toc157170385)

[3.2 Testing Execution 22](#_Toc157170386)

[3.2.1 eUICC - Test Environment 23](#_Toc157170387)

[3.2.5 Integrated eUICC – Test Environment 23](#_Toc157170388)

[4 Interface Compliance Testing 24](#_Toc157170389)

[4.1 General Overview 24](#_Toc157170390)

[4.2 eUICC Interfaces 24](#_Toc157170391)

[4.2.1 ATR and ISD-R Selection 24](#_Toc157170392)

[4.2.2 ES6 (Operator -- eUICC): UpdateMetadata 24](#_Toc157170393)

[4.2.3 ES8+ (SM-DP+ -- eUICC): InitialiseSecureChannel 26](#_Toc157170394)

[4.2.4 ES8+ (SM-DP+ -- eUICC): ConfigureISDP 27](#_Toc157170395)

[4.2.5 ES8+ (SM-DP+ -- eUICC): StoreMetadata 28](#_Toc157170396)

[4.2.6 ES8+ (SM-DP+ -- eUICC): ReplaceSessionKeys 30](#_Toc157170397)

[4.2.7 ES8+ (SM-DP+ -- eUICC): LoadProfileElements 30](#_Toc157170398)

[4.2.8 ES10a (IPA -- eUICC): GetEuiccConfiguredAddresses 32](#_Toc157170399)

[4.2.9 ES10a (IPA -- eUICC): SetDefaultDPAddress 32](#_Toc157170400)

[4.2.10 ES10b (IPA -- eUICC): PrepareDownload 33](#_Toc157170401)

[4.2.11 ES10b (IPA -- eUICC): LoadBoundProfilePackage 36](#_Toc157170402)

[4.2.12 ES10b (LPA -- eUICC): GetEUICCChallenge 37](#_Toc157170403)

[4.2.13 ES10b (IPA -- eUICC): GetEUICCInfo 37](#_Toc157170404)

[4.2.14 ES10b (IPA -- eUICC): ListNotification 38](#_Toc157170405)

[4.2.15 ES10b (IPA -- eUICC): RetrieveNotificationsList 38](#_Toc157170406)

[4.2.16 ES10b (IPA -- eUICC): RemoveNotificationFromList 41](#_Toc157170407)

[4.2.17 VOID 43](#_Toc157170408)

[4.2.18 ES10b (IPA -- eUICC): AuthenticateServer 43](#_Toc157170409)

[4.2.19 ES10b (LPA -- eUICC): CancelSession 47](#_Toc157170410)

[4.2.20 ES10b (LPA -- eUICC): eUICCMemoryReset 50](#_Toc157170411)

[4.2.21 ES10b (LPA -- eUICC): GetEID 51](#_Toc157170412)

[.4.2.22 ES10b (LPA -- eUICC): GetRAT 52](#_Toc157170413)

[4.2.23 ES10b (LPA -- eUICC): GetProfilesInfo 53](#_Toc157170414)

[4.2.24 ES10b(IPA – eUICC) LoadEUICCPackage 56](#_Toc157170415)

[4.2.25 ES10b (IPA -- eUICC): AddInitialEim 59](#_Toc157170416)

[4.2.26 ES10b (IPA -- eUICC): GetCerts 61](#_Toc157170417)

[4.2.27 ES10b (IPA -- eUICC): EnableUsingDD 63](#_Toc157170418)

[4.2.28 ES10b (IPA -- eUICC): ProfileRollback 66](#_Toc157170419)

[4.2.29 ES10b (IPA -- eUICC): ConfigureAutomaticProfileEnabling 76](#_Toc157170420)

[4.2.30 ES10b (IPA -- eUICC): GetEimConfigurationData 80](#_Toc157170421)

[4.2.31 ESep (eIM -- eUICC): eUICC Package with single PSMO command: Enable 81](#_Toc157170422)

[4.2.32 ESep (eIM -- eUICC): eUICC Package with single PSMO command: Disable 88](#_Toc157170423)

[4.2.33 ESep (eIM -- eUICC): eUICC Package with single PSMO command: Delete 94](#_Toc157170424)

[4.2.34 ESep (eIM -- eUICC): eUICC Package with single PSMO command: ListProfileInfo 98](#_Toc157170425)

[4.2.35 ESep (eIM -- eUICC): eUICC Package with single PSMO command: GetRat 103](#_Toc157170426)

[4.2.36 ESep (eIM -- eUICC): eUICC Package with single eCO command: AddEim 104](#_Toc157170427)

[4.2.37 ESep (eIM -- eUICC): eUICC Package with single eCO command: UpdateEim 108](#_Toc157170428)

[4.2.38 ESep (eIM -- eUICC): eUICC Package with single eCO command: DeleteEim 111](#_Toc157170429)

[4.2.39 ESep (eIM -- eUICC): eUICC Package with single eCO command: ListEim 114](#_Toc157170430)

[4.3 IPAe Interfaces 115](#_Toc157170431)

[4.3.1 ES9+ (IPAe -- SM-DP+): InitiateAuthentication 115](#_Toc157170432)

[4.3.2 ES9+ (IPAe -- SM-DP+): GetBoundProfilePackage 117](#_Toc157170433)

[4.3.3 ES9+ (IPAe -- SM-DP+): AuthenticateClient 119](#_Toc157170434)

[4.3.4 ES9+ (IPAe – SM-DP+): HandleNotification 121](#_Toc157170435)

[4.3.5 ES9+ (IPAe – SM-DP+): CancelSession 123](#_Toc157170436)

[4.3.6 ES9+ (IPAe – SM-DP+): HTTPS 125](#_Toc157170437)

[4.3.7 ES11 (IPAe – SM-DS): InitiateAuthentication 126](#_Toc157170438)

[4.3.8 ES11 (IPAe – SM-DS): AuthenticateClient 128](#_Toc157170439)

[4.3.9 ES11 (IPAe -- SM-DS): HTTPS 129](#_Toc157170440)

[5 Procedure - Behaviour Testing 131](#_Toc157170441)

[5.1 General Overview 131](#_Toc157170442)

[7 External Test Specifications 131](#_Toc157170443)

[7.1 TCA eUICC Profile Package Test Specification 131](#_Toc157170444)

[Annex A Constants 132](#_Toc157170445)

[A.1 Generic Constants 132](#_Toc157170446)

[A.2 Test Certificates and Test Keys 134](#_Toc157170447)

[Annex B Dynamic Content 136](#_Toc157170448)

[Annex C Methods And Procedures 137](#_Toc157170449)

[C.1 Methods 137](#_Toc157170450)

[C.2 Procedures 151](#_Toc157170451)

[Annex D Commands And Responses 152](#_Toc157170452)

[D.1 ES10x Requests And Responses 152](#_Toc157170453)

[D.1.1 ES10x Requests 152](#_Toc157170454)

[D.1.2 ES10x Responses 152](#_Toc157170455)

[D.2 APDU 158](#_Toc157170456)

[D.2.1 APDU Commands 158](#_Toc157170457)

[D.2.2 R-APDU Chaining 159](#_Toc157170458)

[D.3 ESep Requests And Responses 160](#_Toc157170459)

[D.3.1 ESep Responses 160](#_Toc157170460)

[Annex E Profiles 164](#_Toc157170461)

[Annex F IUT Settings 164](#_Toc157170462)

[F.1 eUICC Settings 164](#_Toc157170463)

[F.4 Common Settings 165](#_Toc157170464)

[Annex G Initial States 166](#_Toc157170465)

[G.1 eUICC 166](#_Toc157170466)

[G.1.1 Common Initial States 166](#_Toc157170467)

[G.1.2 For eUICC supporting NIST P-256 166](#_Toc157170468)

[G.1.3 For eUICC supporting BrainpoolP256r1 167](#_Toc157170469)

[G.1.4 With default RAT configuration 167](#_Toc157170470)

[G.1.5 With Additional PPARs in the RAT 167](#_Toc157170471)

[G.1.6 Clean-up procedure 167](#_Toc157170472)

[Annex J Integrated eUICC Testing (Normative) 169](#_Toc157170473)

[Annex K Document Management 172](#_Toc157170474)

[J.1 Document History 172](#_Toc157170475)

[J.2 Other Information 173](#_Toc157170476)

# Introduction

## 1.1 Overview

The main aim of the eSIM IoT specifications [2] & [3] is to provide solution for the Remote SIM Provisioning of IoT Devices.

This Test Plan provides a set of test cases to be used for testing the implementations of the provisioning system specifications documents [2] & [3]. This document offers to the involved entities an unified test strategy and ensures interoperability between different implementations.

## 1.2 Scope

This document is intended for:

* Parties which develop test tools and platforms
* Vendors (Device and eUICC Manufacturers, SM-DP+ and SM-DS Providers)
* Operators

The Test Plan consists of a set of relevant test cases for testing the eUICC. The Implementations Under Test (IUT) are:

* the eUICC

The testing scopes developed in this document are:

* Interface compliance testing: Test cases to verify the compliance of the interfaces within the system.
* System behaviour testing: Test cases to verify the functional behaviour of the system.

Each test case specified within this Test Plan refers to one or more requirements.

The Test Plan contains test cases for the following versions of SGP.22 and SGP.32:

* GSMA RSP Technical Specification [4]
* GSMA IoT eSIM Technical Specification [3]

This document includes an applicability table providing an indication whether test cases are relevant for a specific entity.

## 1.3 Definition of Terms

In addition to the terms which are defined below, the terms defined in SGP.22 [4] and SGP.32 [3] also apply

| Term | Description |
| --- | --- |
| Integrated eUICC Test Interface | An external interface provided by its manufacturer for the purpose of testing eUICC functionality. |
| Test Plan | Current document describing the test cases that allow the eUICC to be tested. |

## 1.4 Abbreviations

In addition to the abbreviations which are defined below, the abbreviations defined in SGP.22 [4] and SGP.32 [3] also apply

| Abbreviation | Description |
| --- | --- |
| APDU | Application Protocol Data Unit |
| ATR | Answer To Reset |
| C-APDU | Command APDU |
| CCID | (USB) Chip Card Interface Device |
| DER TLV | Distinguished Encoding Rules - Tag Length Value |
| FCP | File Control Parameters |
| HW | Hardware |
| IUT | Implementation Under Test |
| KVN | Key Version Number |
| OCE | Off-Card Entity |
| OS | Operating System |
| PIR | Profile Installation Result |
| POR | Proof Of Receipt |
| R-APDU | Response APDU |
| SoC | System on a Chip |
| SP | Service Provider |
| SSD | Supplemental Security Domain |
| USB | Universal Serial Bus |

## 1.5 Document Cross-references

| Ref | Document Number | Title |
| --- | --- | --- |
| [1] | AA.35 | Procedures for Industry Specification version 4.0 |
| [2] | SGP.31 | eSIM IoT Architecture and Requirement Specification Version 1.1 |
| [3] | SGP.32 | eSIM IoT Technical Specification Version 1.0 |
| [4] | SGP.22 | RSP Technical Specification V2.5 |
| [5] | SGP.21 | RSP Architecture V2.5 |
| [6] | eUICC Profile Package | Trusted Connectivity Alliance (formerly SIMalliance) eUICC Profile Package: Interoperable Format Technical Specification V2.1 or later |
| [7] | SGP.26 | RSP Test Certificates Definition v3.0.1 |
| [8] | SGP.23 | RSP Test Specification v1.14 |
| [5] | ETSI TS 102 221 | Smart Cards; UICC-Terminal interface |
| [6] | GPC\_SPE\_034 | GlobalPlatform Card Specification v2.3 |
| [7] | ISO/IEC 7816-4:2013 | Identification cards – Integrated circuit cards - Part 4: Organization, security and commands for interchange |
| [8] | RFC 5639 | Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation |
| [9] | 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 |
| [10] | ITU E.118 | The international telecommunication charge card |
| [11] | 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 |
| [12] | 3GPP TS 23.003 | Digital cellular telecommunications system (Phase 2+);  Universal Mobile Telecommunications System (UMTS);  Numbering, addressing and identification |
| [13] | ETSI TS 102 225 | Secured packet structure for UICC based applications; Release 12 |
| [14] | ETSI TS 102 226 | Remote APDU structure for UICC based applications; Release 9 |
| [15] | TS.26 | GSMA NFC Handset Requirements V9.0 |
| [16] | 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 |
| [17] | ETSI TS 102 241 | Smart cards; UICC Application Programming Interface (UICC API) for Java Card™ |
| [18] | 3GPP TS 31.102 | Characteristics of the Universal Subscriber Identity Module (USIM) application |
| [19] | GPC\_SPE\_095 | GlobalPlatform Card - Digital Letter of Approval - Version 1.0 |
| [20] | RFC 2119 | Key words for use in RFCs to Indicate Requirement Levels, S. Bradner  <http://www.ietf.org/rfc/rfc2119.txt> |
| [21] | Void |  |
| [22] | 3GPP TS 23.040 | Technical realization of the Short Message Service (SMS) |
| [23] | TCA Test | Trusted Connectivity Alliance (TCA) eUICC Profile Package: Interoperable Format Test Specification Version 3.3.1 |
| [24] | RFC 4492 | Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) |
| [25] | SGP.26 | RSP Test Certificates Definition v1.X |
| [26] | 3GPP TS 29.002 | Mobile Application Part (MAP) specification |
| [27] | RFC 5246 | The Transport Layer Security (TLS) Protocol Version 1.2 |
| [28] | GSMA PRD AA.35 | Procedures for Industry Specifications Product |
| [29] | CCID Rev 1.1 | CCID Specification for Integrated Circuit(s) Cards Interface Devices |

## 1.6 Conventions

The key words "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document SHALL be interpreted as described in RFC 2119 [[20]](#RFC2119).

# Testing Rules

## Applicability

### Format of the Optional Features Table

The columns in Table 4 have the following meaning:

|  |  |
| --- | --- |
| Column | Meaning |
| Option | The optional feature supported or not by the implementation. |
| Mnemonic | The mnemonic column contains mnemonic identifiers for each item. |

Table 1: Format of the Optional Features Table

### Format of the Applicability Table

The applicability of every test in Table 5 is formally expressed by the use of a Boolean expression defined in the following clause.

The columns in Table 5 have the following meaning:

|  |  |
| --- | --- |
| Column | Meaning |
| Test case | The "Test case" column gives a reference to the test case number detailed in the present document and is required to validate the implementation of the corresponding item in the "Name" column. |
| Name | In the "Name" column, a short non-exhaustive description of the test is found. |
| Roles | SM-DP+, SM-DS, Device, LPAd, LPAe or eUICC Entities under test that take in charge the functions used in the test case. |
| Version | This column indicates which test cases are applicable for the given SGP.22 version.  See clause 2.1.3 'Applicability and Notations'. |
| Test Env. | Test environment used for executing the test case. |

Table 2: Format of the Applicability Table

### Applicability and Notations

The following notations are used for the Applicability column:

| Applicability code | Meaning |
| --- | --- |
| M | mandatory - the capability is required to be supported. |
| N/A | not applicable - in the given context, it is impossible to use the capability. |
| Ci | conditional - the requirement on the capability depends on the support of other items. "i" is an integer identifying an unique conditional status expression which is defined immediately following the table. For nested conditional expressions, the syntax "IF ... THEN (IF ... THEN ... ELSE...) ELSE ..." is to be used to avoid ambiguities. |

Table 3: Applicability and Notations

### Optional Features Table

The supplier of the implementation SHALL state the support of possible options in Table 5.

NOTE: this table should contain the eUICC Options dedicated to IoT test cases. Those test cases that are applicable as SGP.23 [8] test cases should use the eUICC Options as defined by SGP.23 [8].

|  |  |
| --- | --- |
| eUICC Options | Mnemonic |
| The eUICC terminates EnableProfile and DisableProfile with error "catBusy" when a proactive session is ongoing and the refresh Flag is set. | O\_E\_CATBUSY\_EN\_DIS\_REFRESH |
| The eUICC terminates EnableProfile and DisableProfile with error "catBusy" when a proactive session is ongoing and the refresh Flag is not set. | O\_E\_CATBUSY\_EN\_DIS\_NOREFRESH |
| The eUICC supports the IPAe | O\_E\_IPAe |
| -- as defined in SGP.23 [8] | O\_E\_2\_PIR |
| -- as defined in SGP.23 [8] | O\_E\_INTEGRATED |

Table 4: Options

### Applicability Table

Table 5 specifies the applicability of each test case. See clause 2.1.2 for the format of this table.

| Test case | Name | Role | SGP.33-1 V1.0 | Test Env. |
| --- | --- | --- | --- | --- |
| 4.2.1.2.1 | TC\_eUICC\_ATR\_And\_ISDR\_Selection | eUICC | SGP.23 | TE\_eUICC |
| 4.2.2.2.1 | TC\_eUICC\_ES6.UpdateMetadata | eUICC | SGP.23 | TE\_eUICC |
| 4.2.3.2.1 | TC\_eUICC\_ES8+.InitialiseSecureChannel | eUICC | SGP.23 | TE\_eUICC |
| 4.2.4.2.1 | TC\_eUICC\_ES8+.ConfigureISDP | eUICC | SGP.23 | TE\_eUICC |
| 4.2.5.2.1 | TC\_eUICC\_ES8+.StoreMetadata | eUICC | SGP.23 | TE\_eUICC |
| 4.2.5.2.2 | TC\_eUICC\_ES8+.StoreMetadata\_Service\_Specific\_Data | eUICC | SGP.23 | TE\_eUICC |
| 4.2.6.2.1 | TC\_eUICC\_ES8+.ReplaceSessionKeys | eUICC | SGP.23 | TE\_eUICC |
| 4.2.7.2.1 | TC\_eUICC\_ES8+.LoadProfileElements | eUICC | SGP.23 | TE\_eUICC |
| 4.2.8.2.1 | TC\_eUICC\_ES10a.GetEuiccConfiguredAddresses | eUICC | C3004 | TE\_eUICC |
| 4.2.9.2.1 | TC\_eUICC\_ES10a.SetDefaultDPAddress | eUICC | C3004 | TE\_eUICC |
| 4.2.10.2.1 | TC\_eUICC\_ES10b.PrepareDownloadNIST | eUICC | C3004 | TE\_eUICC |
| 4.2.10.2.2 | TC\_eUICC\_ES10b.PrepareDownloadBRP | eUICC | C3004 | TE\_eUICC |
| 4.2.10.2.3 | TC\_eUICC\_ES10b.PrepareDownloadFRP | eUICC | C3004 | TE\_eUICC |
| 4.2.10.2.4 | TC\_eUICC\_ES10b.PrepareDownloadErrorCases | eUICC | C3004 | TE\_eUICC |
| 4.2.11.2.1 | TC\_eUICC\_ES10b.LoadBoundProfilePackageNIST | eUICC | C3004 | TE\_eUICC |
| 4.2.11.2.2 | TC\_eUICC\_ES10b.LoadBoundProfilePackageBRP | eUICC | C3004 | TE\_eUICC |
| 4.2.11.2.3 | TC\_eUICC\_ES10b.LoadBoundProfilePackageFRP | eUICC | C3004 | TE\_eUICC |
| 4.2.11.2.4 | TC\_eUICC\_ES10b.LoadBoundProfilePackage\_ErrorCases | eUICC | C3004 | TE\_eUICC |
| 4.2.12.2.1 | TC\_eUICC\_ES10b.GetEUICCChallenge | eUICC | C3004 | TE\_eUICC |
| 4.2.13.2.1 | TC\_eUICC\_ES10b.GetEUICCInfo1 | eUICC | C3004 | TE\_eUICC |
| 4.2.13.2.2 | TC\_eUICC\_ES10b.GetEUICCInfo2\_RSP\_Integrated\_eUICC | eUICC | C3004 | TE\_eUICC |
| 4.2.13.2.4 | TC\_eUICC\_ ES10b.GetEUICCInfo2\_v1.0 | eUICC | C3004 | TE\_eUICC |
| 4.2.14.2.1 | TC\_eUICC\_ES10b.ListNotification  All test sequences except the sequence #5 | eUICC | C3005 | TE\_eUICC |
| 4.2.14.2.1 | TC\_eUICC\_ES10b.ListNotification  Only the test sequence #5 | eUICC | C025 | TE\_eUICC |
| 4.2.15.2.1 | TC\_eUICC\_ES10b.RetrieveNotificationsList  All test sequences except the sequences #5 and #15 | eUICC | SGP.23 | TE\_eUICC |
| 4.2.15.2.1 | TC\_eUICC\_ES10b.RetrieveNotificationsList  Only the test sequences #5 and #15 | eUICC | SGP.23 | TE\_eUICC |
| 4.2.16.2.1 | TC\_eUICC\_ES10b.RemoveNotificationFromList  All test sequences except the sequence #5 | eUICC | C3004 | TE\_eUICC |
| 4.2.16.2.1 | TC\_eUICC\_ES10b.RemoveNotificationFromList  Only the test sequence #5 | eUICC | C3004 | TE\_eUICC |
| 4.2.16.2.2 | TC\_eUICC\_ES10b. eUICCPackageResultsList | eUICC | C3004 | Te\_eUICC |
| 4.2.18.2.1 | TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_NIST | eUICC | C3004 | TE\_eUICC |
| 4.2.18.2.2 | TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_BRP | eUICC | C3004 | TE\_eUICC |
| 4.2.18.2.3 | TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_FRP | eUICC | C3004 | TE\_eUICC |
| 4.2.18.2.4 | TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_ErrorCases | eUICC | C3004 | TE\_eUICC |
| 4.2.18.2.5 | TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_BRP | eUICC | C3004 | TE\_eUICC |
| 4.2.18.2.6 | TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_NIST | eUICC | C3004 | TE\_eUICC |
| 4.2.18.2.7 | TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_FRP | eUICC | C3004 | TE\_eUICC |
| 4.2.18.2.8 | TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_ErrorCases | eUICC | C3004 | TE\_eUICC |
| 4.2.19.2.1 | TC\_eUICC\_ES10b.CancelSessionNIST | eUICC | C3004 | TE\_eUICC |
| 4.2.19.2.2 | TC\_eUICC\_ES10b.CancelSessionBRP | eUICC | C3004 | TE\_eUICC |
| 4.2.19.2.3 | TC\_eUICC\_ES10b.CancelSessionFRP | eUICC | C3004 | TE\_eUICC |
| 4.2.19.2.4 | TC\_eUICC\_ES10b.CancelSession\_ErrorCase | eUICC | C3004 | TE\_eUICC |
| 4.2.19.2.5 | TC\_eUICC\_ES10b.CancelSession\_Indirect\_Profile\_Downlaod | eUICC | C3004 | TE\_eUICC |
| 4.2.19.2.6 | TC\_eUICC\_ES10b.CancelSession\_Indirect\_Profile\_Downlaod\_ErrorCases | eUCC | C3004 | TE\_eUICC |
| 4.2.20.2.1 | TC\_eUICC\_ES10b.eUICCMemoryReset | eUICC | C3004 | TE\_eUICC |
| 4.2.20.2.2 | TC\_eUICC\_ES10b.eUICCMemoryReset\_ErrorCases | eUICC | C3004 | TE\_eUICC |
| 4.2.21.2.1 | TC\_eUICC\_ES10b.GetEID | eUICC | C3004 | TE\_eUICC |
| 4.2.22.2.1 | TC\_eUICC\_ES10b.GetRAT | eUICC | C3004 | TE\_eUICC |
| 4.2.23.2.1 | TC\_eUICC\_ES10b.GetProfilesInfo | eUICC | C3004 | TE\_eUICC |
| 4.2.24.2.1 | TC\_eUICC\_ES10b. LoadEUICCPackage | eUICC | NA | TE\_eUICC |
| 4.2.24.2.2 | TC\_eUICC\_ES10b. LoadEUICCPackage\_ErrorCases | eUICC | C3004 | TE\_eUICC |
| 4.2.25.2.1 | TC\_eUICC\_ES10b.AddInitialEim | eUICC | C3004 | TE\_eUICC |
| 4.2.25.2.2 | TC\_eUICC\_ES10b.AddInitialEim\_ErrorCases | eUICC | C3004 | TE\_eUICC |
| 4.2.26.2.1 | TC\_eUICC\_ES10b.GetCertificates | eUICC | C3004 | TE\_eUICC |
| 4.2.26.2.2 | TC\_eUICC\_ES10b.GetCertificates\_ErrorCases | eUICC | C3004 | TE\_eUICC |
| 4.2.27.2.1 | TC\_eUICC\_ES10b. EnableUsingDD | eUICC | C3004 | TE\_eUICC |
| 4.2.27.2.2 | TC\_eUICC\_ES10b. EnableUsingDD\_ErrorCases | eUICC | C3004 | TE\_eUICC |
| 4.2.28.2.1 | TC\_eUICC\_ES10b.ProfileRollback  All test sequences except the sequence #3, sequence #4, sequence #5, and sequence #6 | eUICC | C3004 | TE\_eUICC |
| 4.2.28.2.1 | TC\_eUICC\_ES10b.ProfileRollback  Only the sequence #3 and sequence #5 | eUICC | C3001 | TE\_eUICC |
| 4.2.28.2.1 | TC\_eUICC\_ES10b.ProfileRollback  Only the sequence #4 and sequence #6 | eUICC | C3003 | TE\_eUICC |
| 4.2.28.2.2 | TC\_eUICC\_ES10b.ProfileRollback\_ErrorCases  All test sequences except the sequence #3, and sequence #4 | eUICC | C3004 | TE\_eUICC |
| 4.2.28.2.2 | TC\_eUICC\_ES10b.ProfileRollback\_ErrorCases  Only sequence #3 | eUICC | C3000 | TE\_eUICC |
| 4.2.28.2.2 | TC\_eUICC\_ES10b.ProfileRollback\_ErrorCases  Only sequence #4 | eUICC | C3002 | TE\_eUICC |
| 4.2.29.2.1 | TC\_eUICC\_ES10b ConfigureAutomaticProfileEnabling | eUICC | C3004 | TE\_eUICC |
| 4.2.29.2.2 | TC\_eUICC\_ES10b ConfigureAutomaticProfileEnabling\_ErrorCases | eUICC | C3004 | TE\_eUICC |
| 4.2.30.2.1 | TC\_eUICC\_ES10b.GetEimConfigurationData | eUICC | C3004 | TE\_eUICC |
| 4.2.31.2.1 | TC\_eUICC\_ESep.Enable | eUICC | M | TE\_eUICC |
| 4.2.31.2.2 | TC\_eUICC\_ESep.Enable\_ErrorCases | eUICC | M | TE\_eUICC |
| 4.2.32.2.1 | TC\_eUICC\_ESep.Disable | eUICC | M | TE\_eUICC |
| 4.2.32.2.2 | TC\_eUICC\_ESep.Disable\_ErrorCases | eUICC | M | TE\_eUICC |
| 4.2.33.2.1 | TC\_eUICC\_ESep.Delete | eUICC | M | TE\_eUICC |
| 4.2.33.2.2 | TC\_eUICC\_ESep.Delete\_ErrorCases | eUICC | M | TE\_eUICC |
| 4.2.34.2.1 | TC\_eUICC\_ESep.ListProfileInfo | eUICC | M | TE\_eUICC |
| 4.2.34.2.2 | TC\_eUICC\_ESep.ListProfileInfo\_ErrorCases | eUICC | M | TE\_eUICC |
| 4.2.35.2.1 | TC\_eUICC\_ESep.GetRat | eUICC | M | TE\_eUICC |
| 4.2.36.2.1 | TC\_eUICC\_ESep.AddEIM | eUICC | M | TE\_eUICC |
| 4.2.36.2.2 | TC\_eUICC\_ESep.AddEIM\_ErrorCases | eUICC | M | TE\_eUICC |
| 4.2.37.2.1 | TC\_eUICC\_ESep.UpdateEIM | eUICC | M | TE\_eUICC |
| 4.2.37.2.2 | TC\_eUICC\_ESep.UpdateEIM\_ErrorCases | eUICC | M | TE\_eUICC |
| 4.2.38.2.1 | TC\_eUICC\_ESep.DeleteEIM | eUICC | M | TE\_eUICC |
| 4.2.38.2.2 | TC\_eUICC\_ESep.DeleteEIM\_ErrorCases | eUICC | M | TE\_eUICC |
| 4.2.39.2.1 | TC\_eUICC\_ESep.ListEIM | eUICC | M | TE\_eUICC |
| 4.2.39.2.2 | TC\_eUICC\_ESep.ListEIM\_ErrorCases | eUICC | NA | NA |
| Test Specifications | | | | |
| 7.1 | TCA eUICC Profile Package Test Specification | eUICC | SGP.23 | See section 7.1 |

Table 5: Applicability of Tests

| Conditional item | Condition |
| --- | --- |
| C3000 | IF (O\_E\_CATBUSY\_EN\_DIS\_REFRESH AND NOT O\_E\_IPAe) THEN M ELSE N/A |
| C3001 | IF (NOT O\_E\_CATBUSY\_EN\_DIS\_REFRESH AND NOT O\_E\_IPAe) THEN M ELSE N/A |
| C3002 | IF (O\_E\_CATBUSY\_EN\_DIS\_NOREFRESH AND NOT O\_E\_IPAe) THEN M ELSE N/A |
| C3003 | IF (NOT O\_E\_CATBUSY\_EN\_DIS\_NOREFRESH AND NOT O\_E\_IPAe) THEN M ELSE N/A |
| C3004 | IF (NOT O\_E\_IPAe) THEN M ELSE N/A |
| C3005 | IF (O\_E\_2\_PIR AND NOT O\_E\_IPAe) THEN M ELSE N/A |
| C3006 | IF (O\_E\_INTEGRATED AND NOT O\_E\_IPAe) THEN M ELSE N/A |

Table 6: Conditional Items Referenced by Table 5

## General Consideration

This section contains some general considerations about the test cases defined in this document. Note that some external test specifications are referred to in chapter 7. Consequently, the following sub sections SHALL only apply for test cases defined in sections 4 and 5 and 6.

### Test Case Definition

Test descriptions are independent.

For each test described in this document, a chapter provides a general description of the initial conditions applicable for the whole test. This description is completed by specific configurations to each individual sub-case.

It is implicitly assumed that all entities under test SHALL be compliant with the initial states described in Annex G. An initial state SHALL be considered as a pre-requisite to execute all the test cases described in this Test Plan.

After completing the test, the configuration is reset before the execution of the following test.

### Test Cases Format

Here is an explanation of the way to define the test cases in chapters 4, 5 and 6.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **4.X.Y.Z Test Cases**  **4.X.Y.Z.1 TC\_IUT\_TestName1**   |  |  | | --- | --- | | General Initial Conditions | | | Entity | Description of the general initial condition | | Entity1 | Test case - general condition 1 | | Entity2 | Test case - general condition 2 |   **Test Sequence #01: Short Description**  Description of the aim of the test sequence N°1   |  |  | | --- | --- | | Initial Conditions |  | | Entity | Description of the initial condition | | Entity1 | Test sequence N°1 - initial condition 1 | | Entity2 | Test sequence N°1 - initial condition 2 |  | Step | Direction | Sequence / Description | Expected result | | --- | --- | --- | --- | | IC1 | Entity1 → Entity2 | Command or Message to send from Entity1 to Entity2 | Expected result N°1.1 | | 1 | Entity1 → Entity2 | Command or Message to send from Entity1 to Entity2 | 1- expected result N°1.2  2- expected result N°1.3 | | 2 | Entity2 → Entity3 | Command or Message to send from Entity2 to Entity3 |  |   **Test Sequence #02**  Description of the aim of the test sequence N°2   | Step | Direction | Sequence / Description | Expected result | | --- | --- | --- | --- | | 1 | Entity1 → Entity2 | Command or Message to send from Entity1 to Entity2 |  | | 2 | Entity2 → Entity3 | Command or Message to send from Entity2 to Entity3 | 1- expected result N°2.1  2- expected result N°2.2 |   **4.X.Y.Z.2 TC\_IUT\_TestName2**  … |

The test cases TC\_IUT\_TestName1 and TC\_IUT\_TestName2 are referenced in Table 5 that allows indicating the applicability of the tests.

The test sequence #01 SHALL be executed if and only if these conditions are met:

* Test case - general condition 1
* Test case - general condition 2
* Test sequence N°1 - initial condition 1
* Test sequence N°1 - initial condition 2

The test sequence #02 SHALL be executed if and only if these conditions are met:

* Test case - general condition 1
* Test case - general condition 2

The tables defining the different initial conditions are optional.

Initial Conditions are intended to be reached dynamically using the Test Tool when possible.

No additional operation SHALL be done prior to the test sequence besides those indicated in the Initial Conditions (e.g. no other Profiles SHALL be present on the eUICC besides those defined in the Initial Conditions).

Note that all initial states (described in Annex G) SHALL be implemented by the entity under test whatever the test cases to execute.

In addition, following 2.2.1 sub sections present all information (e.g. Methods, Constants…) that MAY be referenced in test sequences.

After execution of each test sequence a clean-up procedure (CU) SHALL be executed to restore the IUT to the Common Initial State as defined in Annex G.

#### Methods and Procedures

A method is referenced as follow:

 MTD\_NAME\_OF\_THE\_METHOD(PARAM1, PARAM2…)

The key word “NO\_PARAM” SHALL be set in method call if the related optional parameter is not used.

All methods and their related parameters are described in Annex C.1.

A procedure is a generic sub-sequence and is referenced as follow:

* PROC\_NAME\_OF\_THE\_PROCEDURE

All procedures are described in Annex C.2.

The implementation of these methods and procedures is under the responsibility of the test tool providers.

#### Constants and Dynamic Content

A constant (e.g. text, ASN.1 structure, hexadecimal string, icon, URI, integer, EID, AID…) is referenced as follow:

 #NAME\_OF\_THE\_CONSTANT

All constants are defined in Annex A.

When provided as an ASN.1 value notation, a constant SHALL be encoded in DER TLV (as specified in ITU-T X.690 [16]) by the test tool.

A dynamic content (e.g. TLV, ASN.1 structure, signature, integer, AID, one-time key pair…) is referenced as follow:

 <NAME\_OF\_THE\_VARIABLE>

All dynamic contents are defined in Annex B.

A dynamic content is either generated by an IUT or by a test tool provider.

#### Requests and Responses

An ASN.1 or a JSON request is referenced as follow:

 #NAME\_OF\_THE\_REQUEST

An ASN.1 or a JSON response is referenced as follows:

 #R\_NAME\_OF\_THE\_RESPONSE

Each ASN.1 or JSON request and response MAY refer to a constant or a dynamic content. All these structures are defined in Annex D.

When provided as an ASN.1 value notation, a request or a response SHALL be encoded in DER TLV (as specified in ITU-T X.690 [16]) by the test tool.

When an ASN.1 element definition contains three points (i.e. “…”), it means that fields MAY be present but SHALL not be checked by the test tool.

In the following example, several fields MAY be part of the ProfileInfoListResponse but only the profileNickname SHALL be verified.

resp ProfileInfoListResponse ::=

profileInfoListOk :{

{   
 ...  
 profileNickname #NICKNAME  
 ...

}

}

This rule applies also for Constants definition.

Some ASN.1 SEQUENCE components have a DEFAULT value (for example, profileClass in StoreMetadataRequest). In this specification, when values are specified in ASN.1 syntax and the DEFAULT value is intended, two different formulations (both of which are valid) may be used:

* the relevant component is specified with the DEFAULT value;
* the relevant component is missing entirely.

These are logically equivalent and lead to the same DER encoding. In both cases, the following rules apply:

* When the test tool is sending the DER value, it SHALL NOT include the component (as per DER rules).
* When the test tool is checking a received DER value from the entity under test, it SHALL check that the component is NOT present.

Test tools SHALL consider two BIT STRINGs to be equivalent if the BIT STRINGs have the same DER encoding. For example, '0101'B shall be considered to be equivalent to '010100'B.

NOTE: this is equivalent to removing any trailing zero bits from the BIT STRINGs in "bstring" notation (e.g. '010100'B à '0101'B) and then comparing the strings textually.

NOTE: according to the DER format, the encoding of transmitted values will remove the trailing zeroes. The definition above allows for values which are specified using ASN.1 value notation and are not transmitted, such as values specified in the Annexes of the current document, including IUT settings which might be specified by a user of the current document and may contain trailing zeroes in the ASN.1 value notation.

#### APDUs

A C-APDU is referenced as follow:

 [NAME\_OF\_THE\_CAPDU]

All C-APDUs are defined in Annex D.4.

An R-APDU is referenced as follow:

 [R\_NAME\_OF\_THE\_RAPDU]

All R-APDUs are defined in Annex D.4.

Each APDU MAY refer to a constant or a dynamic content.

The APDU TERMINAL RESPONSE SHALL be dynamically generated by the test tool according to the related proactive command. Therefore, this particular command is not referenced with brackets in this specification. If not explicitly defined in the step, the general result SHALL be set by default to “Command performed successfully” (i.e. 0x83 01 00).

#### Profiles

In order to execute the test cases described in this document, Operational, Test and Provisioning Profiles are necessary. All these Profiles are defined in Annex E with the Profile Metadata content and the corresponding Profile Package as defined in the eUICC Profile Package Specification [X].

A Profile is referenced as follow:

 PROFILE\_OPERATIONALx with x the identifier of the Operational Profile

or

 PROFILE\_TESTx with x the identifier of the Test Profile

or

 PROFILE\_PROVISIONINGx with x the identifier of the Provisioning Profile

NOTE: Test Profiles and Provisioning Profiles are out of the scope of this version of test specification.

#### IUT Settings

For the purpose of some test cases, Device and eUICC manufacturers and Platforms (i.e. SM-DP+, SM-DS) providers need to give some information related to their products to the test tools providers (e.g. supported Java Card version).

An IUT setting is referenced as follow:

 #IUT\_NAME\_OF\_SETTING

All these settings are defined in Annex F.

### General Rules for eUICC Testing

#### Default Profile Downloading process

By default, when an Operational Profile needs to be downloaded on the eUICC (e.g. As mentioned in an initial condition), the following rules apply except if it is differently defined in the Test Case.

#### Default Profile Downloading process

By default, when an Operational Profile needs to be downloaded on the eUICC (e.g. As mentioned in an initial condition), the following rules apply except if it is differently defined in the Test Case.

The highest priority CI in euiccCiPKIdListForSigning SHALL be used.

In order to execute the Common Mutual Authentication procedure and the Sub-procedure Profile Download and Installation (End User Confirmation), the following requests SHALL be sent by the Test Tool:

 #GET\_EUICC\_INFO1 and #GET\_EUICC\_CHALLENGE

 #AUTH\_SMDP\_MATCH\_ID

* + - * with the <EUICC\_CI\_PK\_ID\_TO\_BE\_USED> set to the CI for signing indicated as highest priority in the #R\_EUICC\_INFO1
      * with the #CERT\_S\_SM\_DPauth\_ECDSA leading to the same CI as the one chosen for signing
      * with the SM-DP+ address #TEST\_DP\_ADDRESS1

 #PREP\_DOWNLOAD\_NO\_CC

* + - * with the #CERT\_S\_SM\_DPpb\_ECDSA leading to the same CI as the one chosen for signing

 Neither ES10b.GetRAT nor ES10b.GetProfilesInfo requests SHALL be executed

During the Profile Installation, the following SCP03t TLVs SHALL be used by default:

 #S\_INIT\_SC\_PROF1

 #CONF\_ISDP\_EMPTY

 no TLV for "ES8+.ReplaceSessionKeys" function SHALL be used (i.e. the Profile SHALL be downloaded by using the session keys <S\_ENC> and <S\_MAC>)

#### Default Local Profile Management process

By default, when an Operational Profile needs to be enabled, disabled or deleted on the eUICC (e.g. As mentioned in an initial condition), the following rules apply except if it is differently defined in the Test Case.

The EnableProfileRequest and the DisableProfileRequest SHALL contain the following parameters:

 ICCID of the Profile to Enable or to Disable

 RefreshFlag set to TRUE

The eUICC SHALL send the REFRESH command in "UICC Reset" mode (i.e. the APDU[TERMINAL\_PROFILE] indicating the support "UICC Reset" SHALL be used by the Test Tool).

The DeleteProfileRequest SHALL contain the following parameter:

 ICCID of the Profile to Delete

#### ASN.1 elements verifications

Each time the eUICC returns an ASN.1 structure containing a SEQUENCE OF elements, the order of elements SHALL be checked by the Test Tool except for the particular responses:

 notificationMetadataList of ListNotificationResponse

 profileInfoListOk of ProfileInfoListResponse

 notificationList of RetrieveNotificationsListResponse

When an Operational Profile class is expected to be indicated in a ProfileInfoListResponse, the Test Tool SHALL accept only one DER encoding if the eUICC supports SGP.22 v2.2.x [2] or SGP.22 V2.2 [2b]: a tag 0x95 containing the integer value 2.

# Testing Architecture

## 3.1 Testing Scope

All the interfaces, intended to be tested in the scope of this document, are presented hereafter:



| **Interface** | **Between** | | **Description** | **SGP.33-1** |
| --- | --- | --- | --- | --- |
| ES2+ | Operator | SM-DP+ | Used by the Operator to order Profiles for specific eUICCs as well as other administrative functions as defined in SGP.31 [2]. | Out of scope |
| ES6 | Operator | eUICC | Used by the Operator for the management of Operator services via OTA services as defined in SGP.31 [2]. | In scope |
| ES8+ | SM-DP+ | eUICC | Provides a secure end-to-end channel between the SM-DP+ and the eUICC for the administration of the ISD-P and the associated Profile during download and installation. It provides Perfect Forward Secrecy as defined in SGP.31 [2]. | In scope |
| ES9+ | SM-DP+ | IPA | Used to provide a secure transport between the SM-DP+ and the IPA for the delivery of the Bound Profile Package as defined in SGP.31 [2]. | Out of scope |
| ES9+’ | SM-DP+ | eIM | Used to provide a secure transport between the SM-DP+ and the eIM for the delivery of the Bound Profile Package as defined in SGP.31 [2]. | Out of scope |
| ES10a | IPA | eUICC | Used between the IPA (in the IoT Device) and the eUICC to handle a Profile discovery as defined in SGP.31 [2]. | In scope |
| ES10b | IPA | eUICC | Used between the IPA (in the IoT Device) and the IPA Services to transfer a Bound Profile Package to the eUICC as defined in SGP.31 [2]. This interface plays no role in the decryption of Profile Packages. | In scope |
| ES11 | IPA | SM-DS | Used by the IPA to retrieve Event Records for the respective eUICC as defined in SGP.31 [2]. | Out of scope |
| ES11’ | eIM | SM-DS | Used by the eIM to retrieve Event Records for the respective eUICC as defined in SGP.31 [2]. | Out of scope |
| ES12 | SM-DP+ | SM-DS | Used by the SM-DP+ to issue or remove Event Registrations on the SM-DS as defined in SGP.31 [2]. | Out of scope |
| ESep | eIM | eUICC | Logical end-to-end interface between the eIM and the eUICC used to transfer eUICC Packages for Profile State management and eIM configuration by eIM, as defined in SGP.31 [2]. | In scope |
| ESipa | eIM | IPA | Logical interface between an eIM and an IPA, as defined in SGP.31 [2], used to trigger a Profile download at the IPA and to provide a secure transport for the delivery of eUICC Packages, unless the underlying transport provides necessary security. | Out of scope |

Table 7: Interfaces Descriptions

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

## 3.2 Testing Execution

This chapter aims to describe the different testing environments and equipments to allow the test cases to be executed.

To permit the execution of the different test cases described in this Test Plan, specifics simulators SHALL be used. The simulators that have been defined are listed hereafter:

 S\_IoT\_Device: the Device Simulator used to send some commands to the eUICC under test using ISO/IEC 7816-4 [7] on the contact interface

 S\_SM-DP+: the SM-DP+ Simulator

 S\_SM-DS: the SM-DS Simulator

 S\_MNO: the MNO Simulator

 S\_IPAd: the IPAd Simulator

 S\_eIM: the eIM Simulator

 Implementation of these simulators remains under the responsibility of the test tool providers.

 The aim of all the test cases is to verify the compliance of an Actor/Component (i.e. eUICC).

Following notations are used:

 S\_ComponentName for a simulated component

 ComponentName for the Implementation Under Test (IUT)

 The use of "-- optional" in any ASN.1 elements defined within this document indicate that the test tool SHALL allow for the value either being present with that value, or being absent.

### 3.2.1 eUICC - Test Environment

The following test environment is used for all eUICC test cases as defined in chapter 4.2 and 5.2 (unless it is specified differently in the specific test case). Following conditions apply:

 Removable eUICC is used



 EUM SHALL provide products compliant with Annex G.1 – eUICC Initial States

 IPAd / MNO / SM-DP+ / SM-DS / eIM / IoT Device Simulators SHALL be implemented by the test tools

eUICC

S\_IoT\_Device, S\_IPA

(S\_MNO, S\_SM-DP+, S\_SM-DS, S\_eIM)

ES6, ES8+, ESep

ES10a, ES10b

The reference of this Test Environment is TE\_eUICC.

### 3.2.5 Integrated eUICC – Test Environment

The following test environment is used for all eUICC test cases as defined in chapter 4.2 and 5.2 (unless it is specified differently in the specific test case). Following conditions apply:

* EUM SHALL provide products compliant with Annex G.1 – eUICC Initial States
* IPAd / MNO / SM-DP+ / SM-DS / eIM / IoT Device Simulators SHALL be implemented by the test tools
* Integrated eUICC shall provide a test interface which includes one of the following:
  + ISO/IEC 7816-4 [7]
  + USB CCID [29]
* For Integrated eUICC providing a USB CCID [29] test interface, the provisions of Annex J SHALL apply
* For Integrated eUICC providing ISO/IEC 7816-4 [7], the requirements of 3.2.1 eUICC – Test Environment with implementing shall apply

ES6, ES8+, ESep, ES10a, ES10b

PC/SC

CCID USB

S\_IoT\_Device,

S\_IPA

(S\_MNO, S\_SM-DP+, S\_SM-DS, S\_eIM)

ES10a, ES10b

**PC/SC**

**USB**

**Integrated eUICC Test Interface (e.g.USB/Bluetooth)**

**Integrated eUICC**

**IOT Device**

The reference of this [29] USB CCID based Test Environment is TE\_Integrated eUICC.

# Interface Compliance Testing

## General Overview

This section focuses on the implementation of the different interfaces according to the eSIM IoT Technical Specification [3]. The aim is to verify the compliance of all interfaces within the system.

## eUICC Interfaces

### 4.2.1 ATR and ISD-R Selection

#### 4.2.1.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specificaiton [3]

#### 4.2.1.2 Test Cases

##### 4.2.1.2.1 TC\_eUICC\_ATR\_And\_ISDR\_Selection

Test Sequence #01 Nominal: ATR and Select ISD-R

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: ATR and Select ISD-R* defined in section 4.2.1.2.1 TC\_eUICC\_ATR\_And\_ISDR\_Selection. Where the S\_LPAd is playing the role of the S\_IPA.

### 4.2.2 ES6 (Operator -- eUICC): UpdateMetadata

#### 4.2.2.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specificaiton [3]

3GPP TS 23.040 - Technical realization of the Short Message Service (SMS) [22]

#### 4.2.2.2 Test Cases

##### 4.2.2.2.1 TC\_eUICC\_ES6.UpdateMetadata

ES6 interface is identical to the one defined in section 5.4 of SGP.22 [4].

Test Sequence #01 Nominal: Unset PPR1

The purpose of this test is to verify that the MNO can unset PPR1 from a Profile and that the eUICC can handle an Update Metadata request with only one field present.

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Unset PPR1* defined in section 4.2.2.2.1 TC\_eUICC\_ES6.UpdateMetadata

Test Sequence #02 Nominal: Unset PPR2 and update icon

The purpose of this test is to verify that the MNO can unset PPR2 and update the icon and icon type values from a Profile.

Test Sequence #03 Nominal: Unset PPR1 and PPR2 and update Profile name and Service Provider name

The purpose of this test is to verify that MNO can unset PPR1 and PPR2 from a Profile and can update the Service Provider Name and Profile Name values.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Unset PPR1 and PPR2 and update Profile name and Service Provider name* defined in section 4.2.2.2.1 TC\_eUICC\_ES6.UpdateMetadata

Test Sequence #04 Nominal: Delete PPRs, Service Provider and Profile names

The purpose of this test is to verify that the MNO can delete all PPRs, the Service Provider name and the Profile name from a Profile.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: Delete PPRs, Service Provider and Profile names* defined in section 4.2.2.2.1 TC\_eUICC\_ES6.UpdateMetadata

Test Sequence #05 Nominal: Delete icon

This test sequence of this errro is not applicable for this version of the specification because is indicated for FFS in SGP.23 [8]

Test Sequence #06 Nominal: Delete Unset PPRs

The purpose of this test is to verify that the MNO can delete already unset PPRs using the Update Metadata request.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Nominal: Delete Unset PPRs* defined in section 4.2.2.2.1 TC\_eUICC\_ES6.UpdateMetadata

Test Sequence #07 Error: Set a pprUpdateControl value to one

The purpose of this test is to verify that the eUICC is correctly handling a pprUpdateControl value error from the MNO request, and return the expected error code status.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Error: Set a pprUpdateControl value to one* defined in section 4.2.2.2.1 TC\_eUICC\_ES6.UpdateMetadata

Test Sequence #08 Error: Update Metadata on a Disable Profile

The purpose of this test is to verify that the eUICC is correctly rejecting an Update Metadata request from the MNO when the targeted Profile is Disabled.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Error: Update Metadata on a Disable Profile*  defined in section 4.2.2.2.1 TC\_eUICC\_ES6.UpdateMetadata

Test Sequence #09 Error: Empty request

The purpose of this test is to verify that the eUICC is correctly rejecting an Update Metadata request from the MNO when no field is present.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #09 Error: Empty request* defined in section 4.2.2.2.1 TC\_eUICC\_ES6.UpdateMetadata

Test Sequence #10 Error: Update Icon without Icon Type field

The purpose of this test is to verify that the eUICC is correctly rejecting an Update Metadata request from the MNO when the icon field is present but not the icon type field.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #10 Error: Update Icon without Icon Type field* defined in section 4.2.2.2.1 TC\_eUICC\_ES6.UpdateMetadata

Test Sequence #11 Error: Update Icon Type without Icon field

The purpose of this test is to verify that the eUICC is correctly rejecting an Update Metadata request from the MNO when the Icon Type field is present but not the Icon field.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #11 Error: Update Icon Type without Icon field* defined in section 4.2.2.2.1 TC\_eUICC\_ES6.UpdateMetadata

### 4.2.3 ES8+ (SM-DP+ -- eUICC): InitialiseSecureChannel

#### 4.2.3.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.3.2 Test Cases

##### 4.2.3.2.1 TC\_eUICC\_ES8+.InitialiseSecureChannel

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

Test Sequence #01 Error: Invalid Remote Operation

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: Invalid Remote Operation* defined in section 4.2.3.2.1 TC\_eUICC\_ES8+.InitialiseSecureChannel

Test Sequence #02 Error: Invalid SM-DP+ Signature

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Error: Invalid SM-DP+ Signature* defined in section 4.2.3.2.1 TC\_eUICC\_ES8+.InitialiseSecureChannel

Test Sequence #03 Error: Invalid Transaction Identifier

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Error: Invalid Transaction Identifier* defined in section 4.2.3.2.1 TC\_eUICC\_ES8+.InitialiseSecureChannel

Test Sequence #04 Error: Invalid CRT Values

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Error: Invalid CRT Values* defined in section 4.2.3.2.1 TC\_eUICC\_ES8+.InitialiseSecureChannel

Test Sequence #05 Error: InitialiseSecureChannel request while Secure Channel Session is ongoing

This test sequence is the same as SGP.23 [8] – the *Test Sequence #05 Error: InitialiseSecureChannel request while Secure Channel Session* is ongoing defined in section 4.2.3.2.1 TC\_eUICC\_ES8+.InitialiseSecureChannel

### 4.2.4 ES8+ (SM-DP+ -- eUICC): ConfigureISDP

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC. Therefore, the Test Sequence referencing int this section can be applied as defined by SGP.23

#### 4.2.4.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.4.2 Test Cases

##### 4.2.4.2.1 TC\_eUICC\_ES8+.ConfigureISDP

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

Test Sequence #01 Nominal: Empty Proprietary Data

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Empty Proprietary Data* defined in section 4.2.4.2.1 TC\_eUICC\_ES8+.ConfigureISDP

Test Sequence #02 Nominal: Proprietary Data with the maximum length authorized (i.e. 128 bytes)

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: Proprietary Data with the maximum length authorized (i.e. 128 bytes)* defined in section 4.2.4.2.1 TC\_eUICC\_ES8+.ConfigureISDP

Test Sequence #03 Error: Proprietary Data with the maximum length exceeded (i.e. 129 bytes)

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Error: Proprietary Data with the maximum length exceeded (i.e. 129 bytes)* defined in section 4.2.4.2.1 TC\_eUICC\_ES8+.ConfigureISDP

### 4.2.5 ES8+ (SM-DP+ -- eUICC): StoreMetadata

#### 4.2.5.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.5.2 Test Cases

##### 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

Test Sequence #01 Nominal: All Metadata fields present (PNG icon used and PPR1 set)

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: All Metadata fields present (PNG icon used and PPR1 set)* defined in section 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata

Test Sequence #02 Nominal: With JPG icon

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: With JPG icon* defined in section 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata.

Test Sequence #03 Nominal: Without providing Profile Class

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Without providing Profile Class* defined in section 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata

Test Sequence #04 Nominal: With PPR2 set

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: With PPR2 set* defined in section 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata

Test Sequence #05 Nominal: With PPR1 and PPR2 set

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: All Metadata fields present (PNG icon used and PPR1 set)* defined in section 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Nominal: With PPR1 and PPR2 set* defined in section 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata

Test Sequence #07 Error: ICCID already present in the eUICC

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Error: ICCID already present in the eUICC* defined in section 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata

Test Sequence #08 Error: Profile Policy Rule is set but Profile Owner is not

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Error: Profile Policy Rule is set but Profile Owner is not* defined in section 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata

Test Sequence #09 Error: Profile Owner is set with a wildcard ('E') digits

This test sequence is the same as SGP.23 [8] - the *Test Sequence #09 Error: Profile Owner is set with a wildcard ('E') digits* defined in section 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata

Test Sequence #10 Error: Icon Type is set but icon is not

This test sequence is the same as SGP.23 [8] - the *Test Sequence #10 Error: Icon Type is set but icon is not* defined in section 4.2.5.2.1 TC\_eUICC\_ES8+.StoreMetadata

##### 4.2.5.2.2 TC\_eUICC\_ES8+.StoreMetadata\_Service\_Specific\_Data

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

Test Sequence #01 Nominal: Metadata include service-specific data, stored

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Metadata include service-specific data, stored* defined in section 4.2.5.2.2 TC\_eUICC\_ES8+.StoreMetadata\_Service\_Specific\_Data

Test Sequence #02 Nominal: Metadata include service-specific data, not stored

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: Metadata include service-specific data, not stored* defined in section 4.2.5.2.2 TC\_eUICC\_ES8+.StoreMetadata\_Service\_Specific\_Data

Test Sequence #03 Nominal: Metadata include service-specific data, stored and not stored

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Metadata include service-specific data, stored and not stored* defined in section 4.2.5.2.2 TC\_eUICC\_ES8+.StoreMetadata\_Service\_Specific\_Data

### 4.2.6 ES8+ (SM-DP+ -- eUICC): ReplaceSessionKeys

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

#### 4.2.6.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specificaiton [3]

#### 4.2.6.2 Test Cases

##### 4.2.6.2.1 TC\_eUICC\_ES8+.ReplaceSessionKeys

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

Test Sequence #01 Error: Incorrect PPK size

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: Incorrect PPK size* defined in section *4.2.6.2.1 TC\_eUICC\_ES8+.ReplaceSessionKeys.*

### 4.2.7 ES8+ (SM-DP+ -- eUICC): LoadProfileElements

#### 4.2.7.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specificaiton [3]

#### 4.2.7.2 Test Cases

##### 4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

Test Sequence #01 Error: EFICCID different from the ICCID provided in the Profile Metadata

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: EFICCID different from the ICCID provided in the Profile Metadata* defined in section *4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements.*

Test Sequence #02 Error: MCC / MNC of EFIMSI different from MCC / MNC of Profile Owner present in Metadata

This test sequence is the same as SGP.23 [8] - the Test Sequence #02 Error: MCC / MNC of EFIMSI different from MCC / MNC of Profile Owner present in Metadata defined in section *4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements.*

Test Sequence #03 Error: Session MAC chaining used instead of new Initial MAC chaining

This test sequence is the same as SGP.23 [8] - the Test Sequence #03 Error: Session MAC chaining used instead of new Initial MAC chaining defined in section *4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements.*

Test Sequence #04 Error: S-MAC used instead of PPK-MAC

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Error: S-MAC used instead of PPK-MAC* defined in section *4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements.*

Test Sequence #05 Error: S-ENC used instead of PPK-ENC

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Error: S-ENC used instead of PPK-ENC* defined in section *4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements.*

Test Sequence #06 Error: Profile Downloading stopped by a Reset

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Error: Profile Downloading stopped by a Reset* defined in section *4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements.*

Test Sequence #07 Nominal: ICCID in the 'ProfileHeader' PE is ignored by the eUICC

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Nominal: ICCID in the 'ProfileHeader' PE is ignored by the eUICC* defined in section *4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements.*

Test Sequence #08 Nominal: With gid1 and gid2 set

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Nominal: With gid1 and gid2 set* defined in section 4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements

Test Sequence #09 Error: gid1 and gid2 provided in the Profile Metadata but not in the Profile Package

This test sequence is the same as SGP.23 [8] - the *Test Sequence #09 Error: gid1 and gid2 provided in the Profile Metadata but not in the Profile Package* defined in section 4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements

Test Sequence #10 Error: gid1 and gid2 not provided in the Profile Metadata but present in Profile Package

This test sequence is the same as SGP.23 [8] - the *Test Sequence #10 Error: gid1 and gid2 not provided in the Profile Metadata but present in Profile Package* defined in section 4.2.7.2.1 TC\_eUICC\_ES8+.LoadProfileElements

### 4.2.8 ES10a (IPA -- eUICC): GetEuiccConfiguredAddresses

#### 4.2.8.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.8.2 Test Cases

##### 4.2.8.2.1 TC\_eUICC\_ES10a.GetEuiccConfiguredAddresses

This ES10a.GetEuiccConfiguredAddresses function is identical to the ES10a.GetEuiccConfiguredAddresses function defined in section 5.7.3 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: Only Root SM-DS Address

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Only Root SM-DS Address* defined in 4.2.8.2.1 TC\_eUICC\_ES10a.GetEuiccConfiguredAddresses, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: Root SM-DS and Default SM-DP+ Addresses

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: Root SM-DS and Default SM-DP+ Addresses* defined in 4.2.8.2.1 TC\_eUICC\_ES10a.GetEuiccConfiguredAddresses, where the S\_LPAd plays the role of S\_IPA.

### 4.2.9 ES10a (IPA -- eUICC): SetDefaultDPAddress

#### 4.2.9.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.9.2 Test Cases

##### 4.2.9.2.1 TC\_eUICC\_ES10a.SetDefaultDPAddress

This ES10a.SetDefaultDPAddress function is identical to the ES10a. SetDefaultDPAddress function defined in section 5.7.4 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: Set SM-DP+ Address with Address Empty in eUICC

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Set SM-DP+ Address with Address Empty in eUICC* defined in 4.2.9.2.1 TC\_eUICC\_ES10a.SetDefaultDPAddress, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: Set SM-DP+ Address with SM-DP+ Address already in eUICC

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: Set SM-DP+ Address with SM-DP+ Address already in eUICC* defined in 4.2.9.2.1 TC\_eUICC\_ES10a.SetDefaultDPAddress, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Nominal: Set Empty SM-DP+ Address with SM-DP+ Address already in eUICC

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Set Empty SM-DP+ Address with SM-DP+ Address already in eUICC* defined in 4.2.9.2.1 TC\_eUICC\_ES10a.SetDefaultDPAddress, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #04 Nominal: Set Empty SM-DP+ Address with Empty SM-DP+ Address in eUICC

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: Set Empty SM-DP+ Address with Empty SM-DP+ Address in eUICC* defined in 4.2.9.2.1 TC\_eUICC\_ES10a.SetDefaultDPAddress, where the S\_LPAd plays the role of S\_IPA.

### 4.2.10 ES10b (IPA -- eUICC): PrepareDownload

#### 4.2.10.1 Conformance Requirements

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.10.2 Test Cases

##### 4.2.10.2.1 TC\_eUICC\_ES10b.PrepareDownloadNIST

This ES10b.PrepareDownload function is identical to the ES10b.PrepareDownload function defined in section 5.7.5 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: Without Confirmation Code

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Without Confirmation Code* defined in 4.2.10.2.1 TC\_eUICC\_ES10b.PrepareDownloadNIST, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: With Confirmation Code

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: With Confirmation Code* defined in 4.2.10.2.1 TC\_eUICC\_ES10b.PrepareDownloadNIST, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Nominal: With an unknown otPK.EUICC.ECKA

The purpose of this test is to verify that the eUICC does not use the one-time key pair given by the SM-DP+ when its value does not correspond to a stored one-time key pair. In this case, the eUICC SHALL generate a new set of key.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: With an unknown otPK.EUICC.ECKA* defined in 4.2.10.2.1 TC\_eUICC\_ES10b.PrepareDownloadNIST, where the S\_LPAd plays the role of S\_IPA.

##### 4.2.10.2.2 TC\_eUICC\_ES10b.PrepareDownloadBRP

This ES10b.PrepareDownload function is identical to the ES10b.PrepareDownload function defined in section 5.7.5 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: Without Confirmation Code

This test sequence SHALL be the same as SGP.23 [8] - the Test Sequence #01 defined in section 4.2.10.2.1 – TC\_eUICC\_ES10b.PrepareDownloadNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #02 Nominal: With Confirmation Code

This test sequence SHALL be the same as SGP.23 [8] - the Test Sequence #02 defined in section 4.2.10.2.1 – TC\_eUICC\_ES10b.PrepareDownloadNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #03 Nominal: With an unknown otPK.EUICC.ECKA

This test sequence SHALL be the same as SGP.23 [8] - the Test Sequence #03 defined in section 4.2.10.2.1 – TC\_eUICC\_ES10b.PrepareDownloadNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

##### 4.2.10.2.3 TC\_eUICC\_ES10b.PrepareDownloadFRP

This test case is defined as FFS and not applicable for this version of test specification.

##### 4.2.10.2.4 TC\_eUICC\_ES10b.PrepareDownloadErrorCases

This ES10b.PrepareDownload function is identical to the ES10b.PrepareDownload function defined in section 5.7.5 of SGP.22 [4], where the IPA plays the role of LPA.

Sequence #01 Error: No Hashed Confirmation Code but with Confirmation Code Required Flag set to TRUE

This test sequence is the same as SGP.23 [8] - the *Sequence #01 Error: No Hashed Confirmation Code but with Confirmation Code Required Flag set to TRUE* defined in 4.2.10.2.4 TC\_eUICC\_ES10b.PrepareDownloadErrorCases, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Error: With incorrect CERT.DPpb.ECDSA (i.e. invalid signature)

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Error: With incorrect CERT.DPpb.ECDSA (i.e. invalid signature)* defined in 4.2.10.2.4 TC\_eUICC\_ES10b.PrepareDownloadErrorCases, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Error: CERT.DPpb.ECDSA and CERT.DPauth.ECDSA not belonging to the same entity

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Error: CERT.DPpb.ECDSA and CERT.DPauth.ECDSA not belonging to the same entity* defined in 4.2.10.2.4 TC\_eUICC\_ES10b.PrepareDownloadErrorCases, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #04 Error: With invalid SM-DP+ signature

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Error: With invalid SM-DP+ signature* defined in 4.2.10.2.4 TC\_eUICC\_ES10b.PrepareDownloadErrorCases, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #05 Error: With invalid Transaction ID

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Error: With invalid Transaction ID* defined in 4.2.10.2.4 TC\_eUICC\_ES10b.PrepareDownloadErrorCases, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #06 Error: SM-DP+ has not been previously authenticated

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Error: SM-DP+ has not been previously authenticated* defined in 4.2.10.2.4 TC\_eUICC\_ES10b.PrepareDownloadErrorCases, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #07 Error: Unsupported curve

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Error: Unsupported curve* defined in 4.2.10.2.4 TC\_eUICC\_ES10b.PrepareDownloadErrorCases, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #08 Error: Invalid Certificate Role OID

The purpose of this sequence is to make sure that the eUICC refuses any SM-DP+ Certificate for Profile Package Binding that does not indicate “id-rspRole-dp-pb” in its extension for Certificate Policies.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Error: Invalid Certificate Role OID* defined in 4.2.10.2.4 TC\_eUICC\_ES10b.PrepareDownloadErrorCases, where the S\_LPAd plays the role of S\_IPA.

### 4.2.11 ES10b (IPA -- eUICC): LoadBoundProfilePackage

#### 4.2.11.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.11.2 Test Cases

##### 4.2.11.2.1 TC\_eUICC\_ES10b.LoadBoundProfilePackageNIST

This ES10b.LoadBoundProfilePackage function is identical to the ES10b.LoadBoundProfilePackage function defined in section 5.7.6 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: By using S-ENC and S-MAC

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: By using S-ENC and S-MAC* defined in 4.2.11.2.1 TC\_eUICC\_ES10b.LoadBoundProfilePackageNIST, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: By using PPK-ENC and PPK-MAC

The purpose of this test is to download the PROFILE\_OPERATIONAL1 by using a new set of random session keys: PPK-ENC, PPK-MAC and Initial MAC chaining value.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: By using PPK-ENC and PPK-MAC* defined in 4.2.11.2.1 TC\_eUICC\_ES10b.LoadBoundProfilePackageNIST, where the S\_LPAd plays the role of S\_IPA.

##### 4.2.11.2.2 TC\_eUICC\_ES10b.LoadBoundProfilePackageBRP

This ES10b.LoadBoundProfilePackage function is identical to the ES10b.LoadBoundProfilePackage function defined in section 5.7.6 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: By using S-ENC and S-MAC

This test sequence SHALL be the same as SGP.23 [8] - the Test Sequence #01 defined in section 4.2.11.2.1 – TC\_eUICC\_ES10b.LoadBoundProfilePackageNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #02 Nominal: By using PPK-ENC and PPK-MAC

This test sequence SHALL be the same as SGP.23 [8] - the *Test Sequence #02 Nominal* defined in section 4.2.11.2.1 – TC\_eUICC\_ES10b. LoadBoundProfilePackageNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

##### 4.2.11.2.3 TC\_eUICC\_ES10b.LoadBoundProfilePackageFRP

This test case is defined as FFS and not applicable for this version of test specification.

##### 4.2.11.2.4 TC\_eUICC\_ES10b.LoadBoundProfilePackage\_ErrorCases

This ES10b.LoadBoundProfilePackage function is identical to the ES10b.LoadBoundProfilePackage function defined in section 5.7.6 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Error: Unrecognized leading tag in BPP

The purpose of this test is to ensure that the eUICC rejects any BPP segment with an unrecognized leading tag during Profile download. In such case, the eUICC SHALL return a SW of 0x6A88 and SHALL not discard the download session state.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: Unrecognized leading tag in BPP* defined in 4.2.11.2.4 TC\_eUICC\_ES10b.LoadBoundProfilePackage\_ErrorCases, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Error: GetEUICCChallenge during BPP loading

The purpose of this test is to ensure that the eUICC accepts an ES10b.GetEUICCChallenge request indicating the start of a new RSP session while a BPP is loaded.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Error: GetEUICCChallenge during BPP loading* defined in 4.2.11.2.4 TC\_eUICC\_ES10b.LoadBoundProfilePackage\_ErrorCases, where the S\_LPAd plays the role of S\_IPA.

### 4.2.12 ES10b (LPA -- eUICC): GetEUICCChallenge

#### 4.2.12.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.12.2 Test Cases

##### 4.2.12.2.1 TC\_eUICC\_ES10b.GetEUICCChallenge

This function is identical to the ES10b.GetEUICCChallenge function defined in section 5.7.7 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal* defined in 4.2.12.2.1 TC\_eUICC\_ES10b.GetEUICCChallenge, where the S\_LPAd plays the role of S\_IPA.

### 4.2.13 ES10b (IPA -- eUICC): GetEUICCInfo

#### 4.2.13.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.13.2 Test Cases

##### 4.2.13.2.1 TC\_eUICC\_ES10b.GetEUICCInfo1

This function is identical to the ES10b.GetEUICCInfo function defined in section 5.7.8 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal* defined in 4.2.13.2.1 TC\_eUICC\_ES10b.GetEUICCInfo1, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: GetEUICCInfo call after GetEUICCChallenge

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: GetEUICCInfo call after GetEUICCChallenge* defined in 4.2.13.2.1 TC\_eUICC\_ES10b.GetEUICCInfo1, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Nominal: GetEUICCInfo1 call after AuthenticateServer

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal* defined in 4.2.13.2.1 TC\_eUICC\_ES10b.GetEUICCInfo1, where the S\_LPAd plays the role of S\_IPA.

##### 4.2.13.2.2 TC\_eUICC\_ES10b.GetEUICCInfo2\_RSP\_Integrated\_eUICC

NOTE: This test case is FFS.

### 4.2.14 ES10b (IPA -- eUICC): ListNotification

#### 4.2.14.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.14.2 Test Cases

Throughout all the ListNotification test cases the maximum number of Notifications simultaneously tested has been set as to two as there is not minimum defined in SGP.21 [5], SGP.22 [4], SGP.31 [2] or SGP.32 [3] for the number of Notifications that can be stored by the eUICC.

##### 4.2.14.2.1 TC\_eUICC\_ES10b.ListNotification

### 4.2.15 ES10b (IPA -- eUICC): RetrieveNotificationsList

#### 4.2.15.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.15.2 Test Cases

##### 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList

Test Sequence #01 Nominal: Retrieve by Sequence Number for Install Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal Retrieve by Sequence Number for Install Notification Step Direction Sequence / Description Expected result REQ* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: Retrieve by Sequence Number for Enable Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Retrieve by Sequence Number for Enable Notification* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Nominal: Retrieve by Sequence Number for Disable Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Retrieve by Sequence Number for Disable Notification* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #04 Nominal: Retrieve by Sequence Number for Delete Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: Retrieve by Sequence Number for Delete Notification* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #05 Nominal: Retrieve by Sequence Number for Two Install (PIR) Notifications with different Notification Addresses

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Nominal: Retrieve by Sequence Number for Two Install (PIR) Notifications with different Notification Addresses* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #06 Nominal: Retrieve by Sequence Number for Install (PIR) and Enable Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Nominal: Retrieve by Sequence Number for Install (PIR) and Enable Notifications* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #07 Nominal: Retrieve by Sequence Number for Disable and Delete Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Nominal: Retrieve by Sequence Number for Disable and Delete Notifications* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #08 Nominal: Retrieve by Sequence Number for Install (OtherSignedNotification) and Enable Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Nominal: Retrieve by Sequence Number for Install (OtherSignedNotification) and Enable Notifications* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #09 Nominal: Retrieve by Sequence Number for Enable and Install (PIR) notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #09 Nominal: Retrieve by Sequence Number for Enable and Install (PIR) notifications* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #10 Nominal: Retrieve Sequence Numbers that are not present

This test sequence is the same as SGP.23 [8] - the *Test Sequence #10 Nominal: Retrieve Sequence Numbers that are not present* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #11 Nominal: Retrieve by Notification Type for Install Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #11 Nominal: Retrieve by Notification Type for Install Notifications* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #12 Nominal: Retrieve by Notification Type for Enable Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #12 Nominal: Retrieve by Notification Type for Enable Notification* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #13 Nominal: Retrieve by Notification Type for Disable Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #13 Nominal: Retrieve by Notification Type for Disable Notification* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #14 Nominal: Retrieve by Notification Type for Delete Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #14 Nominal: Retrieve by Notification Type for Delete Notification* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA

Test Sequence #15 Nominal: Retrieve by Notification Type for Two Install (PIR) Notifications with different Notification Addresses

This test sequence is the same as SGP.23 [8] - the *Test Sequence #15 Nominal: Retrieve by Notification Type for Two Install (PIR) Notifications with different Notification Addresses* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA

Test Sequence #16 Nominal: Retrieve by Notification Type for Install (PIR) and Enable Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #16 Nominal: Retrieve by Notification Type for Install (PIR) and Enable Notifications* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA

Test Sequence #17 Nominal: Retrieve by Notification Type for Disable and Delete Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #17 Nominal: Retrieve by Notification Type for Disable and Delete Notifications* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA

Test Sequence #18 Nominal: Retrieve by Notification Type for Install (OtherSignedNotification) and Enable Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #18 Nominal: Retrieve by Notification Type for Install (OtherSignedNotification) and Enable Notifications* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA

Test Sequence #19 Nominal: Retrieve by Notification Type for Enable and Install (PIR) notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #19 Nominal: Retrieve by Notification Type for Enable and Install (PIR) notifications* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA

Test Sequence #20 Nominal: Retrieve by Notification Type for No Notifications available

This test sequence is the same as SGP.23 [8] - the *Test Sequence #20 Nominal: Retrieve by Notification Type for No Notifications available* defined in 4.2.15.2.1 TC\_eUICC\_ES10b.RetrieveNotificationsList, where the S\_LPAd plays the role of S\_IPA

### 4.2.16 ES10b (IPA -- eUICC): RemoveNotificationFromList

#### 4.2.16.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.16.2 Test Cases

Throughout all the RemoveNotificationFromList test cases the maximum number of Notifications simultaneously tested has been set as to two as there is no minimum defined in SGP.21 [5], SGP.22 [4], SGP.31 [2] or SGP.32 [3] for the number of Notifications that can be stored by the eUICC.

The rule specified in section 4.2.15.2 explaining the way to distinguish a ProfileInstallationResult from an OtherSignedNotification for installation also applies for the test cases defined below.

##### 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromList

This function is identical to the ES10b.RemoveNotificationFromList function defined in section 5.7.11 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: Install Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Install Notification* defined in 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromListwhere the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: Enable Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: Enable Notification* defined in 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromListwhere the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Nominal: Disable Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Disable Notification* defined in 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromListwhere the S\_LPAd plays the role of S\_IPA.

Test Sequence #04 Nominal: Delete Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: Delete Notification* defined in 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromListwhere the S\_LPAd plays the role of S\_IPA.

Test Sequence #05 Nominal: Two Install (PIR) Notifications with different Notification Addresses

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Nominal: Two Install (PIR) Notifications with different Notification Addresses* defined in 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromListwhere the S\_LPAd plays the role of S\_IPA.

Test Sequence #06 Nominal: Install (PIR) and Enable Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Nominal: Install (PIR) and Enable Notifications* defined in 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromListwhere the S\_LPAd plays the role of S\_IPA.

Test Sequence #07 Nominal: Disable and Delete Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Nominal: Disable and Delete Notifications* defined in 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromListwhere the S\_LPAd plays the role of S\_IPA.

Test Sequence #08 Nominal: Install (OtherSignedNotification) and Enable Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Nominal: Install (OtherSignedNotification) and Enable Notifications* defined in 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromListwhere the S\_LPAd plays the role of S\_IPA.

Test Sequence #09 Nominal: Enable and Install (PIR) notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #09 Nominal: Enable and Install (PIR) notifications* defined in 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromListwhere the S\_LPAd plays the role of S\_IPA.

Test Sequence #10 Nominal: Removing Sequence Numbers that are not present

This test sequence is the same as SGP.23 [8] - the *Test Sequence #10 Nominal: Removing Sequence Numbers that are not present* defined in 4.2.16.2.1 TC\_eUICC\_ES10b.RemoveNotificationFromListwhere the S\_LPAd plays the role of S\_IPA.

##### 4.2.16.2.2 VOID

### 4.2.17 VOID

### 4.2.18 ES10b (IPA -- eUICC): AuthenticateServer

#### 4.2.18.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.18.2 Test Cases

##### 4.2.18.2.1 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_NIST

***Test Sequence #01 Nominal: Without MatchingID in CtxParams1***

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Without MatchingID in CtxParams1* defined in 4.2.18.2.1 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_NIST where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: With MatchingID in CtxParams1

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: With MatchingID in CtxParams1* defined in 4.2.18.2.1 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_NIST where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Nominal: With IMEI in Device Capabilities

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: With IMEI in Device Capabilities* defined in 4.2.18.2.1 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_NIST where the S\_LPAd plays the role of S\_IPA.

##### 4.2.18.2.2 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_BRP

Test Sequence #01 Nominal: Without MatchingID in CtxParams1

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Without MatchingID in CtxParams1* defined in 4.2.18.2.2 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_BRP where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: With MatchingID in CtxParams1

This test sequence is the same as SGP.23 [8] - the Test Sequence #02 Nominal: With MatchingID in CtxParams1 defined in 4.2.18.2.2 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_BRP where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Nominal: With IMEI in Device Capabilities

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: With IMEI in Device Capabilities* defined in 4.2.18.2.2 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_BRP where the S\_LPAd plays the role of S\_IPA.

##### 4.2.18.2.3 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_FRP

This test case is defined as FFS and not applicable for this version of test specification.

##### 4.2.18.2.4 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_ErrorCases

This function is identical to the ES10b.AuthenticateServer function defined in section 5.7.13 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Error: With Incorrect SM-DPauth certificate (i.e. invalid signature)

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: With Incorrect SM-DPauth certificate (i.e. invalid signature)* defined in 4.2.18.2.4 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Error: With Invalid SM-DP+ Signature

This test sequence is the same as SGP.23 [8] - the Test Sequence #02 Error: With Invalid SM-DP+ Signaturedefined in 4.2.18.2.4 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Error: Unsupported Curve

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Error: Unsupported Curve* defined in 4.2.18.2.4 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #04 Error: eUICC Challenge Mismatch

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Error: eUICC Challenge Mismatch* defined in 4.2.18.2.4 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #05 Error: Unknown CI PK

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Error: Unknown CI PK* defined in 4.2.18.2.4 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #06 Error: Invalid Certificate Role OID

The purpose of this sequence is to make sure that the eUICC refuses any SM-DP+ Certificate for authentication that does not indicate “id-rspRole-dp-auth” in its extension for Certificate Policies.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Error: Invalid Certificate Role OID* defined in 4.2.18.2.4 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #07 Error: No RSP session on-going

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Error: No RSP session on-going* defined in 4.2.18.2.4 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DP+\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

##### 4.2.18.2.5 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_BRP

This function is identical to the ES10b.AuthenticateServer function defined in section 5.7.13 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: With EventID in CtxParams1

This test sequence is the same as SGP.23 [8] - *the Test Sequence #01 Nominal: With EventID in CtxParams1* defined in 4.2.18.2.5 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_BRP where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: With IMEI and EventID in Device Capabilities

This test sequence is the same as SGP.23 [8] - *Test Sequence #02 Nominal: With IMEI and EventID in Device Capabilities* defined in 4.2.18.2.5 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_BRP where the S\_LPAd plays the role of S\_IPA.

##### 4.2.18.2.6 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_NIST

This function is identical to the ES10b.AuthenticateServer function defined in section 5.7.13 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: With EventID in CtxParams1

This test sequence is the same as SGP.23 [8] – *the Test Sequence #01 Nominal: With EventID in CtxParams1* defined in 4.2.18.2.6 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_NIST where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: With IMEI and EventID in Device Capabilities

This test sequence is the same as SGP.23 [8] –the *Test Sequence #02 Nominal: With IMEI and EventID in Device Capabilities* defined in 4.2.18.2.6 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_NIST where the S\_LPAd plays the role of S\_IPA.

##### 4.2.18.2.7 VOID

##### 4.2.18.2.8 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_ErrorCases

This function is identical to the ES10b.AuthenticateServer function defined in section 5.7.13 of SGP.22 [4], where the IPA plays the role of LPA

Test Sequence #01 Error: With Incorrect SM-DSauth certificate (i.e. invalid signature)

This test sequence is the same as SGP.23 [8] –the *Test Sequence #01 Error: With Incorrect SM-DSauth certificate (i.e. invalid signature)* defined in 4.2.18.2.8 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Error: With Invalid SM-DS Signature

This test sequence is the same as SGP.23 [8] –the *Test Sequence #02 Error: With Invalid SM-DS Signature* defined in 4.2.18.2.8 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Error: Unsupported Curve

This test sequence is the same as SGP.23 [8] –the *Test Sequence #03 Error: Unsupported Curve* defined in 4.2.18.2.8 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #04 Error: eUICC Challenge Mismatch

This test sequence is the same as SGP.23 [8] –the *Test Sequence #04 Error: eUICC Challenge Mismatch* defined in 4.2.18.2.8 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #05 Error: Unknown CI PK

This test sequence is the same as SGP.23 [8] –the *Test Sequence #05 Error: Unknown CI PK* defined in 4.2.18.2.8 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #06 Error: No RSP session on-going

This test sequence is the same as SGP.23 [8] –the *Test Sequence #06 Error: No RSP session on-going* defined in 4.2.18.2.8 TC\_eUICC\_ES10b.AuthenticateServer\_SM-DS\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

### 4.2.19 ES10b (LPA -- eUICC): CancelSession

#### 4.2.19.1 Conformance Requirements

This function is identical to the ES10b.CancelSession function defined in section 5.7.14 of SGP.22 [4], where the IPA plays the role of LPA, with the following additional eUICC behaviour:

* The eUICC SHALL revoke any grant to automatically enable a Profile.

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.19.2 Test Cases

##### 4.2.19.2.1 TC\_eUICC\_ES10b.CancelSessionNIST

This function is identical to the ES10b.CancelSession function defined in section 5.7.14 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: VOID

***Test Sequence #02 Nominal: VOID***

Test Sequence #03 Nominal: VOID

Test Sequence #04 Nominal: PPR not allowed

The RSP session is terminated because the IPAd detected that PPR(s) set in the Profile Metadata is/are not allowed.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: PPR not allowed* defined in 4.2.19.2.1 TC\_eUICC\_ES10b.CancelSessionNIST where the S\_LPAd plays the role of S\_IPA.

Test Sequence #05 Nominal: Metadata Mismatch

The RSP session is terminated because the IPAd detected that the Profile Metadata in the response to "ES9+.AuthenticateClient" does not match the Profile Metadata in the Bound Profile Package.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Nominal: Metadata Mismatch* defined in 4.2.19.2.1 TC\_eUICC\_ES10b.CancelSessionNIST where the S\_LPAd plays the role of S\_IPA.

Test Sequence #06 Nominal: BPP Parsing Error

The RSP session is terminated because the LPAd has encountered an error while parsing the Bound Profile Package received from the SM-DP+.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Nominal: Metadata Mismatch* defined in 4.2.19.2.1 TC\_eUICC\_ES10b.CancelSessionNIST where the S\_LPAd plays the role of S\_IPA.

Test Sequence #07 Nominal: Load BPP Execution Error

The RSP session is terminated because the LPAd has encountered an error while installing the Bound Profile Package received from the SM-DP+.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Nominal: Load BPP Execution Error* defined in 4.2.19.2.1 TC\_eUICC\_ES10b.CancelSessionNIST where the S\_LPAd plays the role of S\_IPA.

Test Sequence #08 Nominal: Undefined Reason

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Nominal: Undefined Reason* defined in 4.2.19.2.1 TC\_eUICC\_ES10b.CancelSessionNIST where the S\_LPAd plays the role of S\_IPA.

##### 4.2.19.2.2 TC\_eUICC\_ES10b.CancelSessionBRP

In these test sequences, once the RSP session has been cancelled, verifications are performed in order to check that it is neither possible to execute the Download Confirmation procedure nor to execute the Common Mutual Authentication procedure by referring to the cancelled TransactionID.

Test Sequence #01 Nominal: VOID

Test Sequence #02 Nominal: VOID

Test Sequence #03 Nominal: VOID

The RSP session is delayed because the End User does not confirm the download of the Profile within the timeout interval defined by the LPAd.

Test Sequence #04 Nominal: PPR not allowed

This test sequence SHALL be the same as the Test Sequence #04 defined in section 4.2.19.2.1 – TC\_eUICC\_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #05 Nominal: Metadata Mismatch

This test sequence SHALL be the same as the Test Sequence #05 defined in section 4.2.19.2.1 – TC\_eUICC\_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #06 Nominal: BPP Parsing Error

This test sequence SHALL be the same as the Test Sequence #06 defined in section 4.2.19.2.1 – TC\_eUICC\_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #07 Nominal: Load BPP Execution Error

This test sequence SHALL be the same as the Test Sequence #07 defined in section 4.2.19.2.1 – TC\_eUICC\_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #08 Nominal: Undefined Reason

This test sequence SHALL be the same as the Test Sequence #08 defined in section 4.2.19.2.1 – TC\_eUICC\_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

##### 4.2.19.2.3 TC\_eUICC\_ES10b.CancelSessionFRP

This test case is defined as FFS and not applicable for this version of test specification.

##### 4.2.19.2.4 TC\_eUICC\_ES10b.CancelSession\_ErrorCase

Test Sequence #01 Error: No on-going RSP session

On receiving a CancelSession request whereas there is no on-going RSP session, the eUICC SHALL return an error code.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: No on-going RSP session* defined in 4.2.19.2.4 TC\_eUICC\_ES10b.CancelSession\_ErrorCasewhere the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Error: Invalid Transaction ID

On receiving a CancelSession request with a TransactionID different from the on-going one, the eUICC SHALL not discard the current RSP session and return an error code.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Error: Invalid Transaction ID* defined in 4.2.19.2.4 TC\_eUICC\_ES10b.CancelSession\_ErrorCasewhere the S\_LPAd plays the role of S\_IPA.

##### 4.2.19.2.5 VOID

##### 4.2.19.2.6 VOID

### 4.2.20 ES10b (LPA -- eUICC): eUICCMemoryReset

#### 4.2.20.1 Conformance Requirements

This function is the same from the ES10c.eUICCMemoryReset function defined in SGP.22 [4] with the exception of new Nominal and error cases that are for further study.

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.20.2 Test Cases

##### 4.2.20.2.1 TC\_eUICC\_ES10b.eUICCMemoryReset

Test Sequence #01 Nominal: Reset All Operational Profiles

This test sequence is the same as SGP.23 [8] – the *Test Sequence #01 Nominal: Reset All Operational Profiles (without Enabled Profile)* defined in 4.2.24.2.1 TC\_eUICC\_ES10c.eUICCMemoryReset where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: Reset Field Loaded Test Profiles

This test sequence is the same as SGP.23 [8] – the *Test Sequence #02 Nominal: Reset Field Loaded Test Profile)* defined in 4.2.24.2.1 TC\_eUICC\_ES10c.eUICCMemoryReset where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Nominal: Reset the Default SM-DP+ Address only

This test sequence is the same as SGP.23 [8] – the *Test Sequence #03 Nominal: Reset the Default SM-DP+ Address only* defined in 4.2.24.2.1 TC\_eUICC\_ES10c.eUICCMemoryReset where the S\_LPAd plays the role of S\_IPA.

Test Sequence #04 Nominal: EIM Reset

NOTE: This test sequence is FFS.

Test Sequence #05 Nominal: Auto Enable Reset

NOTE: This test sequence is FFS.

##### 4.2.24.2.2 TC\_eUICC\_ES10b.eUICCMemoryReset\_ErrorCases

Test Sequence #01 Error: eUICC Memory Reset while there is no Profile to delete

This test sequence is the same as SGP.23 [8] – the *Test Sequence #01 Error: eUICC Memory Reset while there is no Profile to delete* defined in 4.2.24.2.2 TC\_eUICC\_ES10c.eUICCMemoryReset\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Error: eUICC Memory Reset – Cat Busy

This test sequence is the same as SGP.23 [8] – the *Test Sequence #02 Error: eUICC Memory Reset – Cat Busy* defined in 4.2.24.2.2 TC\_eUICC\_ES10c.eUICCMemoryReset\_ErrorCases where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Error: EIM Reset – Nothing to delete

NOTE: This test sequence is FFS.

Test Sequence #04 Error: EIM Reset – EIM Reset not supported

NOTE: This test sequence is FFS.

Test Sequence #05 Error: Auto Enable Reset – Auto Enable Reset not supported

NOTE: This test sequence is FFS.

### 4.2.21 ES10b (LPA -- eUICC): GetEID

This function is identical to the ES10c.GetEID function defined in section 5.7.20 of SGP.22 [4], where the IPA plays the role of LPA.

#### 4.2.21.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.21.2 Test Cases

##### 4.2.21.2.1 TC\_eUICC\_ES10b.GetEID

Test Sequence #01 Nominal

The purpose of this test is to ensure that it is possible to retrieve the EID. This test sequence is the same as the *Test Sequence #01 Nominal* defined in SGP.23 [8] - 4.2.25.2.1 TC\_eUICC\_ES10c.GetEID, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Error

The purpose of this test is to ensure that if the provided tagList is invalid or unsupported, the eUICC returns an error status word.

This test sequence is the same as the *Test Sequence #02 Error* defined in SGP.23 [8] - 4.2.25.2.1 TC\_eUICC\_ES10c.GetEID, where the S\_LPAd plays the role of S\_IPA.

### .4.2.22 ES10b (LPA -- eUICC): GetRAT

#### 4.2.22.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.22.2 Test Cases

##### 4.2.22.2.1 TC\_eUICC\_ES10b.GetRAT

This eUICC\_ES10b.GetRAT function is identical to the ES10b.GetRAT function defined in in SGP.23 [8] - section 5.7.22 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: Get Default RAT

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Only Root SM-DS Address* defined in 4.2.27.2.1 TC\_eUICC\_ES10b.GetRAT, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: With additional PPARs

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: With additional PPARs* defined in 4.2.27.2.1 TC\_eUICC\_ES10b.GetRAT, where the S\_LPAd plays the role of S\_IPA.

### 4.2.23 ES10b (LPA -- eUICC): GetProfilesInfo

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

This function is identical to the ES10c.GetProfilesInfo function defined in section 5.7.15 of SGP.22 [4], where the IPA plays the role of LPA.

#### 4.2.23.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.23.2 Test Cases

##### 4.2.23.2.1 TC\_eUICC\_ES10b.GetProfilesInfo

Test Sequence #01 Nominal: Get All Profiles

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #01 Nominal: Get All Profiles - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #02 Nominal: Get Profile by ICCID

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #02 Nominal: Get Profile by ICCID - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #03 Nominal: Get Profile by AID

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #03 Nominal: Get Profile by AID - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #04 Nominal: Get All Operational Profiles

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #04 Nominal: Get All Operational Profiles Test Sequence #02 Nominal: Get Profile by ICCID 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #05 Nominal: Get Profile ICCID list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #05 Nominal: Get Profile ICCID list defined 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #06 Nominal: Get Profile AID list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #06 Nominal: Get Profile AID list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #07 Nominal: Get Profile Nickname list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #07 Nominal: Get Profile Nickname list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #08 Nominal: Get Profile SP Name list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #08 Nominal: Get Profile SP Name list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #09 Nominal: Get Profile Name list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #09 Nominal: Get Profile Name list defined - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #10 Nominal: Get Profile Icon list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #10 Nominal: Get Profile Icon list defined - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #11 Nominal: Get Profile Owner list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #11 Nominal: Get Profile Owner list by ICCID - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #12 Nominal: Get Profile SM-DP+ proprietary data list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #12 Nominal: Get Profile SM-DP+ proprietary data list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #13 Nominal: Get Profile ICCID and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #13 Nominal: Get Profile ICCID and State list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #14 Nominal: Get Profile Nickname and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #14 Nominal: Get Profile Nickname and State list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #15 Nominal: Get Profile Icon and Icon Type list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #15 Nominal: Get Profile Icon and Icon Type list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #16 Nominal: Get Profile Icon and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #16 Nominal: Get Profile Icon and State list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #17 Nominal: Get Operational Profile ICCID and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #17 Nominal: Get Operational Profile ICCID and State list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #18 Nominal: Get Operational Profile Nickname and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #18 Nominal: Get Operational Profile Nickname and State list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #19 Nominal: Get Operational Profile Icon and Icon type list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #19 Nominal: Get Operational Profile Icon and Icon type list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #20 Nominal: Get Operational Profile Icon and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #20 Nominal: Get Operational Profile Icon and State list - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #21 Nominal: Get Profile State of the defined Profile

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #21 Nominal: Get Profile State of the defined Profile - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #22 Nominal: Get Profile Icon Type of the defined Profile

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #22 Nominal: Get Profile Icon Type of the defined Profile - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #23 Nominal: Get Profile Class of the defined Profile

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #23 Nominal: Get Profile Class of the defined Profile - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #24 Nominal: Get Notification Configuration of the defined Profile

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #24 Nominal: Get Notification Configuration of the defined Profile - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #25 Nominal: Get Profile Policy Rules of the defined Profile

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #25 Nominal: Get Profile Policy Rules of the defined Profile - 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo, where the S\_LPAd plays the role of S\_IPA.

Test Sequence #26 Nominal: Get empty Profile list. No Profile installed

***This test sequence is the same as defined*** in SGP.23 [8] - ***the*** Test Sequence #26 Nominal: Get empty Profile list. No Profile installed ***- 4.2.20.2.1 TC\_eUICC\_ES10c.GetProfilesInfo,*** where the S\_LPAd plays the role of S\_IPA.

### 4.2.24 ES10b(IPA – eUICC) LoadEUICCPackage

This function executes a eUICC Package.

#### 4.2.24.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.24.2 Test Cases

##### 4.2.24.2.1 TC\_eUICC\_ES10b. LoadEUICCPackage

The Nominal Test Cases for LoadeUICCPackage function are covered by the defined test cases for ESep interface (Section from 4.2.31 to 4.2.39).**4.2.24.2.2 TC\_eUICC\_ES10b. LoadEUICCPackage\_ErrorCases**

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

Test Sequence #01 Error: eimID is unknown

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM(  TEST\_EIM\_ID2,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA1\_UPDATE\_COUNTER,  S\_EIM\_SIGNATURE  )  ) | SW=0x9000 with response data:  MTD\_EUICC\_PKG\_ERROR\_UNSIGNED (  #TEST\_EIM\_ID2  ) |

Test Sequence #02 Error: Signature Invalid

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA1\_UPDATE\_COUNTER,  S\_EIM\_SIGN\_RANDOM  )  ) | SW=0x9000 with response data:  MTD\_EUICC\_PKG\_ERROR\_UNSIGNED (  #TEST\_EIM\_ID1  ) |

Test Sequence #03 Error: EID does not match/invalid

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM(  TEST\_EIM\_ID1,  EID2,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA1\_UPDATE\_COUNTER,  S\_EIM\_SIGN\_RANDOM  )  ) | SW=0x9000 with response data:  MTD\_EUICC\_PKG\_ ERROR\_SIGNED (  #LOAD\_EUICC\_PKG\_ ERR\_EID  ) |

Test Sequence #04 Error: Counter value less than the counter value storted on the eUICC

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_LOW\_CNTR\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA1\_UPDATE\_COUNTER,  S\_EIM\_SIGNATURE)  ) | SW=0x9000 with response data:  MTD\_EUICC\_PKG\_ ERROR\_SIGNED (  #LOAD\_EUICC\_PKG\_ ERR\_LOW\_CNTR  ) |

Test Sequence #05 Error: PSMO invalid due to the size limit of the response data

This test sequence of this error is not defined for this version of the specification.

Test Sequence #06 Error: eCO invalid or unsupported

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PKG\_REQUEST(  #TEST\_EIM\_ID1,  NO\_PARAM,  <S\_TRANSACTION\_ID>,  euiccPackage #LOAD\_EUICC\_PKG\_INVALID\_ECO  )  )  ) | SW=0x9000 with response data:  MTD\_EUICC\_PKG\_RESULT(  (  #LOAD\_EUICC\_PKG\_ ERR\_ UNKNOWN\_CMD  ) |

Test Sequence #07 Error: PSMO invalid or unsupported

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PKG\_REQUEST(  #TEST\_EIM\_ID1,  NO\_PARAM,  <S\_TRANSACTION\_ID>,  euiccPackage #LOAD\_EUICC\_PKG\_INVALID\_PSMO  )  )  ) | SW=0x9000 with response data:  MTD\_EUICC\_PKG\_RESULT(  (  #LOAD\_EUICC\_PKG\_ ERR\_ UNKNOWN\_CMD) |

### 4.2.25 ES10b (IPA -- eUICC): AddInitialEim

This function is used by the IPAd to store eIM Configuration Data to the eUICC in case the eUICC does not contain any eIM Configuration Data.

#### 4.2.25.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.24.2 Test Cases

##### 4.2.25.2.1 TC\_eUICC\_ES10b.AddInitialEim

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | The eUICC does not contain any eIM Configuration Data. |

Test Sequence #01 Nominal Case

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ADD\_INITIAL\_EIM\_SINGLE(  #EIM\_CONFIG\_DATA1  )  ) | response AddInitialEimResponse ::= {  #ADD\_INITIAL\_EIM\_RES\_OK}  SW=0x9000 |

##### 4.2.25.2.1 TC\_eUICC\_ES10b.AddInitialEim\_ErrorCases

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | The eUICC does not contain any eIM Configuration Data. |

Test Sequence #01 Error: Unsigned Eim Config Disallowed

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ADD\_INITIAL\_EIM\_SINGLE(  #EIM\_CONFIG\_DATA1  )  ) | response AddInitialEimResponse ::= {  #ADD\_INITIAL\_EIM\_RES\_OK  }  SW=0x9000 |
| 2 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ADD\_INITIAL\_EIM\_SINGLE(  #EIM\_CONFIG\_DATA2  )  ) | response AddInitialEimResponse ::= {  #ADD\_INITIAL\_EIM\_ ERR\_EIM\_CONF  }  SW=0x9000 |

Test Sequence #02 Error: Insufficient Memory

Note: This error cannot be tested because there is no interoperable way at the time of writing to set a specified amount of available memory in an eUICC

Test Sequence #03 Error: Unknown CI PK

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ADD\_INITIAL\_EIM\_SINGLE(  #EIM\_CONFIG\_DATA2\_ADD\_UKNOWN\_EUICC\_CI\_PKID  )  ) | response AddInitialEimResponse ::= {  #ADD\_INITIAL\_EIM\_ERR\_UNKNOWN\_CI  }  SW=0x9000 |

Test Sequence #04 Error: Counter Value out of Range

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ADD\_INITIAL\_EIM\_SINGLE(  #EIM\_CONFIG\_DATA2\_ADD\_COUNTER\_OUT\_OF\_RANGE  )  ) | response AddInitialEimResponse ::= {  #ADD\_INITIAL\_EIM\_ERR\_CNTR\_OUT\_OF\_RANGE  }  SW=0x9000 |

Test Sequence #05 Error: Invalid Association Token

The test sequence for this rrror case is not defined in this version of the specification.

### 4.2.26 ES10b (IPA -- eUICC): GetCerts

This function is used by the IPA to retrieve the eUICC Certificate and the EUM Certificate from the eUICC. This function can be used at any time by the IPA.

#### 4.2.26.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.26.2 Test Cases

##### 4.2.26.2.1 TC\_eUICC\_ES10b.GetCertificates

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |
| eUICC | Set the <EUICC\_CI\_PK\_ID\_TO\_BE\_USED> to the CI Key ID based on NIST P-256 curve |

Test Sequence #01 Nominal Case: euiccCiPKId set

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_CERTS\_REQUEST (  NO\_PARAM  )  ) | SW=0x9000 with response data:  GetCertsResponse ::= {  #GET\_CERTS\_RESULT\_OK  }  Verify that the EUM’s Certificate #CERT\_EUM\_SIG contains an EUM’s Public Key #PK\_EUM\_SIG based on NIST P-256 curve  Verify that the eUICC’s Certificate #CERT\_EUICC\_SIG contains an eUICC’s Public Key #PK\_EUICC\_SIG based on NIST P-256 curve |

Test Sequence #02 Nominal Case with euiccCiPKId set

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_CERTS\_REQUEST (  #CI\_KEY\_ID\_BP256  )  ) | SW=0x9000 with response data:  GetCertsResponse ::= {  #GET\_CERTS\_RESULT\_OK  }  Verify that the EUM’s Certificate #CERT\_EUM\_SIG contains an EUM’s Public Key #PK\_EUM\_SIG based on brainpoolP256r1 curve.  Verify that the eUICC’s Certificate #CERT\_EUICC\_SIG contains an eUICC’s Public Key #PK\_EUICC\_SIG based on brainpoolP256r1 curve. |

##### 4.2.26.2.2 TC\_eUICC\_ES10b.GetCertificates\_ErrorCases

Test Sequence #01 Error: Invalid CI PKid

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_CERTS\_REQUEST (  #CI\_PKI\_RANDOM  )  ) | SW=0x9000 with response data:  GetCertsResponse ::= {  #GET\_CERTS\_ERR\_UNKNOWN\_CI  } |

### 4.2.27 ES10b (IPA -- eUICC): EnableUsingDD

This function is used by IPA to request automatic Profile enabling without eIM involvement. The eUICC MAY be configured to support the automatic enabling.

#### 4.2.27.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

* Section 5.9.15

#### 4.2.27.2 Test Cases

##### 4.2.27.2.1 TC\_eUICC\_ES10b.EnableUsingDD

Test Sequence #01 Nominal Case

Test the nominal case when the Automatic Profile Enabling is activated

|  |  |
| --- | --- |
| Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | Automatic Profile Enabling is activated |
| eUICC | The configured OID is set to S\_SM\_DP+\_OID1 and the SMDP\_ADDRESS is set to SMDP\_ADDRESS1 |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | Install PROFILE\_OPERATIONAL1 with OID value set to S\_SM\_DP+\_OID1 and address value set to SMDP\_ADDRESS1 | Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult) |
| 2 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ENABLE\_USING\_DD()  ) | response EnableUsingDDResponse ::= {  #ENABLE\_USING\_DD\_RESULT\_OK  }  SW=0x91XX |
| 3 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 4 | Repeat IC1 and IC2 | | |
| 5 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 6 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 7 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

##### 4.2.27.2.2 TC\_eUICC\_ES10b.EnableUsingDD\_ErrorCases

Test Sequence #01 Error: No Session Context

|  |  |
| --- | --- |
| Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | Automatic Profile Enabling is activated |
| eUICC | The configured OID is set to S\_SM\_DP+\_OID1 and the SMDP\_ADDRESS is set to SMDP\_ADDRESS1 |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | Install PROFILE\_OPERATIONAL1 with OID value set to S\_SM\_DP+\_OID1 and address value set to SMDP\_ADDRESS1 | Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult) |
| 2 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_GET\_RAT ()  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  getRATResult : {  #PPRS\_ALLOWED  }  }  },  euiccSignEPR EUICC\_SIGN\_EPR  }  SW = 0x9000 |
| 3 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ENABLE\_USING\_DD()  ) | response EnableUsingDDResponse ::= {  #ENABLE\_USING\_DD\_RESULT\_NO\_SESSION\_CONTEXT  }  SW=0x9000 |

Test Sequence #02 Error: Auto-Enable is Not Available

Test the nominal case when the Automatic Profile Enabling is not activated

|  |  |
| --- | --- |
| Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | Automatic Profile Enabling is not activated |
| eUICC | The configured OID is set to S\_SM\_DP+\_OID1 and the SMDP\_ADDRESS is set to SMDP\_ADDRESS1 |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | Install PROFILE\_OPERATIONAL1 with OID value set to S\_SM\_DP+\_OID1 and address value set to SMDP\_ADDRESS1 | Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult) |
| 2 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ENABLE\_USING\_DD()  ) | response EnableUsingDDResponse ::= {  #ENABLE\_USING\_DD\_RESULT\_AUTO\_ENABLE\_NOT\_AVAILABLE  }  SW=0x9000 |

### 4.2.28 ES10b (IPA -- eUICC): ProfileRollback

This function is used by the IPA to request to roll back to the previously Enabled Profile, if any.

#### 4.2.28.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

* Section 2.11.1.1.3
* Section 2.11.2.1
* Section 3.3.1
* Section 3.4.1
* Section 5.9.16

#### 4.2.28.2 Test Cases

##### 4.2.28.2.1 TC\_eUICC\_ES10b.ProfileRollback

|  |  |
| --- | --- |
| **General Initial Conditions** | |
| **Entity** | **Description of the general initial condition** |
| eUICC | The PROFILE\_OPERATIONAL1 has been installed on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL2 has been installed on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 is enabled with rollbackFlag set |

Test Sequence #1 Nominal Case: Profile Rollback with refreshFlag set

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF2,  TRUE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 5 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_PROFILE\_ROLLBACK\_REQUEST(  TRUE)  ) | response ProfileRollbackResponse::= { #PROFILE\_ROLLBACK\_RES\_OK  }  SW=0x9000 |
| 6 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 7 | Repeat IC1 and IC2 | | |
| 8 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 9 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 10 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

Test Sequence #2 Nominal Case: Profile Rollback with refreshFlag not set

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF2,  TRUE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 5 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_PROFILE\_ROLLBACK\_REQUEST(  FALSE)  ) | response ProfileRollbackResponse::= { #PROFILE\_ROLLBACK\_RES\_OK  }  SW=0x9000 |
| 6 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 7 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 8 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

Test Sequence #3 Nominal Case: Profile Rollback with refreshFLag set while proactive session is ongoing – catBusy not supported

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF2,  TRUE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 5 | S\_Device → eUICC | MTD\_SEND\_SMS\_PP(  [GET\_MNO\_SD]) | SW=0x91XX |
| 6 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_PROFILE\_ROLLBACK\_REQUEST(  TRUE)  ) | response ProfileRollbackResponse::= { #PROFILE\_ROLLBACK\_RES\_OK  }  SW=0x91YY |
| 7 | S\_Device → eUICC | FETCH 'YY' | REFRESH Command (“UICC Reset”) |
| 8 | Repeat IC1 and IC2 | | |
| 9 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 10 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 11 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

Test Sequence #4 Nominal Case: Profile Rollback with refreshFLag not set while proactive session is ongoing – catBusy not supported

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF2,  TRUE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 5 | S\_Device → eUICC | MTD\_SEND\_SMS\_PP(  [GET\_MNO\_SD]) | SW=0x91XX |
| 6 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_PROFILE\_ROLLBACK\_REQUEST(  FALSE)  ) | response ProfileRollbackResponse::= { #PROFILE\_ROLLBACK\_RES\_OK  }  SW=0x9000 |
| 7 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 8 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 9 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

Test Sequence #5 Nominal Case: Profile Rollback with refreshFLag set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF2,  TRUE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 5 | S\_Device → eUICC | MTD\_SEND\_SMS\_PP(  [GET\_MNO\_SD]) | SW=0x91XX |
| 6 | S\_Device → eUICC | FETCH 'XX' | SMS POR received  SCP80 response status code equal to 0x00 – POR OK |
| 7 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_PROFILE\_ROLLBACK\_REQUEST(  TRUE)  ) | response ProfileRollbackResponse::= { #PROFILE\_ROLLBACK\_RES\_OK  }  SW=0x9000 |
| 8 | S\_Device → eUICC | TERMINAL RESPONSE | SW=0x91YY |
| 9 | S\_Device → eUICC | FETCH 'YY' | REFRESH Command (“UICC Reset”) |
| 10 | Repeat IC1 and IC2 | | |
| 11 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 12 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 13 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

Test Sequence #6 Nominal Case: Profile Rollback with refreshFLag not set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF2,  TRUE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 5 | S\_Device → eUICC | MTD\_SEND\_SMS\_PP(  [GET\_MNO\_SD]) | SW=0x91XX |
| 6 | S\_Device → eUICC | FETCH 'XX' | SMS POR received  SCP80 response status code equal to 0x00 – POR OK |
| 7 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_PROFILE\_ROLLBACK\_REQUEST(  FALSE)  ) | response ProfileRollbackResponse::= { #PROFILE\_ROLLBACK\_RES\_OK  }  SW=0x9000 |
| 8 | S\_Device → eUICC | TERMINAL RESPONSE | SW= any value except91XX |
| 9 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 10 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 11 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

##### 4.2.28.2.2 TC\_eUICC\_ES10b.ProfileRollback\_ErrorCases

Test Sequence #01 Error: Roll Back is not allowed

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF2,  FALSE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_PROFILE\_ROLLBACK\_REQUEST(  FALSE)  ) | response ProfileRollbackResponse::= { #PROFILE\_ROLLBACK\_RES\_ROLLBACK\_NOT\_ALLOWED  }  SW=0x9000 |
| 5 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 6 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 7 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF2  SW=0x9000 |

Test Sequence #02 Error: Command Error

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_PROFILE\_ROLLBACK\_REQUEST(  FALSE)  ) | response ProfileRollbackResponse::= { #PROFILE\_ROLLBACK\_RES\_COMMAND\_ERROR  }  SW=0x9000 |
| 2 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1\_ENABLED  }  SW=0x9000 |
| 3 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 4 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

Test Sequence #3 Error: catBusy with refreshFlag set while proactive session is ongoing – catBusy supported

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF2,  TRUE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 5 | S\_Device → eUICC | MTD\_SEND\_SMS\_PP(  [GET\_MNO\_SD]) | SW=0x91XX |
| 6 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_PROFILE\_ROLLBACK\_REQUEST(  TRUE)  ) | response ProfileRollbackResponse::= { #PROFILE\_ROLLBACK\_RES\_CATBUSY  }  SW=0x91XX |
| 7 | S\_Device → eUICC | FETCH 'XX' | SMS POR received  SCP80 response status code equal to 0x00 – POR OK |
| 8 | S\_Device → eUICC | TERMINAL RESPONSE | SW=0x9000 |
| 9 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 10 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 11 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF2  SW=0x9000 |

Test Sequence #4 Error: catBusy with refreshFlag not set while proactive session is ongoing – catBusy supported

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF2,  TRUE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 5 | S\_Device → eUICC | MTD\_SEND\_SMS\_PP(  [GET\_MNO\_SD]) | SW=0x91XX |
| 6 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_PROFILE\_ROLLBACK\_REQUEST(  FALSE)  ) | response ProfileRollbackResponse::= { #PROFILE\_ROLLBACK\_RES\_CATBUSY  }  SW=0x91XX |
| 7 | S\_Device → eUICC | FETCH 'XX' | SMS POR received  SCP80 response status code equal to 0x00 – POR OK |
| 8 | S\_Device → eUICC | TERMINAL RESPONSE | SW=0x9000 |
| 9 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID2>)  ) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |
| 10 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 11 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF2  SW=0x9000 |

### 4.2.29 ES10b (IPA -- eUICC): ConfigureAutomaticProfileEnabling

This function is used by the IPA to activate or deactivate automatic Profile enabling in the eUICC and to add or update automatic enabling data used in the automatic Profile enabling.

#### 4.2.29.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

* Section 5.9.17

#### 4.2.29.2 Test Cases

##### 4.2.29.2.1 TC\_eUICC\_ES10b.ConfigureAutomaticProfileEnabling

Test Sequence #01 Nominal Case- active automatic profile enabling

Test the nominal case when: autoEnableFlag, smdpOid and smdpAddress are present in the command.

|  |  |
| --- | --- |
| **Initial Conditions** | |
| **Entity** | **Description of the general initial condition** |
| eUICC | Automatic Profile Enabling is not activated |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_CONFIGURE\_AUTOMATIC\_PROFILE\_ENABLING (  AUTO\_ENABLE\_FLAG,  S\_SM\_DP+\_OID1,  SMDP\_ADDRESS1,  )  ) | #CONFIG\_AUTO\_ENABLE\_RES\_OK  SW=0x9000 |
| 2 | S\_IPAd → eUICC | Install PROFILE\_OPERATIONAL1 with OID value set to S\_SM\_DP+\_OID1 and address value set to SMDP\_ADDRESS1 | Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult) |
| 3 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ENABLE\_USING\_DD()  ) | response EnableUsingDDResponse ::= {  #ENABLE\_USING\_DD\_RESULT\_OK  }  SW=0x91XX |
| 4 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 5 | Repeat IC1 and IC2 | | |
| 6 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 7 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 8 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

Test Sequence #02 Nominal Case - deactivate automatic profile enabling

Test the nominal case when autoEnableFlag is not present, smdpOid and smdpAddress are present in the command.

|  |  |
| --- | --- |
| **Initial Conditions** | |
| **Entity** | **Description of the general initial condition** |
| eUICC | Automatic Profile Enabling is activated |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_CONFIGURE\_AUTOMATIC\_PROFILE\_ENABLING (  NO\_PARAM,  S\_SM\_DP+\_OID1,  SMDP\_ADDRESS1  )  ) | #CONFIG\_AUTO\_ENABLE\_RES\_OK  SW=0x9000 |
| 2 | S\_IPAd → eUICC | Install PROFILE\_OPERATIONAL1 with OID value set to S\_SM\_DP+\_OID1 and address value set to SMDP\_ADDRESS1 | Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult) |
| 3 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ENABLE\_USING\_DD()  ) | response EnableUsingDDResponse ::= {  #ENABLE\_USING\_DD\_RESULT\_AUTO\_ENABLE\_NOT\_AVAILABLE  }  SW=0x9000 |

Test Sequence #03 Nominal Case - fail the check of SD-DP+ parameter

|  |  |
| --- | --- |
| **Initial Conditions** | |
| **Entity** | **Description of the general initial condition** |
| eUICC | Automatic Profile Enabling is not activated |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_CONFIGURE\_AUTOMATIC\_PROFILE\_ENABLING (  AUTO\_ENABLE\_FLAG,  S\_SM\_DP+\_OID1,  SMDP\_ADDRESS1  )  ) | #CONFIG\_AUTO\_ENABLE\_RES\_OK  SW=0x9000 |
| 2 | S\_IPAd → eUICC | Install PROFILE\_OPERATIONAL1 with OID value set to S\_SM\_DP+\_OID2 and address value set to SMDP\_ADDRESS1 | Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult) |
| 3 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_ENABLE\_USING\_DD()  ) | response EnableUsingDDResponse ::= {  #ENABLE\_USING\_DD\_RESULT\_AUTO\_ENABLE\_NOT\_AVAILABLE  }  SW=0x9000 |

##### 4.2.29.2.2 TC\_eUICC\_ES10b.ConfigureAutomaticProfileEnabling\_ErrorCase

Test Sequence #01 Error: Unsigned Auto Enable Configuration Disallowed

|  |  |
| --- | --- |
| **Initial Conditions** | |
| **Entity** | **Description of the general initial condition** |
| eUICC | No Eim configuration present on card. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_CONFIGURE\_AUTOMATIC\_PROFILE\_ENABLING (  AUTO\_ENABLE\_FLAG,  S\_SM\_DP+\_OID1,  SMDP\_ADDRESS1  )  ) | #CONFIG\_AUTO\_ENABLE\_RES\_CONFIG\_DISALLOWED  SW=0x9000 |

Test Sequence #02 Error: Insufficient Memory

This Test Sequence of this error is not defned in this version of the specification.

### 4.2.30 ES10b (IPA -- eUICC): GetEimConfigurationData

This function is used by the IPA to read eIM Configuration Data stored in the eUICC.

#### 4.2.30.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.30.2 Test Cases

##### 4.2.30.2.1 TC\_eUICC\_ES10b.GetEimConfigurationData

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | The eUICC does not contain any eIM Configuration Data. |

Test Sequence #01 Nominal Case: empty list of EIM

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_EIM(  )  ) | response GetEimConfigurationDataResponse ::= {  #EIM\_CONFIG\_DATA\_LIST\_EMPTY}  SW=0x9000 |

Test Sequence #02 Nominal Case: single EIM

|  |  |
| --- | --- |
| Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_EIM(  )  ) | response GetEimConfigurationDataResponse ::= {  #EIM\_CONFIG\_DATA\_LIST1}  SW=0x9000 |

Test Sequence #03 Nominal Case: multiple EIMs

|  |  |
| --- | --- |
| Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | #EIM\_CONFIG\_DATA1 and #EIM\_CONFIG\_DATA2 are configured in the eUICC. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_EIM(  )  ) | response GetEimConfigurationDataResponse ::= {  #EIM\_CONFIG\_DATA\_LIST2}  SW=0x9000 |

### 4.2.31 ESep (eIM -- eUICC): eUICC Package with single PSMO command: Enable

This function can be tested as Enable Procedures

#### 4.2.31.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

* Section 2.11.1.1
* Section 2.11.1.1.3
* Section 2.11.2.1
* Section 3.3.1
* Section 5.13.1

#### 4.2.31.2 Test Cases

##### 4.2.31.2.1 TC\_eUICC\_ESep.Enable

|  |  |
| --- | --- |
| **General Initial Conditions** | |
| **Entity** | **Description of the general initial condition** |
| eUICC | The PROFILE\_OPERATIONAL1 has been installed on the eUICC. |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

Test Sequence #01 Nominal Case: Enable Profile No transaction Id.

The purpose of this test is to verify that a Profile can be Enabled when the transaction Id is not present in the request.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Disabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 5 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 6 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

***Test Sequence #02 Nominal Case: Enable Profile with transaction Id.***

The purpose of this test is to verify that a Profile can be Enabled when the transaction Id is present in the request.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Disabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_TRANSACTION\_ID,  ICCID\_OP\_PROF1,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_2  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 5 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 6 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

***Test Sequence #03 Nominal Case: Enable Profile with transaction and the rollbackFlag.***

The purpose of this test is to verify that a Profile can be Enabled when the transaction Id and the rollbackFlag are present in the request.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Disabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_TRANSACTION\_ID,  ICCID\_OP\_PROF1,  TRUE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_2  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 5 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 6 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

***Test Sequence #04 Nominal Case: Enable Profile Device supports “eUICC Profile State Change”.***

The purpose of this test is to verify that a Profile can be Enabled when the transaction Id and the rollbackFlag are present in the request.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Disabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE\_eUICCProfileStateChanged | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“eUICC Profile State change”) |
| 3 | Execute IC1 from step 2 to step 4 | | |
| 4 | Repeat IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |
| 5 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 6 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |

##### 4.2.31.2.2 TC\_eUICC\_ESep.Enable\_ErrorCases

Test Sequence #01 Error: Command already processed

The purpose of this test is to verify that a Profile cannot be enabled because the command has been already processed.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Disabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x9000 |
| 5 | S\_Device → eUICC | [READ\_BINARY] with <L>=0x0A | #ICCID\_OP\_PROF1  SW=0x9000 |
| 6 | S\_eIM → eUICC | Resend the same command as in step 1, so the counter is also the same. | response EuiccPackageResult ::= euiccPackageErrorDataSigned: {  #PACKAGE\_RES\_ERROR\_CODE\_REPLAY\_ERROR  }  SW=0x9000 |

Test Sequence #02 Error: Target profile is not found

The purpose of this test is to verify that a Profile cannot be enabled because the target profile is not found.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Enabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |
| eUICC | The Operational Profile identified by the ICCID #ICCID\_OP\_PROFX is not loaded. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROFX,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ENABLE\_RES\_PROFILE\_ICCID\_NOT\_FOUND  }  SW=0x9000 |
| 2 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  # PROFILE\_INFO1\_DISABLED  }  SW=0x9000 |

Test Sequence #03 Error: Target profile is not in a Disable State

The purpose of this test is to verify that a Profile cannot be enabled because the target profile is not in a Disable State.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Enabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  # ENABLE\_RES\_PROFILE\_NOT\_IN\_DISABLED\_STATE  }  SW=0x9000 |
| 2 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  # PROFILE\_INFO1  }  SW=0x9000 |

Test Sequence #04 Error: Usage of the Rollback Mechanism is granted

The purpose of this test is to verify that a Profile cannot be enabled because the usage of the Rollback Mechanism is granted and there is no currently Enabled Profile.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Disable on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |
| eUICC | There is no currently Enabled Profile. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  # ENABLE\_RES\_PROFILE\_ UNDEFINED\_ERROR  }  SW=0x9000 |

### 4.2.32 ESep (eIM -- eUICC): eUICC Package with single PSMO command: Disable

#### 4.2.32.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

* Section 2.11.1.1
* Section 2.11.1.1.3
* Section 2.11.2.1
* Section 5.13.2
* Section 3.3.1

#### 4.2.32.2 Test Cases

##### 4.2.32.2.1 TC\_eUICC\_ESep.Disable

|  |  |
| --- | --- |
| **General Initial Conditions** | |
| **Entity** | **Description of the general initial condition** |
| eUICC | The PROFILE\_OPERATIONAL1 has been installed on the eUICC. |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

Test Sequence #01 Nominal Case: Disable Profile No transaction Id

The purpose of this test is to verify that a Profile can be disabled when the transaction Id is not present in the request.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Enabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DISABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DISABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1\_DISABLED  }  SW=0x9000 |
| 5 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x6A82 |

Test Sequence #02 Nominal Case: Disable Profile when the transaction Id is present

The purpose of this test is to verify that a Profile can be disabled when the transaction Id is present.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Enabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DISABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  S\_TRANSACTION\_ID,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DISABLE\_RES\_OK\_2  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1\_DISABLED  }  SW=0x9000 |
| 5 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x6A82 |

Test Sequence #03 Nominal Case: Disable Profile when Device supports “eUICC Profile State Change”

The purpose of this test is to verify that a Profile can be disabled with REFRESH type “eUICC Profile State Change”

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Enabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE\_eUICCProfileStateChanged | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DISABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  S\_TRANSACTION\_ID,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DISABLE\_RES\_OK\_2  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“eUICC Profile State changed”) |
| 3 | Execute IC1 from step 2 to step 4 | | |
| 4 | Repeat IC2 | | |
| 5 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1\_DISABLED  }  SW=0x9000 |
| 6 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x6A82 |

##### 4.2.32.2.2 TC\_eUICC\_ESep.Disable\_ErrorCases

Test Sequence #01 Error: Command already processed

The purpose of this test is to verify that a Profile cannot be disabled because the command has been already processed.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Enabled on the eUICC. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DISABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DISABLE\_RES\_OK\_1  }  SW=0x91XX |
| 2 | S\_Device → eUICC | FETCH 'XX' | REFRESH Command (“UICC Reset”) |
| 3 | Repeat IC1 and IC2 | | |
| 4 | S\_Device → eUICC | [SELECT\_ICCID] | SW=0x6A82 |
| 5 | S\_eIM → eUICC | Resend the same command as in step 1, so the counter is also the same. | response EuiccPackageResult ::= euiccPackageErrorDataSigned: {  #PACKAGE\_RES\_ERROR\_CODE\_REPLAY\_ERROR  }  SW=0x9000 |

Test Sequence #02 Error: Target profile is not found

The purpose of this test is to verify that a Profile cannot be disabled because the target profile is not found.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Enabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |
| eUICC | The Operational Profile identified by the ICCID #ICCID\_OP\_PROFX is not loaded. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DISABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROFX,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DISABLE\_RES\_PROFILE\_ICCID\_NOT\_FOUND  }  SW=0x9000 |
| 2 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1  }  SW=0x9000 |

Test Sequence #03 Error: Target profile is not in Enabled State

The purpose of this test is to verify that a Profile cannot be disabled because the target profile is not in Enabled State.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Disabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DISABLE(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DISABLE\_RES\_PROFILE\_NOT\_IN\_ENABLED\_STATE  }  SW=0x9000 |
| 2 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk : {  #PROFILE\_INFO1\_DISABLED  }  SW=0x9000 |

### 4.2.33 ESep (eIM -- eUICC): eUICC Package with single PSMO command: Delete

#### 4.2.33.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

* Section 2.11.1.1
* Section 2.11.1.1.3
* Section 2.11.2.1
* Section 3.3.1
* Section 5.13.3

#### 4.2.33.2 Test Cases

##### 4.2.33.2.1 TC\_eUICC\_ESep.Delete

|  |  |
| --- | --- |
| **General Initial Conditions** | |
| **Entity** | **Description of the general initial condition** |
| eUICC | The PROFILE\_OPERATIONAL1 has been installed on the eUICC. |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

Test Sequence #01 Nominal Case: Delete Profile No transaction Id.

The purpose of this test is to verify that a Profile can be deleted when the transaction Id is not present in the request.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Disabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF1,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DELETE\_RES\_OK\_1  }  SW=0x9000 |
| 2 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk: {  }  SW=0x9000 |

Test Sequence #02 Nominal Case: Delete Profile Transaction Id.

The purpose of this test is to verify that a Profile can be deleted when the transaction Id is present in the request.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 is Disabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 corresponds to <ISD\_P\_AID1>. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_TRANSACTION\_ID,  ICCID\_OP\_PROF1,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DELETE\_RES\_OK\_2  }  SW=0x9000 |
| 2 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_GET\_PROFILE\_INFO(  NO\_PARAM,  <ISD\_P\_AID1>)) | response ProfileInfoListResponse::= profileInfoListOk: {  }  SW=0x9000 |

##### 4.2.33.2.2 TC\_eUICC\_ESep.Delete\_ErrorCases

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 has been installed on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL2 has been installed on the eUICC. |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

Test Sequence #01 Error: Target profile is not found

The purpose of this test is to verify that a Profile cannot be deleted because the target profile is not found.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The Operational Profile identified by the ICCID #ICCID\_OP\_PROFX is not loaded. |
| eUICC | The PROFILE\_OPERATIONAL1 is Disabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL2 is Enabled on the eUICC. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROFX,  S\_EIM\_SIGNATURE  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DELETE\_PROFILE\_ICCID\_NOT**\_**FOUND  }    SW=0x9000 |
| 2 | S\_IPAd 🡪 eUICC | MTD\_STORE\_DATA( #GET\_PROFILES\_INFO\_ALL) | response ProfileInfoListResponse::= profileInfoListOk: {  #PROFILE\_INFO1\_DISABLED,  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |

Test Sequence #02 Error: Profile not in disable state

The purpose of this test is to verify that a Profile cannot be deleted because the target profile is not in disable state.

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The Operational Profile identified by the ICCID #ICCID\_OP\_PROFX is not loaded. |
| eUICC | The PROFILE\_OPERATIONAL1 is Disabled on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL2 is Enabled on the eUICC. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_eIM → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ICCID\_OP\_PROF2,  S\_EIM\_SIGNATURE  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DELETE\_PROFILE\_ICCID\_NOT**\_**IN\_DISABLED\_STATE  }    SW=0x9000 |
| 2 | S\_IPAd 🡪 eUICC | MTD\_STORE\_DATA( #GET\_PROFILES\_INFO\_ALL) | response ProfileInfoListResponse::= profileInfoListOk: {  #PROFILE\_INFO1\_DISABLED,  #PROFILE\_INFO2\_ENABLED  }  SW=0x9000 |

### 4.2.34 ESep (eIM -- eUICC): eUICC Package with single PSMO command: ListProfileInfo

This function allows the eIM to retrieve the list of Profile information for installed Profiles including their current state (Enabled/Disabled) and their associated Profile Metadata..

#### 4.2.34.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

* Section 2.11.1.1
* Section 2.11.1.1.3
* Section 2.11.2.1
* Section 3.3.1
* Section 5.13.4

#### 4.2.34.2 Test Cases

##### 4.2.34.2.1 TC\_eUICC\_ESep.ListProfileInfo

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 has been installed on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL2 has been installed on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL3 has been installed on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 is Enabled. |
| eUICC | The Nickname of PROFILE\_OPERATIONAL1 and PROFILE\_OPERATIONAL2 is empty. |
| eUICC | The Nickname of the PROFILE\_OPERATIONAL3 is equal to #NICKNAME3. |

Test Sequence #01 Nominal Case: Retrieve all Profiles with no searchCriteria

The purpose of this test is to verify the ListProfileInfo function when the searchCriteria is not present.

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_LIST\_PRO\_FILE\_INFO (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  NO\_PARAM,  NO\_PARAM,  NO\_PARAM, S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  listProfileInfoResult : profileInfoListOk : {  #PROFILE\_INFO1,  #PROFILE\_INFO2,  #PROFILE\_INFO3  }  }  },  euiccSignEPR EUICC\_SIGN\_EPR  }  SW = 0x9000 |

Test Sequence #02 Nominal Case: Retrieve Profile by AID

The purpose of this test is to verify the ListProfileInfo function when the AID criteria is provided.

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_LIST\_PRO\_FILE\_INFO (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  ISD\_P\_AID1,  NO\_PARAM,  NO\_PARAM, S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  listProfileInfoResult : profileInfoListOk : {  #PROFILE\_INFO1  }  }  },  euiccSignEPR EUICC\_SIGN\_EPR  }  SW = 0x9000 |

Test Sequence #03 Nominal Case: Retrieve Profile by Iccid

The purpose of this test is to verify the ListProfileInfo function when the IccID criteria is provided

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_LIST\_PRO\_FILE\_INFO (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  ICCID\_OP\_PROF2,  NO\_PARAM,  NO\_PARAM,  NO\_PARAM, S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  listProfileInfoResult : profileInfoListOk : {  #PROFILE\_INFO2  }  }  },  euiccSignEPR EUICC\_SIGN\_EPR  }  SW = 0x9000 |

Test Sequence #04 Nominal Case: Retrieve Profiles by profileClass

The purpose of this test is to verify the ListProfileInfo function when the profileClass criteria is provided

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_LIST\_PRO\_FILE\_INFO (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  NO\_PARAM,  operational,  NO\_PARAM, S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  listProfileInfoResult : profileInfoListOk : {  #PROFILE\_INFO1,  #PROFILE\_INFO2,  #PROFILE\_INFO3  }  }  },  euiccSignEPR EUICC\_SIGN\_EPR  }  SW = 0x9000 |

Test Sequence #05 Nominal Case: Retrieve Profiles with tagList

The purpose of this test is to verify the ListProfileInfo function when the tagList criteria is provided

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_LIST\_PRO\_FILE\_INFO (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  NO\_PARAM,  NO\_PARAM,  TAGS\_AID\_PROFILENAME, S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  #PROFILE\_INFO\_TAG\_LIST  },  euiccSignEPR EUICC\_SIGN\_EPR  }  SW = 0x9000 |

##### 4.2.34.2.2 TC\_eUICC\_ESep.ListProfileInfo\_ErrorCases

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | The PROFILE\_OPERATIONAL1 has been installed on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL2 has been installed on the eUICC. |
| eUICC | The PROFILE\_OPERATIONAL1 is Enabled. |

Test Sequence #01 Error: Incorrect Input Values

The purpose of this test is to verify the ListProfileInfo function when the tagList value is invalid.

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_LIST\_PRO\_FILE\_INFO (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  NO\_PARAM,  NO\_PARAM,  PROFILE\_TAG\_INVALID, S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  listProfileInfoResult : profileInfoListError : 1  }  } , euiccSignEPR EUICC\_SIGN\_EPR  }  SW = 0x9000 |

Test Sequence #02 Error: Proflie Change Ongoing

The purpose of this test is to verify that a listProfileInfo function SHALL NOT occur after an enable command in an eUICC Package.

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD LIST\_PROFILE\_INFO CHANGE\_ONGOING\_ERROR (  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  enableResult : 0,  listProfileInfoResult : profileInfoListError : 11  }  },  euiccSignEPR EUICC\_SIGN\_EPR  }  SW = 0x9000 |

### 4.2.35 ESep (eIM -- eUICC): eUICC Package with single PSMO command: GetRat

This function allows the eIM to retrieve the Rules Authorisation Table (RAT) from the eUICC.

#### 4.2.35.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

* Section 2.11.1.1
* Section 2.11.1.1.3
* Section 2.11.2.1
* Section 3.3.1
* Section 5.13.5

#### 4.2.35.2 Test Cases

4.2.35.2.1 TC\_eUICC\_ESep.GetRat

Test Sequence #01 Nominal Case

|  |  |
| --- | --- |
| Initial Conditions |  |
| Entity | Description of the initial condition |
| eUICC | The EUM has configured the eUICC's RAT as defined in section G.1.4. |

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_IPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_GET\_RAT ( )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  getRATResult : {  #PPRS\_ALLOWED  }  }  },  euiccSignEPR EUICC\_SIGN\_EPR  }  SW = 0x9000 |

### 4.2.36 ESep (eIM -- eUICC): eUICC Package with single eCO command: AddEim

This function adds an Associated eIM to the eUICC by providing its eIM Configuration Data including the eimID to the eUICC.

#### 4.2.36.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.36.2 Test Cases

4.2.36.2.1 TC\_eUICC\_ESep.AddEim

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

Test Sequence #01 Nominal Case: AddEim no eimTransactionId

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ADD\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA2,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ADD\_EIM\_RES\_OK\_1  }  SW=0x9000 |

Test Sequence #02 Nominal Case: AddEim with eimTransactionId

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ADD\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_EIM\_TRANSACTION\_ID,  EIM\_CONFIG\_DATA2,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ADD\_EIM\_RES\_OK\_2  }  SW=0x900 |

4.2.36.2.2 TC\_eUICC\_ESep.AddEim\_ErrorCases

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

Test Sequence #01 Error: Insufficient Memory

The test sequence of this error is not defined for this version of the specification.

Test Sequence #02 Error: CI PK Unknown

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ADD\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA2\_ADD\_UNKNOWN\_EUICC\_CI\_PKID,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ADD\_EIM\_RES\_ERR\_CI\_PK\_UNKNOWN  }  SW=0x9000 |

Test Sequence #03 Error: invalid Association Token

The test sequence of this error is not defined for this version of the specification.

Test Sequence #04 Error: counter Value Out Of Range

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ADD\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA2\_ADD\_COUNTER\_OUT\_OF\_RANGE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ADD\_EIM\_RES\_ERR\_COUNTER\_OUT\_OF\_RANGE  }  SW=0x9000 |

Test Sequence #05 Error: no eimId is provided

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ADD\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA\_NO\_EIM\_ID,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ADD\_EIM\_RES\_ERR\_COMMAND\_ERROR  }  SW=0x9000 |

Test Sequence #06 Error: eimId exceeds the limit

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ADD\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA2\_EIM\_ID\_129\_CHARACTERS,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ADD\_EIM\_RES\_ERR\_COMMAND\_ERROR  }  SW=0x9000 |

Test Sequence #07 Error: eimId already presents

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_ADD\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA1,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #ADD\_EIM\_RES\_ERR\_COMMAND\_ERROR  }  SW=0x9000 |

### 4.2.37 ESep (eIM -- eUICC): eUICC Package with single eCO command: UpdateEim

This function updates eIM Configuration Data, i.e., the public key or Certificate and the related anti-replay counter value of an Associated eIM with a given eimID within the eUICC while keeping the same eimID.

#### 4.2.37.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.37.2 Test Cases

4.2.37.2.1 TC\_eUICC\_ESep.UpdateEIM

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

Test Sequence #01 Nominal Case: UpdateEim no eimTransactionId

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA1\_UPDATE\_COUNTER,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #UPDATE\_EIM\_RES\_OK\_1  }  SW=0x9000 |

Test Sequence #02 Nominal Case: UpdateEim with eimTransactionId

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_EIM\_TRANSACTION\_ID,  EIM\_CONFIG\_DATA1\_UPDATE\_COUNTER,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #UPDATE\_EIM\_RES\_OK\_2  }  SW=0x9000 |

4.2.37.2.2 TC\_eUICC\_ESep.UpdateEim\_ErrorCases

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

Test Sequence #01 Error: EIM not Found

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA2\_UPDATE\_COUNTER,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #UPDATE\_EIM\_RES\_ERR\_EIM\_NOT\_FOUND  }  SW=0x9000 |

Test Sequence #02 Error: CI PK Unknown

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA1\_UPDATE\_UNKNOWN\_EUICC\_CI\_PKID,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #UPDATE\_EIM\_RES\_ERR\_UNKNOWN\_EUICC\_CI\_PKID  }  SW=0x9000 |

Test Sequence #03 Error: Counter Value Out Of Range

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA1\_UPDATE\_COUNTER\_OUT\_OF\_RANGE,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #UPDATE\_EIM\_RES\_ERR\_COUNTER\_OUT\_OF\_RANGE  }  SW=0x9000 |

Test Sequence #04 Error: Lowering Counter Value

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  EIM\_CONFIG\_DATA1\_UPDATE\_LOWER\_COUNTER,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #UPDATE\_EIM\_RES\_ERR\_COMMAND\_ERROR  }  SW=0x9000 |

### 4.2.38 ESep (eIM -- eUICC): eUICC Package with single eCO command: DeleteEim

This function deletes an Associated eIM identified by its eimID from the eUICC. If the successfully deleted Associated eIM was the last available Associated eIM, the eUICC SHALL allow ES10b.AddInitialEim again.

#### 4.2.38.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.38.2 Test Cases

4.2.38.2.1 TC\_eUICC\_ESep.DeleteEim

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC.  #EIM\_CONFIG\_DATA2 is configured in the eUICC. |

Test Sequence #01 Nominal Case: DeleteEim no eimTransactionId

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  TEST\_EIM\_ID2,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DELETE\_EIM\_RES\_OK\_1  }  SW=0x9000 |

Test Sequence #02 Nominal Case: DeleteEim with eimTransactionId

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_EIM\_TRANSACTION\_ID,  TEST\_EIM\_ID2,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DELETE\_EIM\_RES\_OK\_2  }  SW=0x9000 |

4.2.38.2.2 TC\_eUICC\_ESep.DeleteEim\_ErrorCases

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC. |

Test Sequence #01 Error: EIM Not Found

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_TRANSACTION\_ID,  TEST\_EIM\_ID2,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DELETE\_EIM\_RES\_ERR\_EIM\_NOT\_FOUND  }  SW=0x9000 |

Test Sequence #02 Error: Last EIM Deleted

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_TRANSACTION\_ID,  TEST\_EIM\_ID1,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DELETE\_EIM\_RES\_ERR\_LAST\_EIM\_DELETED  }  SW=0x9000 |

Test Sequence #03 Error: eimId is provided

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_TRANSACTION\_ID,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #DELETE\_EIM\_RES\_ERR\_COMMAND\_ERROR  }  SW=0x9000 |

### 4.2.39 ESep (eIM -- eUICC): eUICC Package with single eCO command: ListEim

This function requests the eUICC to provide a list of all currently configured Associated eIMs to the eIM.

#### 4.2.39.1 Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### 4.2.39.2 Test Cases

4.2.39.2.1 TC\_eUICC\_ESep.ListEim

|  |  |
| --- | --- |
| General Initial Conditions | |
| Entity | Description of the general initial condition |
| eUICC | S\_eIM with #EIM\_CONFIG\_DATA1 is configured in the eUICC.  #EIM\_CONFIG\_DATA2 is configured in the eUICC. |

Test Sequence #01 Nominal Case: ListEim no eimTransactionId

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_LIST\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  NO\_PARAM,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #LIST\_EIM\_RES\_OK\_1  }  SW=0x9000 |

Test Sequence #02 Nominal Case: ListEim with eimTransactionId

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Expected result |
| IC1 | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | | |
| IC2 | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | | |
| 1 | S\_LPAd → eUICC | MTD\_STORE\_DATA(  MTD\_EUICC\_PACKAGE\_REQUEST\_LIST\_EIM(  TEST\_EIM\_ID1,  EID1,  S\_COUNTER\_VALUE,  S\_EIM\_TRANSACTION\_ID,  NO\_PARAM,  S\_EIM\_SIGNATURE  )  ) | response EuiccPackageResult ::= euiccPackageResultSigned : {  #LIST\_EIM\_RES\_OK\_2  }  SW=0x9000 |

4.2.39.2.2 TC\_eUICC\_ESep.ListEim\_ErrorCases

***Test Sequence #01 Error: Command Error***

The test sequence of this error is not defined in this version of the specification.

## IPAe Interfaces

### ES9+ (IPAe -- SM-DP+): InitiateAuthentication

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

#### Conformance Requirements

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### Test Cases

##### TC\_LPAe\_InitiateAuthentication\_Nominal

***Test Sequence #01 Nominal: Initiate Authentication***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Initiate Authentication* defined in section 4.4.21.2.1 TC\_LPAe\_InitiateAuthentication\_Nominal where the LPAd plays the IPAe role

##### TC\_LPAd\_InitiateAuthentication\_ErrorCases

***Test Sequence #01 Error: Invalid SM-DP+ Address***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Invalid SM-DP+ Address* defined in section 4.4.21.2.2 TC\_LPAe\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #02 Error: Unsupported Security Configuration***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Unsupported Security Configuration* defined in section 4.4.21.2.2 TC\_LPAe\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #03 Error: Unsupported SVN***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Unsupported SVN* defined in section 4.4.21.2.2 TC\_LPAe\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #04 Error: Unavailable SM-DP+ Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Unavailable SM-DP+ Certificate* defined in section 4.4.21.2.2 TC\_LPAe\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #05 Error: Invalid SM-DP+ Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Error: Invalid SM-DP+ Certificate* defined in section 4.4.21.2.2 TC\_LPAe\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #06 Error: Invalid SM-DP+ Signature***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Error: Invalid SM-DP+ Signature* defined in section 4.4.21.2.2 TC\_LPAe\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #07 Error: Invalid SM-DP+ Address sent by the SM-DP+***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Invalid SM-DP+ Address sent by the SM-DP+* defined in section 4.4.21.2.2 TC\_LPAe\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #08 Error: Unsupported CI Key ID***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #08 Error: Unsupported CI Key ID* defined in section 4.4.21.2.2 TC\_LPAe\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #09 Error: Invalid SM-DP+ OID***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #09 Error: Invalid SM-DP+ OID* defined in section 4.4.21.2.2 TC\_LPAe\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

### ES9+ (IPAe -- SM-DP+): GetBoundProfilePackage

#### Conformance Requirements

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### Test Cases

##### TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Nominal

***Test Sequence #01 Nominal: Get BPP using S-ENC and S-MAC without Confirmation Code***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Get BPP using S-ENC and S-MAC without Confirmation Code* defined in section 4.4.22.2.1 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Nominal where the LPAd plays the IPAe role

***Test Sequence #02 Nominal: Get BPP using S-ENC and S-MAC with Confirmation Code***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: Get BPP using S-ENC and S-MAC with Confirmation Code* defined in section 4.4.22.2.1 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Nominal where the LPAd plays the IPAe role

***Test Sequence #03 Nominal: Get BPP using PPK-ENC and PPK-MAC without Confirmation Code***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Nominal: Get BPP using PPK-ENC and PPK-MAC without Confirmation Code* defined in section 4.4.22.2.1 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Nominal where the LPAd plays the IPAe role

***Test Sequence #04 Nominal: Get BPP using PPK-ENC and PPK-MAC with Confirmation Code***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Nominal: Get BPP using PPK-ENC and PPK-MAC with Confirmation Code* defined in section 4.4.22.2.1 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Nominal where the LPAd plays the IPAe role

##### TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Retry

***Test Sequence #01 Nominal: Get BPP Retry after incorrect Confirmation Code***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Get BPP Retry after incorrect Confirmation Code*defined in section 4.4.22.2.2 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Retry where the LPAd plays the IPAe role

##### TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Error

***Test Sequence #01 Error: Wrong eUICC Signature***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Wrong eUICC Signature* defined in section 4.4.22.2.3 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Error where the LPAd plays the IPAe role

***Test Sequence #02 Error: BPP Not Available***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: BPP Not Available* defined in section 4.4.22.2.3 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Error where the LPAd plays the IPAe role

***Test Sequence #03 Error: Unknown TransactionID received by SM-DP+***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Unknown TransactionID received by SM-DP+* defined in section 4.4.22.2.3 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Error where the LPAd plays the IPAe role

***Test Sequence #04 Error: Missing Confirmation Code***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Missing Confirmation Code received by SM-DP+* defined in section 4.4.22.2.3 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Error where the LPAd plays the IPAe role

***Test Sequence #05 Error: Download Order Expired***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Error: Download Order Expired* defined in section 4.4.22.2.3 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Error where the LPAd plays the IPAe role

***Test Sequence #06 Error: Wrong Confirmation Code***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Error: Wrong Confirmation Code* defined in section 4.4.22.2.3 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Error where the LPAd plays the IPAe role

***Test Sequence #07 Error: Maximum number of Confirmation Code retries exceeded***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Maximum number of Confirmation Code retries exceeded* defined in section 4.4.22.2.3 TC\_LPAd\_ES9+\_GetBoundProfilePackage\_Error where the LPAd plays the IPAe role

### ES9+ (IPAe -- SM-DP+): AuthenticateClient

#### Conformance Requirements

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### Test Cases

##### TC\_IPAe\_AuthenticateClient\_Nominal

***Test Sequence #01 Nominal: Authenticate Client without Confirmation Code***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Authenticate Client without Confirmation Code* defined in section 4.4.23.2.1 TC\_LPAd\_AuthenticateClient\_Nominal where the LPAd plays the IPAe role

***Test Sequence #02 Nominal: Authenticate Client with Confirmation Code***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: Authenticate Client with Confirmation Code* defined in section 4.4.23.2.1 TC\_LPAd\_AuthenticateClient\_Nominal where the LPAd plays the IPAe role

***Test Sequence #03 Nominal: Authenticate Client with Confirmation Code Retry***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Nominal: Authenticate Client with Confirmation Code Retry* defined in section 4.4.23.2.1 TC\_LPAd\_AuthenticateClient\_Nominal where the LPAd plays the IPAe role

##### TC\_IPAe\_AuthenticateClient\_ErrorCases

***Test Sequence #01 Error: Invalid EUM Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Invalid EUM Certificate* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #02 Error: Expired EUM Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Expired EUM Certificate* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #03 Error: Invalid eUICC Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Invalid eUICC Certificate* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #04 Error: Expired eUICC Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Expired eUICC Certificate* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #05 Error: Invalid eUICC Signature or serverChallenge***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Error: Invalid eUICC Signature or serverChallenge* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #06 Error: Insufficient Memory***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Error: Insufficient Memory* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #07 Error: Unknown CI Root Key***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Error: Insufficient Memory* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #08 Error: Profile not Allowed (Not in 'released' State)***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #08 Error: Profile not Allowed (Not in 'released' State)* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #09 Error: Unknown TransactionID***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #09 Error: Unknown TransactionID* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #10 Error: Refused MatchingID***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #10 Error: Refused MatchingID* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #11 Error: Refused EID***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #11 Error: Refused EID* in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #12 Error: No Eligible Profile for this eUICC/Device***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #12 Error: No Eligible Profile for this eUICC/Device* in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #13 Error: Expired Download Order***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #13 Error: Expired Download Order* in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #14 Error: Maximum Number of Retries Exceeded***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #14 Error: Maximum Number of Retries Exceeded* in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #15 Error: Invalid SM-DP+(pb) certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #15 Error: Invalid SM-DP+(pb) certificate* in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #16 Error: Different OID for SM-DP+ Certificates (CERT.DPpb.ECDSA and CERT.DPauth.ECDSA not belonging to the same entity)***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #16 Error: Different OID for SM-DP+ Certificates (CERT.DPpb.ECDSA and CERT.DPauth.ECDSA not belonging to the same entity)* in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #17 Error: Invalid SM-DP+ signature (smdpSignature2)***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #17 Error: Invalid SM-DP+ signature (smdpSignature2)* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #18 Error: Invalid TransactionID sent by SM-DP+***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #18 Error: Invalid TransactionID sent by SM-DP+* defined in section 4.4.23.2. TC\_LPAd\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

### ES9+ (IPAe – SM-DP+): HandleNotification

#### Conformance Requirements

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### Test Cases

##### TC\_IPAe\_ES9+\_HandleNotification\_Nominal

Throughout all the test cases the maximum number of Notifications simultaneously tested has been set as to two as there is not minimum defined in SGP.21 [3] or SGP.22 [2] for the number of Notifications that can be stored by the eUICC.

***Test Sequence #01 Nominal: Successful PIR and Install Notifications to the Same SM-DP+ Address***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Successful PIR and Install Notifications to the Same SM-DP+ Address* in section4.4.24.2.1 TC\_LPAd\_ES9+\_HandleNotification\_Nominal where the LPAd plays the IPAe role

***Test Sequence #02 Nominal: Successful PIR and Enable Notifications to the Same SM-DP+ Address***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: Successful PIR and Enable Notifications to the Same SM-DP+ Address* in section4.4.24.2.1 TC\_LPAd\_ES9+\_HandleNotification\_Nominal where the LPAd plays the IPAe role

***Test Sequence #03 Nominal: Disable and Delete Notifications to the Same SM-DP+ Address***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Nominal: Disable and Delete Notifications to the Same SM-DP+ Address* in section4.4.24.2.1 TC\_LPAd\_ES9+\_HandleNotification\_Nominal where the LPAd plays the IPAe role

***Test Sequence #04 Nominal: Enable and Disable Notifications with Different SM-DP+ Addresses***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Nominal: Enable and Disable Notifications with Different SM-DP+ Addresses* in section4.4.24.2.1 TC\_LPAd\_ES9+\_HandleNotification\_Nomina lwhere the LPAd plays the IPAe role

***Test Sequence #05 Nominal: Different SM-DP+ Addresses in PIR and Install Notifications***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Nominal: Different SM-DP+ Addresses in PIR and Install Notifications* in section 4.4.24.2.1 TC\_LPAd\_ES9+\_HandleNotification\_Nominal where the LPAd plays the IPAe role

***Test Sequence #06 Nominal: Profile Download with PIR Failed***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Nominal: Profile Download with PIR Failed* in section 4.4.24.2.1 TC\_LPAd\_ES9+\_HandleNotification\_Nominal where the LPAd plays the IPAe role

***Test Sequence #07 Nominal: Successful PIR and Install Notifications after Connectivity Interruption***

This Test Sequence is FFS as indicated in SGP.23 [8].

***Test Sequence #08 Nominal: No Acknowledge for Successful PIR results in No Further Notifications***

The purpose of this test case is to verify that the next Notification of a group is not sent until LPA receives a successful response from the SM-DP+ for the previous Notification.

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #08 Nominal: No Acknowledge for Successful PIR results in No Further Notifications* in section 4.4.24.2.1 TC\_LPAd\_ES9+\_HandleNotification\_Nominal where the LPAd plays the IPAe role

***Test Sequence #09 Nominal: Disable and Delete Notifications to the Same SM-DP+ Address using Delete Operation***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #09 Nominal: Disable and Delete Notifications to the Same SM-DP+ Address using Delete Operation* in section 4.4.24.2.1 TC\_LPAd\_ES9+\_HandleNotification\_Nominal where the LPAd plays the IPAe role

***Test Sequence #10 Nominal: Disable and Delete Notifications with Different SM-DP+ Addresses***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #10 Nominal: Disable and Delete Notifications with Different SM-DP+ Addresses* in section 4.4.24.2.1 TC\_LPAd\_ES9+\_HandleNotification\_Nominal where the LPAd plays the IPAe role

### ES9+ (IPAe – SM-DP+): CancelSession

#### Conformance Requirements

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### Test Cases

##### TC\_IPAe\_ES9+\_CancelSession\_Nominal

***Test Sequence #01 Nominal: Profile Download with PPR1 not allowed due to Operational Profile already present***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Profile Download with PPR1 not allowed due to Operational Profile already present* in section 4.4.25.2.1 TC\_LPAd\_ES9+\_CancelSession\_Nominal where the LPAd plays the IPAe role

***Test Sequence #02 Nominal: End User rejection***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: End User rejection* in section 4.4.25.2.1 TC\_LPAd\_ES9+\_CancelSession\_Nominal where the LPAd plays the IPAe role

***Test Sequence #03 Nominal: Load BPP Error***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Nominal: Load BPP Error* in section 4.4.25.2.1 TC\_LPAd\_ES9+\_CancelSession\_Nominal where the LPAd plays the IPAe role

***Test Sequence #04 Nominal: End User Timeout***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Nominal: End User Timeout* in section 4.4.25.2.1 TC\_LPAd\_ES9+\_CancelSession\_Nominal where the LPAd plays the IPAe role

***Test Sequence #05 Nominal: Load BPP Error due to unknown TAG***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Nominal: Load BPP Error due to unknown TAG* in section 4.4.25.2.1 TC\_LPAd\_ES9+\_CancelSession\_Nominal where the LPAd plays the IPAe role

##### TC\_IPAe\_ES9+\_CancelSession\_EndUserPostponed\_Nominal

***Test Sequence #01 Nominal: End User Postponed***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: End User Postponed* in section 4.4.25.2.2 TC\_LPAd\_ES9+\_CancelSession\_EndUserPostponed\_Nominal where the LPAd plays the IPAe role

##### TC\_IPAe\_ES9+\_CancelSession\_Error

***Test Sequence #01 Error: Unknown TransactionID after End User Rejection/Postpone***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Unknown TransactionID after End User Rejection/Postpone* in section 4.4.25.2.3 TC\_LPAd\_ES9+\_CancelSession\_Error where the LPAd plays the IPAe role

***Test Sequence #02 Error: Invalid eUICC Signature after End User Rejection/Postpone***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Invalid eUICC Signature after End User Rejection/Postpone* in section 4.4.25.2.3 TC\_LPAd\_ES9+\_CancelSession\_Error where the LPAd plays the IPAe role

***Test Sequence #03 Error: Invalid SM-DP+ OID after End User Rejection/Postpone***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Invalid SM-DP+ OID after End User Rejection/Postpone* in section 4.4.25.2.3 TC\_LPAd\_ES9+\_CancelSession\_Error where the LPAd plays the IPAe role

##### TC\_IPAe\_ES9+\_CancelSession\_PPRs

***Test Sequence #01 Nominal: End User rejection/postpone after PPR1 consent requested***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: End User rejection/postpone after PPR1 consent requested* in section 4.4.25.2.4 TC\_LPAd\_ES9+\_CancelSession\_PPRs where the LPAd plays the IPAe role

***Test Sequence #02 Nominal: End User rejection/posptone after PPR2 consent requested***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: End User rejection/posptone after PPR2 consent requested* in section 4.4.25.2.4 TC\_LPAd\_ES9+\_CancelSession\_PPRs where the LPAd plays the IPAe role

### ES9+ (IPAe – SM-DP+): HTTPS

#### Conformance Requirements

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### Test Cases

##### TC\_IPAe\_HTTPS\_Nominal

***Test Sequence #01 Nominal: HTTPS Session Establishment***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: HTTPS Session Establishment* in section 4.4.26.2.1 TC\_LPAd\_HTTPS\_Nominal where the LPAd plays the IPAe role

***Test Sequence #02 Nominal: non-reuse of session keys***

The purpose of this test sequence is to verify that the IPAe is not reusing ephemeral keys from the previous session.

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: non-reuse of session keys* in section 4.4.26.2.1 TC\_LPAd\_HTTPS\_Nominal where the LPAd plays the IPAe role

##### TC\_IPAe\_HTTPS\_ErrorCases

***Test Sequence #01 Error: Invalid (SM-DP+) TLS Certificate signature***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Invalid (SM-DP+) TLS Certificate signature* in section 4.4.26.2.2 TC\_LPAd\_HTTPS\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #02 Error: Expired TLS Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Expired TLS Certificate* in section 4.4.26.2.2 TC\_LPAd\_HTTPS\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #03 Error: VOIDTest Sequence #04 Error: VOID***

***Test Sequence #05 Error: VOID***

***Test Sequence #06 Error: VOID***

***Test Sequence #07 Error: Invalid TLS Certificate based on Invalid CI (Invalid Curve)***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Invalid TLS Certificate based on Invalid CI (Invalid Curve)* in section 4.4.26.2.2 TC\_LPAd\_HTTPS\_ErrorCases where the LPAd plays the IPAe role

### ES11 (IPAe – SM-DS): InitiateAuthentication

#### Conformance Requirements

ES11 interface between the IPAe and SM-DS is identical to the LPAd and SM-DS.

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### Test Cases

##### TC\_IPAe\_ES11\_InitiateAuthentication\_Nominal

***Test Sequence #01 Nominal: Initiate Authentication***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Initiate Authentication* in section 4.4.27.2.1 TC\_LPAd\_ES11\_InitiateAuthentication\_Nominal where the LPAd plays the IPAe role

##### TC\_IPAe\_ES11\_InitiateAuthentication\_ErrorCases

***Test Sequence #01 Error: Invalid SM-DS Address***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Initiate Authentication* in section 4.4.27.2.2 TC\_LPAd\_ES11\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #02 Error: Unsupported Security Configuration***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Unsupported Security Configuration* in section 4.4.27.2.2 TC\_LPAd\_ES11\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #03 Error: Unsupported SVN***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Unsupported SVN* in section 4.4.27.2.2 TC\_LPAd\_ES11\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #04 Error: Unavailable SM-DS Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Unavailable SM-DS Certificate* in section 4.4.27.2.2 TC\_LPAd\_ES11\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #05 Error: Invalid SM-DS Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Error: Invalid SM-DS Certificate* in section 4.4.27.2.2 TC\_LPAd\_ES11\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #06 Error: Invalid SM-DS Signature***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Error: Invalid SM-DS Signature* in section 4.4.27.2.2 TC\_LPAd\_ES11\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #07 Error: Invalid SM-DS Address sent by the SM-DS***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Invalid SM-DS Address sent by the SM-DS*  in section 4.4.27.2.2 TC\_LPAd\_ES11\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #08 Error: Unsupported CI Key ID***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #08 Error: Unsupported CI Key ID* in section 4.4.27.2.2 TC\_LPAd\_ES11\_InitiateAuthentication\_ErrorCases where the LPAd plays the IPAe role

### ES11 (IPAe – SM-DS): AuthenticateClient

#### Conformance Requirements

ES11 interface between the IPAe and SM-DS is identical to the LPAd and SM-DS.

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### Test Cases

##### TC\_IPAe\_ES11\_AuthenticateClient\_Nominal

***Test Sequence #01 Nominal: Authenticate Client with empty MatchingID***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Authenticate Client with empty MatchingID* in section 4.4.27.2.2 LPAd\_ES11\_AuthenticateClient\_Nominal where the LPAd plays the IPAe role

***Test Sequence #02 Nominal: Authenticate Client with MatchingID set to EventID***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Authenticate Client with empty MatchingID* section 4.4.27.2.2 LPAd\_ES11\_AuthenticateClient\_Nominal where the LPAd plays the IPAe role

##### TC\_IPAe\_ES11\_AuthenticateClient\_ErrorCases

***Test Sequence #01 Error: Invalid EUM Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Invalid EUM Certificate* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #02 Error: Expired EUM Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Expired EUM Certificate* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #03 Error: Invalid eUICC Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Invalid eUICC Certificate* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #04 Error: Expired eUICC Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Expired eUICC Certificate* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #05 Error: Invalid eUICC signature or serverChallenge***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Expired eUICC Certificate* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #06 Error: Unknown TransactionID***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Expired eUICC Certificate* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #07 Error: Unknown Event Record***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Unknown Event Record* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

### ES11 (IPAe -- SM-DS): HTTPS

#### Conformance Requirements

ES11 interface between the IPAe and SM-DS is identical to the LPAd and SM-DS.

**References**

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

#### Test Cases

##### TC\_IPAe\_ES11\_HTTPS\_Nominal

***Test Sequence #01 Nominal: HTTPS Session Establishment***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: HTTPS Session Establishment* section 4.4.29.2.1 TC\_LPAd\_ES11\_HTTPS\_Nominal where the LPAd plays the IPAe role

***Test Sequence #02 Nominal: non-reuse of session keys***

The purpose of this test sequence is to verify that the IPAe is not reusing ephemeral keys from the previous session.

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: non-reuse of session keys* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

##### TC\_IPAe\_ES11\_HTTPS\_Error

***Test Sequence #01 Error: Invalid (SM-DS) TLS Certificate signature***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Invalid (SM-DS) TLS Certificate signature* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #02 Error: Expired TLS Certificate***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Expired TLS Certificate* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

***Test Sequence #03 Error: VOID***

***Test Sequence #04 Error: VOID***

***Test Sequence #05 Error: VOID***

***Test Sequence #06 Error: VOID***

***Test Sequence #07 Error: Invalid TLS Certificate based on Invalid CI (Invalid Curve)***

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Invalid TLS Certificate based on Invalid CI (Invalid Curve)* section 4.4.28.2.2 TC\_LPAd\_ES11\_AuthenticateClient\_ErrorCases where the LPAd plays the IPAe role

# 5 Procedure - Behaviour Testing

## 5.1 General Overview

NOTE: Those test cases related to eUICC Behaviour are FFS.

# 7 External Test Specifications

Some test specifications related to the RSP ecosystem have been developed by external organisations (e.g. TCA (formerly SIMalliance)). These organisations defined their own requirements for test benches, test applicability and pass criteria.

This section lists the test specifications that relate to SGP.22 [4] and SGP.32 [3].

## 7.1 TCA eUICC Profile Package Test Specification

The TCA eUICC Profile Package: Interoperable Format Test Specification [23] SHALL be executed on the eUICC in order to check its compliance with the eUICC Profile Package Specification [4].

Test cases are applicable according to the eUICC Profile Package Specification [4] version and the additional eUICC Profile Package Specification [4] versions (if any) supported by the eUICC, in conjunction with the applicability table of the referred Test Specification [23].

The table below describes the versions of the eUICC Profile Package Specification [4] allowed depending on the SGP.22 and SGP.32 versions supported by the eUICC:

|  |  |  |  |
| --- | --- | --- | --- |
| SGP.22 version | SGP.32 version | eUICC Profile Package Specification [4] versions required for the given SGP.22 version | Allowed values for #IUT\_EUICC\_ADD\_PP\_VERSIONS |
| 2.5 | 1.0 | 2.1 or 2.2 or 2.3.1 | 3.1 or 3.2 or 3.3.1 |

Moreover, eUICC Manufacturers SHALL declare that the following options (as defined in [23]) are supported by the eUICC:

 O\_MILENAGE

 O\_TUAK\_128

 O\_JAVACARD

Annex A Constants

A.1 Generic Constants

| Name | Content |
| --- | --- |
| AUTO\_ENABLE\_FLAG | NULL |
| EID1 | -- as defined in SGP.23 [8] |
| EID2 | -- as defined in SGP.23 [8] |
| EIM\_CONFIG\_DATA\_NO\_EIM\_ID | EimConfiguraitonData ::= {  counterValue 1  } |
| EIM\_CONFIG\_DATA1 | EimConfigurationData ::= {  eimId #TEST\_EIM\_ID1,  eimIdType 3, -- proprietary  counterValue 1,  eimPublicKeyData #EIM\_PUBLIC\_KEY\_DATA\_PK  } |
| EIM\_CONFIG\_DATA1\_UPDATE\_COUNTER | EimConfiguraitonData ::= {  eimId #TEST\_EIM\_ID1,  counterValue 99  } |
| EIM\_CONFIG\_DATA1\_UPDATE\_COUNTER\_OUT\_OF\_RANGE | EimConfigurationData ::= {  eimId #TEST\_EIM\_ID1,  counterValue #EIM\_COUNTER\_OUT\_OF\_RANGE  } |
| EIM\_CONFIG\_DATA1\_UPDATE\_LOWER\_COUNTER | EimConfigurationData ::= {  eimId #TEST\_EIM\_ID1,  counterValue 0  } |
| EIM\_CONFIG\_DATA1\_UPDATE\_UKNOWN\_EUICC\_CI\_PKID | EimConfigurationData ::= {  eimId #TEST\_EIM\_ID1,  euiccCiPKId #CI\_PKI\_RANDOM  } |
| EIM\_CONFIG\_DATA2 | EimConfigurationData ::= {  eimId #TEST\_EIM\_ID2,  eimIdType 3, -- proprietary  counterValue 1,  eimPublicKeyData #EIM\_PUBLIC\_KEY\_DATA\_PK  } |
| EIM\_CONFIG\_DATA2\_ADD\_COUNTER\_OUT\_OF\_RANGE | EimConfigurationData ::= {  eimId #TEST\_EIM\_ID2,  eimIdType 3, -- proprietary  counterValue #EIM\_COUNTER\_OUT\_OF\_RANGE  eimPublicKeyData #EIM\_PUBLIC\_KEY\_DATA\_PK  } |
| EIM\_CONFIG\_DATA2\_ADD\_UKNOWN\_EUICC\_CI\_PKID | EimConfigurationData ::= {  eimId #TEST\_EIM\_ID2,  eimIdType 3, -- proprietary  counterValue 1,  eimPublicKeyData #EIM\_PUBLIC\_KEY\_DATA\_PK  euiccCiPKId #CI\_PKI\_RANDOM  } |
| EIM\_CONFIG\_DATA2\_UPDATE\_COUNTER | EimConfiguraitonData ::= {  eimId #TEST\_EIM\_ID2,  counterValue 99  } |
| EIM\_CONFIG\_DATA2\_EIM\_ID\_129\_CHARACTERS | EimConfiguraitonData ::= {  eimId #TEST\_EIM\_ID129,  counterValue 1  } |
| EIM\_COUNTER\_OUT\_OF\_RANGE | 8388608 (0x800000) |
| EIM\_PUBLIC\_KEY\_DATA\_PK | eimPublicKey #PK\_S\_EIMsign\_ECDSA |
| ICCID\_OP\_PROF1 | -- as defined in SGP.23 [8] |
| ICCID\_OP\_PROF2 | -- as defined in SGP.23 [8] |
| ICCID\_OP\_PROF3 | -- as defined in SGP.23 [8] |
| ICCID\_OP\_PROFX | -- as defined in SGP.23 [8] |
| ICON\_OP\_PROF1 | -- as defined in SGP.23 [8] |
| ICON\_OP\_PROF2 | -- as defined in SGP.23 [8] |
| ICON\_OP\_PROF3 | -- as defined in SGP.23 [8] |
| LIST\_EIM\_RESULT | {  {eimId #TEST\_EIM\_ID1},  {eimId #TEST\_EIM\_ID2}  } |
| NAME\_OP\_PROF1 | -- as defined in SGP.23 [8] |
| NAME\_OP\_PROF2 | -- as defined in SGP.23 [8] |
| NAME\_OP\_PROF3 | -- as defined in SGP.23 [8] |
| PROFILE\_TAG\_INVALID | 0xAA |
| SP\_NAME1 | -- as defined in SGP.23 [8] |
| SP\_NAME2 | -- as defined in SGP.23 [8] |
| SP\_NAME3 | -- as defined in SGP.23 [8] |
| S\_SM\_DP+\_OID1 | -- as defined in SGP.23 [8] |
| S\_SM\_DP+\_OID2 | -- as defined in SGP.23 [8] |
| SMDP\_ADDRESS1 | 0x 73 6D 64 70 33 33 2E 67 73 6D 61 2E 63 6F 6D |
| TAGS\_AID\_PROFILENAME | -- as defined in SGP.23 [8] |
| TEST\_EIM\_ID1 | testeim1 |
| TEST\_EIM\_ID2 | testeim2 |
| TEST\_EIM\_ID129  (129 characters) | testeim1234567890abcdefghijklmnopqrstuvwxyz1234567890abcdefghijklmnopqrstuvwxyz1234567890abcdefghijklmnopqrstuvwxyz1234567890abcd |

A.2 Test Certificates and Test Keys

All ECC certificates and keys described below are based on either:

 NIST P-256 curve, defined in Digital Signature Standard [11]

 brainpoolP256r1 curve, defined in RFC 5639 [8]

NOTE: SGP.26 [25] contains test keys, valid test certificates and instructions for how to generate invalid certificates. Unless specified differently, the test keys and test certificates used in the present document are bundled with SGP.26 [25].

|  |  |
| --- | --- |
| Name | Description |
| CERT\_EUICC\_SIG | -- as CERT\_EUICC\_ECDSA defined in SGP.23 [8]. |
| CERT\_EUM\_SIG | -- as CERT\_EUM\_ECDSA defined in SGP.23 [8] |
| CERT\_S\_EIMsign\_ ECDSA | Certificate of S EIM |
| PK\_EUICC\_SIG | -- as PK\_EUICC\_ECDSA defined in SGP.23 [8]. |
| PK\_EUM\_SIG | Public Key of the eUICC, contained within  #CERT\_EUM\_SIG. |
| PK\_S\_EIMsign\_ECDSA | Public Key of S\_EIM contained within #CERT\_S\_EIMsign\_ ECDSA |

Annex B Dynamic Content



| **Variable** | **Description** |
| --- | --- |
| CI\_KEY\_ID\_BP256 | Subject Key Identifier of the PK CI Key ID based on BrainpoolP256r1 curve |
| CI\_PKI\_RANDOM | Random Subject Key Identifier of the PK CI different from all the PK CI Identifiers defined in SGP.26 [25]. This random value has the same length as the ones defined in SGP.26 [25]. |
| EUICC\_CI\_PK\_ID\_TO\_BE\_USED | -- as defined in SGP.23 [XX] |
| EUICC\_SIGN\_EPR | EuiccPackageResult signature computed as defined in SGP.32 [X]. |
| ISD\_P\_AID1 | -- as defined in SGP.23 [8] |
| ISD\_P\_AID2 | -- as defined in SGP.23 [8] |
| ISD\_P\_AID3 | -- as defined in SGP.23 [8] |
| S\_COUNTER\_VALUE | Counter used by the eIM for replay protection |
| S\_EIM\_SIGNATURE | Signature computed as defined in SGP.32 [X]. |
| S\_EIM\_SIGN\_RANDOM | Random eIM signature with a size corresponding to a valid one as defined in SGP.32 [X]. |
| S\_LOW\_CNTR\_VALUE | Any value lower than <S\_COUNTER\_VALUE> |
| S\_TRANSACTION\_ID | The transactionId which allows the eIM to link a received eUICC Package Result to a sent eUICC Package. |
| SEQ\_NUMBER | The sequnce number used to return either a Notification or an eUICC Package Result |

Annex C Methods And Procedures

This section describes methods and procedures used in the interfaces compliance test cases. They are part of test cases and SHALL not be executed in standalone mode.

C.1 Methods

If the method is used in the “expected result” column, all parameters SHALL be verified by the simulated entity (test tool). If the method is used in the “Sequence / Description” column, the command SHALL be generated by the simulated entity.

|  |  |
| --- | --- |
| Method | MTD\_ADD\_INITIAL\_EIM\_SINGLE |
| Description | Generate the ASN.1 AddInitialEimRequest structure with a single EIM. |
| Parameter(s) |  paramAddInitialEimData: list of eimConfigurationData data |
| Details | AddInitialEimRequest ::= {  eimConfigurationDataList {  paramAddInitialEimData  } |

|  |  |
| --- | --- |
| **Method** | MTD\_EUICC\_PACKAGE\_REQUEST\_DISABLE |
| Description | Generate the ASN.1 EuiccPackageRequest structure according to the input parameters. |
| Parameter(s) | * paramEIMId :  identifier of the eIM that issues the eUICC Package. * paramEidValue: value of EID of the targeted eUICC * paramCounterValue: used by the eIM for replay protection * paramTransactionId: optional, used to uniquely identify the RSP session * paramIccidValue: The ICCID of the Profile to Enable. * paramEimSignature: Signature computed as defined in SGP.32 [X]. |
| Details | IF TransactionId is provided Then  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  transactionId paramTransactionId,  euiccPackage psmoList : {  disable : {  iccid paramIccidValue  }  }  },  eimSignature paramEimSignature  }  End if  IF TransactionId is not provided Then  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  disable : {  iccid paramIccidValue  }  }  },  eimSignature paramEimSignature  }  End if |

|  |  |
| --- | --- |
| **Method** | MTD\_CONFIGURE\_AUTOMATIC\_PROFILE\_ENABLING |
| Description | Generate the ASN.1 ConfigureAutomaticProfileEnabling structure according to the input parameters. |
| Parameter(s) | * autoEnableFlag: optional, this flag indicates whether the automatic enable should be active or not. * paramSmdpOid: optional, value of smdpOid for the default SM-DP+ * paramSmdpAddress: optional, value of smdpAddress for the default SM-DP+ |
| Details | IF autoEnableFlag, paramSmdpOid and paramSmdpAddress are provided then  req configureAutoEnable : {  autoEnableFlag NULL,  smdpOid paramSmdpOid,  smdpAddress paramSmdpAddress  }  End if  IF autoEnableFlag is not present, paramSmdpOid and paramSmdpAddress are provided then  req configureAutoEnable : {  smdpOid paramSmdpOid,  smdpAddress paramSmdpAddress  }  End if |

|  |  |
| --- | --- |
| **Method** | MTD\_ENABLE\_USING\_DD |
| Description | Generate the ASN.1 EnableUsingDDRequest structure. |
| Parameter(s) | no parameters |
| Details | req EnableUsingDDRequest ::= {  } |

|  |  |
| --- | --- |
| **Method** | MTD\_EUICC\_PACKAGE\_REQUEST\_ADD\_EIM |
| Description | Generate the ASN.1 EuiccPackageRequest structure according to the input parameters. |
| Parameter(s) | * paramEIMId : identifier of the eIM that issues the eUICC Package * paramEidValue: value of EID of the targeted eUICC * paramCounterValue: used by the eIM for replay protection * paramEimTransactionId: optional, used by the eIM to uniquely identify an RSP session * paramEimConfigData: eIM Configuration Data to be added * paramEimSignature: Signature computed as defined in SGP.32 [X] |
| Details | IF paramEimTransactionId is provided THEN  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  eimTransactionId paramEimTransactionId,  euiccPackage ecoList : {  addEim paramEimConfigData  }  },  eimSignature paramEimSignature  }  ELSE THEN  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage ecoList : {  addEim paramEimConfigData  }  },  eimSignature paramEimSignature  }  END IF |

|  |  |
| --- | --- |
| **Method** | MTD\_EUICC\_PACKAGE\_GET\_RAT |
| Description | Generate the ASN.1 EuiccPackageRequest structure according to the input parameters. |
| Parameter(s) | * paramEIMId :  identifier of the eIM that issues the eUICC Package. * paramEidValue: value of EID of the targeted eUICC * paramCounterValue: used by the eIM for replay protection |
| Details | req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  getRAT : {  }  }  },  eimSignature paramEimSignature  } End if |

|  |  |
| --- | --- |
| **Method** | MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE\_EIM |
| Description | Generate the ASN.1 EuiccPackageRequest structure according to the input parameters. |
| Parameter(s) | * paramEIMId : identifier of the eIM that issues the eUICC Package. * paramEidValue: value of EID of the targeted eUICC * paramCounterValue: used by the eIM for replay protection * paramEimTransactionId: optional, used by the eIM to uniquely identify an RSP session * paramTargetEIMId: identifier of the eIM of which eIM Configuration Data is to be deleted * paramEimSignature: Signature computed as defined in SGP.32 [X]. |
| Details | IF paramEimTransactionId is provided THEN  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  eimTransactionId paramEimTransactionId,  euiccPackage ecoList : {  deleteEim paramTargetEIMId  }  },  eimSignature paramEimSignature  }  ELSE THEN  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage ecoList : {  deleteEim paramTargetEIMId  }  },  eimSignature paramEimSignature  }  END IF |

|  |  |
| --- | --- |
| **Method** | MTD\_EUICC\_PACKAGE\_REQUEST\_LIST\_EIM |
| Description | Generate the ASN.1 EuiccPackageRequest structure according to the input parameters. |
| Parameter(s) | * paramEIMId : identifier of the eIM that issues the eUICC Package. * paramEidValue: value of EID of the targeted eUICC * paramCounterValue: used by the eIM for replay protection * paramEimTransactionId: optional, used by the eIM to uniquely identify an RSP session * paramEimSignature: Signature computed as defined in SGP.32 [X]. |
| Details | IF paramEimTransactionId is provided THEN  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  eimTransactionId paramEimTransactionId,  euiccPackage ecoList : {  listEim { }  }  },  eimSignature paramEimSignature  }  ELSE THEN  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage ecoList : {  listEim : { }  }  },  eimSignature paramEimSignature  }  END IF |

|  |  |
| --- | --- |
| **Method** | MTD\_EUICC\_PACKAGE\_REQUEST\_LIST\_PRO\_FILE\_INFO |
| Description | Generate the ASN.1 EuiccPackageRequest structure according to the input parameters. |
| Parameter(s) | * paramEIMId:  identifier of the eIM that issues the eUICC Package. * paramEidValue: value of EID of the targeted eUICC * paramCounterValue: used by the eIM for replay protection * paramIccidValue: optional ICCID of target Profile. * paramIsdpAid: optional AID of the ISD-P * paramProfileClass: optional Profile Class * paramTagList: optional tagList * paramEimSignature: Signature computed as defined in SGP.32 [X]. |
| Details | IF all optional parameters are not provided Then  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId ,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  listProfileInfo : {  }  }  },  eimSignature paramEimSignature  } End if  IF paramIsdpAid is provided and the other parameters are not providedThen  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId ,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  listProfileInfo : {  searchCriteria isdpAid : paramIsdpAid  }  }  },  eimSignature paramEimSignature  } End if  IF paramIccid is provided and the other parameters are not providedThen  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId ,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  listProfileInfo : {  searchCriteria iccid : paramIccidValue  }  }  },  eimSignature paramEimSignature  } End if  IF paramProfileClass is provided and the other parameters are not providedThen  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  listProfileInfo : {  searchCriteria profileClass: paramProfileClass  }  }  },  eimSignature paramEimSignature  } End if  IF tagList is provided and the other parameters are not providedThen  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  listProfileInfo : {  tagList tagList  }  }  },  eimSignature paramEimSignature  } End if |

|  |  |
| --- | --- |
| **Method** | MTD LIST\_PROFILE\_INFO CHANGE\_ONGOING\_ERROR |
| Description | Generate the ASN.1 EuiccPackageRequest structure according to the input parameters. |
| Parameter(s) | * paramEIMId :  identifier of the eIM that issues the eUICC Package. * paramEidValue: value of EID of the targeted eUICC * paramCounterValue: used by the eIM for replay protection * paramEimSignature: Signature computed as defined in SGP.32 [X]. |
| Details | req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  enable : {  iccid ICCID\_OP\_PROF2  },  listProfileInfo : {  searchCriteria iccid : ICCID\_OP\_PROF2  }  }  },  eimSignature paramEimSignature  } End if |

|  |  |
| --- | --- |
| **Method** | MTD\_EUICC\_PACKAGE\_REQUEST\_UPDATE\_EIM |
| Description | Generate the ASN.1 EuiccPackageRequest structure according to the input parameters. |
| Parameter(s) | * paramEIMId : identifier of the eIM that issues the eUICC Package. * paramEidValue: value of EID of the targeted eUICC * paramCounterValue: used by the eIM for replay protection * paramEimTransactionId: optional, used by the eIM to uniquely identify an RSP session * paramEimConfigData: eIM Configuration Data to be updated * paramEimSignature: Signature computed as defined in SGP.32 [X]. |
| Details | IF paramEimTransactionId is provided THEN  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  eimTransactionId paramEimTransactionId,  euiccPackage ecoList : {  updateEim paramEimConfigData  }  },  eimSignature paramEimSignature  }  ELSE THEN  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage ecoList : {  updateEim : paramEimConfigData  }  },  eimSignature paramEimSignature  }  END IF |

|  |  |
| --- | --- |
| **Method** | MTD\_EUICC\_PACKAGE\_REQUEST\_ENABLE |
| Description | Generate the ASN.1 EuiccPackageRequest structure according to the input parameters. |
| Parameter(s) | * paramEIMId :  identifier of the eIM that issues the eUICC Package. * paramEidValue: value of EID of the targeted eUICC * paramCounterValue: used by the eIM for replay protection * paramTransactionId: optional, used to uniquely identify the RSP session * paramIccidValue: The ICCID of the Profile to Enable. * paramRollbackFlag: optional rollback condition * paramEimSignature: Signature computed as defined in SGP.32 [X]. |
| Details | IF TransactionId and rollbackFlag are provided Then  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId ,  eidValue paramEidValue,  counterValue paramCounterValue,  transactionId paramTransactionId,  euiccPackage psmoList : {  enable : {  iccid paramIccidValue,  rollbackFlag paramRollbackFlag  }  }  },  eimSignature paramEimSignature  }  End if  IF TransactionId is provided and rollbackFlag is not provided Then  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId ,  eidValue paramEidValue,  counterValue paramCounterValue,  transactionId paramTransactionId,  euiccPackage psmoList : {  enable : {  iccid paramIccidValue,  }  }  },  eimSignature paramEimSignature  }  End if  IF TransactionId is not provided and rollbackFlag is provided Then  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId ,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  enable : {  iccid paramIccidValue,  rollbackFlag paramRollbackFlag  }  }  },  eimSignature paramEimSignature  } End if  IF TransactionId and rollbackFlag are not provided Then  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId ,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  enable : {  iccid paramIccidValue,  }  }  },  eimSignature paramEimSignature  } End if |

|  |  |
| --- | --- |
| **Method** | MTD\_EUICC\_PACKAGE\_REQUEST\_DELETE |
| Description | Generate the ASN.1 EuiccPackageRequest structure according to the input parameters for the delete. |
| Parameter(s) | * paramEIMId :  identifier of the eIM that issues the eUICC Package. * paramEidValue: value of EID of the targeted eUICC * paramCounterValue: used by the eIM for replay protection * paramTransactionId: optional, used to uniquely identify the RSP session * paramIccidValue: The ICCID of the Profile to Enable. paramEimSignature: Signature computed as defined in SGP.32 [X]. |
| Details | IF TransactionId is provided  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId ,  eidValue paramEidValue,  counterValue paramCounterValue,  euiccPackage psmoList : {  delete : {  iccid paramIccidValue  }  }  },  eimSignature paramEimSignature  }  Else  req EuiccPackageRequest ::= {  euiccPackageSigned {  eimId paramEIMId ,  eidValue paramEidValue,  counterValue paramCounterValue,  transactionId paramTransactionId,  euiccPackage psmoList : {  delete : {  iccid paramIccidValue  }  }  },  eimSignature paramEimSignature  }  End If |

|  |  |
| --- | --- |
| Method | MTD\_EUICC\_PKG\_ERROR\_UNSIGNED |
| Description | Generate the ASN.1 EuiccPackageResult structure for euiccPackageErrorUnsigned |
| Parameter(s) |  paramEIMId: eIMId to be returned |
| Details | resp EuiccPackageResult ::=  euiccPackageErrorUnsigned : {  eimId [0] paramEIMId  } |

|  |  |
| --- | --- |
| Method | MTD\_EUICC\_PKG\_ ERROR\_SIGNED |
| Description | Generate the ASN.1 EuiccPackageResult structure euiccPackageErrorSigned. |
| Parameter(s) |  paramEuiccPackageResultErrorCode: The Package Result Error code (Mandatory) |
| Details | resp EuiccPackageResult ::=  euiccPackageErrorSigned : {  euiccPackageErrorDataSigned {  eimId [0] <S\_EIM\_ID>,  counterValue [1] <S\_COUNTER\_VALUE>,  transactionId[2] <S\_TRANSACTION\_ID>,  euiccPackageErrorCode paramEuiccPackageResultErrorCode  },  euiccSignEPR <EUICC\_SIGNATURE>  } |

|  |  |
| --- | --- |
| Method | MTD\_GET\_CERTS\_REQUEST |
| Description | Generate the ASN.1 GetCerts structure |
| Parameter(s) |  paramEuiccCiPkiId: Id of the targeted eUICC CI PKI (Optional) |
| Details | GetCertsRequest ::= {  euiccCiPKId paramEuiccCiPkiId  } |

|  |  |
| --- | --- |
| Method | MTD\_GET\_EIM |
| Description | Generate the ASN.1 GetEimRequest structure. |
| Parameter(s) |  |
| Details | GetEimConfigurationDataRequest ::= {  } |

|  |  |
| --- | --- |
| **Method** | MTD\_GET\_PROFILE\_INFO |
| Description | Generate the ASN.1 ProfileInfoListRequest according to the input parameters |
| Parameter(s) | * paramIccidValue: The ICCID of the Profile * paramIsdpAidValue: The ISD-P AID of the Profile   Either paramIccidValue or paramIsdpAidValue is passed as a parameter. |
| Details | IF paramIccidValue is provided Then  req ProfileInfoListRequest::= {  searchCriteria iccid: paramIccidValue  }  Else  req ProfileInfoListRequest::= {  searchCriteria isdpAid: paramIsdpAidValue  }  End If |

|  |  |
| --- | --- |
| Method | MTD\_PROFILE\_ROLLBACK\_REQUEST |
| Description | Generate the ASN.1 ProfileRollbackRequest according to the input parameters. |
| Parameter(s) | · paramRefreshFlag: indicating whether REFRESH is required |
| Details | ProfileRollbackRequest ::= {  refreshFlag paramRefreshFlag  } |

|  |  |
| --- | --- |
| Method | MTD\_SEND\_SMS\_PP |
| Description | Generate and send an envelope SMS-PP download to the MNO-SD |
| Parameter(s) | · paramApdusList: the list of APDUs (plain) to send |
| Details | Generate and send the following envelope:  80 C2 00 00 <L>  D1 <L>  02 02 83 81 *-- Device identity Tag*  06 07 91 33 86 09 40 00 F0 *-- Address Tag (TON/NPI/..)*  0B <L> -- SMS TPDU  44 -- SMS-DELIVER  05 85 02 13 F2 -- TP-Originating-Address  7F -- TP-Protocol-Identifier  F6 -- TP-Data-Coding-Scheme  71 30 12 41 55 74 40 -- TP-Service-Centre-Time-Stamp  <L> -- TP-User-Data-Length  02 -- User-Data-Header-Length  70 -- IEIa  00 -- IEIDLa  <L> -- Command Packet Length (2 bytes)  <L> -- Command Header Length (1 byte)  12 21 -- SPI  00 -- KIC  15 –- KID (SCP80 Keyset version 0x01 in Triple DES)  B2 01 00 –- MNO-SD TAR  <MNO\_SCP80\_COUNTER>  00 -- Padding Counter  <CC> -- Cryptographic checksum  <C\_APDUS\_SCRIPT> -- Command APDUs script  <C\_APDUS\_SCRIPT> SHALL contain the paramApdusList (i.e. each APDU is named <APDU1>; <APDU2>; …; <APDUn> here after) formatted as an expanded structure with definite length as defined in ETSI TS 102 226 [14]:  AA <L>  22 <L> <APDU1>  22 <L> <APDU2>  …  22 <L> <APDUn>  The Cryptographic checksum <CC> SHALL be generated in Triple DES (outer-CBC mode using two different keys) with the #MNO\_SCP80\_AUTH\_KEY as defined in ETSI TS 102 225 [13].  If the command packet length is higher than 140 bytes, it SHALL be sent over several envelopes: SMS concatenation as defined in 3GPP TS 23.040 [22] SHALL be used. |

|  |  |
| --- | --- |
| Method | MTD\_SELECT |
| Description | Generates the SELECT command as defined in GlobalPlatform Card Specification [6]. |
| Parameter(s) |  paramAID: the AID to select |
| Details | - CLA = 0x or 4x (x = <CHANNEL\_NUMBER>)  - INS = A4  - P1 = 04  - P2 = 00  - LC = <L>  - paramAID  - LE = 00 |

|  |  |
| --- | --- |
| Method | MTD\_STORE\_DATA |
| Description | Generates the STORE DATA command (Case 4) as defined in GlobalPlatform Card Specification [X]. |
| Parameter(s) | * paramCommandData: the command data |
| Details | - CLA = 8x or Cx (x = <CHANNEL\_NUMBER>)  - INS = E2  - P1 = 91  - P2 = 00  - LC = <L>  - paramCommandData  - LE = 00 |

C.2 Procedures

|  |  |  |  |
| --- | --- | --- | --- |
|  | Procedure | PROC\_EUICC\_INITIALIZATION\_SEQUENCE | |
|  | Description | Initialize communication between the S\_Device and the eUICC. | |
| Step | Direction | Sequence / Description | Expected result |
| 1 | S\_Device → eUICC | RESET | ATR present |
| 2 | S\_Device → eUICC | [SELECT\_MF] | FCP Template present  SW=0x9000 |
| 3 | S\_Device → eUICC | [TERMINAL\_CAPABILITY\_LPAd] | SW=0x9000 |
| 4 | S\_Device → eUICC | [TERMINAL\_PROFILE] | Toolkit initialization THEN SW=0x9000 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Procedure | PROC\_OPEN\_LOGICAL\_CHANNEL\_AND\_SELECT\_ISDR | |
|  | Description | The LPAd opens a logical channel and selects the ISD-R. | |
| Step | Direction | Sequence / Description | Expected result |
| 1 | S\_LPAd → eUICC | [MANAGE\_CHANNEL\_OPEN] | Extract the <CHANNEL\_NUMBER> from response data  SW=0x9000 |
| 2 | S\_LPAd → eUICC | MTD\_SELECT(#ISD\_R\_AID) | SW=0x9000 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Procedure | PROC\_EUICC\_INITIALIZATION\_SEQUENCE\_eUICCProfileStateChanged | |
|  | Description | Initialize communication between the S\_Device and the eUICC. | |
| Step | Direction | Sequence / Description | Expected result |
| 1 | S\_Device → eUICC | RESET | ATR returned by eUICC |
| 2 | S\_Device → eUICC | [SELECT\_MF] | FCP Template present  SW=0x9000 |
| 3 | S\_Device → eUICC | [TERMINAL\_CAPABILITY\_LPAd] | SW=0x9000 |
| 4 | S\_Device → eUICC | [TERMINAL\_PROFILE\_eUICCProfileStateChanged] | Toolkit initialization THEN SW=0x9000 |

Annex D Commands And Responses

D.1 ES10x Requests And Responses

D.1.1 ES10x Requests

|  |  |
| --- | --- |
| Name | Content |
| GET\_PROFILES\_INFO\_ALL | request ProfileInfoListRequest ::= { } |
| LOAD\_EUICC\_PKG\_ INVALID\_ECO | {  ecoList : {  update\_Eim,  #EIM\_CONFIG\_DATA1\_UPDATE\_COUNTER  }  } |
| LOAD\_EUICC\_PKG\_ INVALID\_PSMO | {  psmoList : {  remove : {  iccid #ICCID\_OP\_PROF1  }  }  } |

D.1.2 ES10x Responses

|  |  |
| --- | --- |
| Name | Content |
| ADD\_INITIAL\_EIM\_RES\_OK | addInitialEimOk {  addOk  } |
| ADD\_INITIAL\_EIM\_ ERR\_EIM\_CONF | addInitialEimError {  unsignedEimConfigDisallowed  } |
| ADD\_INITIAL\_EIM\_ERR\_UNKNOWN\_CI | addInitialEimError {  ciPKUnknown  } |
| ADD\_INITIAL\_EIM\_ERR\_CNTR\_OUT\_OF\_RANGE | addInitialEimError {  counterValueOutOfRange  } |
| CONFIG\_AUTO\_ENABLE\_RES\_OK | response ConfigureAutoProfileEnablingResponse ::= {  configAutoEnableResult 0  } |
| CONFIG\_AUTO\_ENABLE\_RES\_CONFIG\_DISALLOWED | response ConfigureAutoProfileEnablingResponse ::= {  configAutoEnableResult 2  } |
| DELETE\_PROFILE\_ICCID\_NOT**\_**FOUND | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  deleteResult : 1  }  },  euiccSignEPR paramEimSignature |
| DELETE\_PROFILE\_ICCID\_NOT**\_**IN\_DISABLE\_STATE | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  deleteResult : 2  }  },  euiccSignEPR paramEimSignature |
| DELETE\_RES\_OK\_1 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  deleteResult: 0  }  },  euiccSignEPR paramEimSignature |
| DELETE\_RES\_OK\_2 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  transactionId S\_TRANSACTION\_ID,  seqNumber SEQ\_NUMBER,  euiccResult {  deleteResult: 0  }  },  euiccSignEPR paramEimSignature |
| DISABLE\_RES\_OK\_1 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  disableResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| DISABLE\_RES\_OK\_2 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  transactionId paramTransactionId  seqNumber SEQ\_NUMBER,  euiccResult {  disableResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| DISABLE\_RES\_PROFILE\_ICCID\_NOT\_FOUND | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  disableResult : 1  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| DISABLE\_RES\_PROFILE\_NOT\_IN\_ENABLED\_STATE | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  disableResult : 2  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| ENABLE\_RES\_OK\_1 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  enableResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| ENABLE\_RES\_OK\_2 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  transactionId S\_TRANSACTION\_ID,  seqNumber SEQ\_NUMBER,  euiccResult {  enableResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| ENABLE\_RES\_PROFILE\_ICCID\_NOT\_FOUND | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  enableResult : 1  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| ENABLE\_RES\_PROFILE\_NOT\_IN\_DISABLE\_STATE | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  enableResult : 2  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| ENABLE\_RES\_PROFILE\_ UNDEFINED\_ERROR | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  enableResult : 127  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| ENABLE\_USING\_DD\_RESULT\_OK | enableUsingDDResult 0 |
| ENABLE\_USING\_DD\_RESULT\_AUTO\_ENABLE\_NOT\_AVAILABLE | enableUsingDDResult 1 |
| ENABLE\_USING\_DD\_RESULT\_NO\_SESSION\_CONTEXT | enableUsingDDResult 4 |
| GET\_CERTS\_RES\_OK | certs {  eumCertificate #CERT\_EUM\_SIG,  euiccCertificate #CERT\_EUICC\_SIG  } |
| GET\_CERTS\_ERR\_UNKNOWN\_CI | getCertsError {  invalidCiPKId  } |
| GET\_EIM\_CONFIG\_DATA\_LIST\_EMPTY | eimConfigurationDataList {} |
| GET\_EIM\_CONFIG\_DATA\_LIST1 | eimConfigurationDataList {  {  eimId #TEST\_EIM\_ID1,  eimIdType 3, -- proprietary  eimSupportedProtocol 1, -- proprietary  eimPublicKeyData #EIM\_PUBLIC\_KEY\_DATA\_PK  }  } |
| GET\_EIM\_CONFIG\_DATA\_LIST2 | eimConfigurationDataList {  {  eimId #TEST\_EIM\_ID1,  eimIdType 3, -- proprietary  eimSupportedProtocol 1, -- proprietary  eimPublicKeyData #EIM\_PUBLIC\_KEY\_DATA\_PK  },  {  eimId #TEST\_EIM\_ID2,  eimIdType 3, -- proprietary  eimSupportedProtocol 1, -- proprietary  eimPublicKeyData #EIM\_PUBLIC\_KEY\_DATA\_PK  }  } |
| LOAD\_EUICC\_PKG\_ ERR\_EID | { invalidEid } |
| LOAD\_EUICC\_PKG\_ ERR\_LOW\_CNTR | { counterValueOutOfRange } |
| LOAD\_EUICC\_PKG\_ ERR\_ UNKNOWN\_CMD | euiccResult : {  processingTerminated {  unknownOrDamagedCommand  }  } |
| PACKAGE\_RES\_ERROR\_CODE\_REPLAY\_ERROR | euiccPackageErrorDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  euiccPackageErrorCode 4  },  euiccSignEPE EUICC\_SIGN\_EPR |
| PPRS\_ALLOWED  (ProfilePolicyAuthorisationRule) | -- as defined in SGP.23[XX] |
| PROFILE\_INFO1 (ProfileInfo) | {  iccid #ICCID\_OP\_PROF1,  isdpAid <ISD\_P\_AID1>,  profileState enabled,  serviceProviderName #SP\_NAME1,  profileName #NAME\_OP\_PROF1,  iconType png,  icon #ICON\_OP\_PROF1,  profileClass operational  } |
| PROFILE\_INFO1\_DISABLED (ProfileInfo) | {  iccid #ICCID\_OP\_PROF1,  isdpAid <ISD\_P\_AID1>,  profileState disabled,  serviceProviderName #SP\_NAME1,  profileName #NAME\_OP\_PROF1,  iconType png,  icon #ICON\_OP\_PROF1,  profileClass operational  } |
| PROFILE\_INFO1\_DISABLED  (ProfileInfo) | {  iccid #ICCID\_OP\_PROF1,  isdpAid <ISD\_P\_AID1>,  profileState disabled,  serviceProviderName #SP\_NAME1,  profileName #NAME\_OP\_PROF1,  iconType png,  icon #ICON\_OP\_PROF1,  profileClass operational  } |
| PROFILE\_INFO2  (ProfileInfo) | -- as defined in SGP.23 [XX] |
| PROFILE\_INFO3  (ProfileInfo) | -- as defined in SGP.23 [XX] |
| PROFILE\_INFO2\_ENABLED  (ProfileInfo) | {  iccid #ICCID\_OP\_PROF2,  isdpAid <ISD\_P\_AID2>,  profileState enabled,  serviceProviderName #SP\_NAME2,  profileName #NAME\_OP\_PROF2,  iconType png,  icon #ICON\_OP\_PROF2,  profileClass operational  } |
| PROFILE\_ROLLBACK\_RES\_OK | cmdResult : 0,  eUICCPackageResult {  euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult {  rollbackResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR  } |
| PROFILE\_ROLLBACK\_RES\_ROLLBACK\_NOT\_ALLOWED | cmdResult : 1 |
| PROFILE\_ROLLBACK\_RES\_COMMAND\_ERROR | cmdResult : 7 |
| PROFILE\_INFO\_TAG\_LIST | listProfileInfoResult : profileInfoListOk : {  {  isdpAid ISD\_P\_AID1,  profileName NAME\_OP\_PROF1  },  {  isdpAid ISD\_P\_AID2,  profileName NAME\_OP\_PROF2  }  {  isdpAid ISD\_P\_AID3,  profileName NAME\_OP\_PROF3  }  } |

D.2 APDU

D.2.1 APDU Commands

|  |  |
| --- | --- |
| Name | Content |
| GET\_MNO\_SD | - CLA = 80, INS = F2, P1 = 80, P2 = 02, LC = <L>  - Data = 4F 00  - LE = 00 |
| MANAGE\_CHANNEL\_OPEN | - CLA = 00, INS = 70, P1 = 00, P2 = 00, LE = 01 |
| READ\_BINARY | - CLA = 00, INS = B0, P1 = 00, P2 = 00, LE = <L> |
| SELECT\_ICCID | - CLA = 00, INS = A4, P1 = 00, P2 = 0C, LC = 02 - Data = 2F E2 |
| SELECT\_MF | - CLA = 00, INS = A4, P1 = 00, P2 = 04, LC = <L>  - Data = 3F 00  - LE = 00 |
| SELECT\_USIM | - CLA = 00, INS = A4, P1 = 04, P2 = 04, LC = <L> - Data = #USIM\_AID  - LE = 00 |
| TERMINAL\_CAPABILITY\_LPAd | - CLA = 80, INS = AA, P1 = 00, P2 = 00, LC = <L>  - Data = A9 05 81 00 83 01 07 |
| TERMINAL\_PROFILE | - CLA = 80, INS = 10, P1 = 00, P2 = 00, LC = <L>  - Data = FF FF FF FF 7F 9D 00 DF BF 00 00 1F E2 00 00 00 C7 EB 00 00 00 01 68 00 50 00 00 00 00 00 02 00 |
| TERMINAL\_PROFILE\_eUICCProfileStateChanged | - CLA = 80, INS = 10, P1 = 00, P2 = 00, LC = <L>  - Data = FF FF FF FF FF FF 1F FF FF 03 02 FF FF 9F FF EF DF FF 0F FF 0F FF FF 0F FF 03 00 3F 7F FF 03 FF FF 20 |

D.2.2 R-APDU Chaining

During the execution of all sequences related to the eUICC testing (i.e. section 4.2), for commands where the response exceeds 256 bytes, the chaining mechanism defined in ISO/IEC 7816-4 [7], using the 61XX status word and multiple GET RESPONSE commands, SHALL be used.

As an example, the following generic sequence, which describes this mechanism, SHALL apply.

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Direction | Sequence / Description | Result |
| 1 | OCE à eUICC | Send APDU command on logical channel x | <R\_APDU\_PART1>  SW=0x61XX |
| 2 | OCE à eUICC | Send [GET\_RESPONSE] on logical channel x with LE='XX' | <R\_APDU\_PART2>  SW=0x61XX |
| 3 | OCE à eUICC | Send [GET\_RESPONSE] on logical channel x with LE='XX' | <R\_APDU\_PART3>  SW=0x61XX |
| 4 | OCE à eUICC | Send [GET\_RESPONSE] on logical channel x with LE='XX' | <R\_APDU\_PART4>  SW=0x9000  The complete response is the result of the concatenation of all R-APDUs from <R\_APDU\_PART1> to <R\_APDU\_PART4> |

D.3 ESep Requests And Responses

D.3.1 ESep Responses

|  |  |
| --- | --- |
| Name | Content |
| ADD\_EIM\_RES\_ERR\_CI\_PK\_UNKNOWN | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  addEimResult : 3  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| ADD\_EIM\_RES\_ERR\_COMMAND\_ERROR | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  addEimResult : 7  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| ADD\_EIM\_RES\_ERR\_COUNTER\_OUT\_OF\_RANGE | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  addEimResult : 6  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| ADD\_EIM\_RES\_OK\_1 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  addEimResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| ADD\_EIM\_RES\_OK\_2 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  eimEimTransactionId S\_EIM\_TRANSACTION\_ID,  seqNumber SEQ\_NUMBER,  euiccResult : {  addEimResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| DELETE\_EIM\_RES\_ERR\_COMMAND\_ERROR | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  deleteEimResult : 7  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| DELETE\_EIM\_RES\_ERR\_EIM\_NOT\_FOUND | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  deleteEimResult : 1  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| DELETE\_EIM\_RES\_ERR\_LAST\_EIM\_DELETED | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  deleteEimResult : 2  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| DELETE\_EIM\_RES\_OK\_1 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  deleteEimResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| DELETE\_EIM\_RES\_OK\_2 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  eimEimTransactionId S\_EIM\_TRANSACTION\_ID,  seqNumber SEQ\_NUMBER,  euiccResult : {  deleteEimResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| LIST\_EIM\_RES\_OK\_1 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  listEim : { LIST\_EIM\_RESULT }  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| LIST\_EIM\_RES\_OK\_2 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  eimEimTransactionId S\_EIM\_TRANSACTION\_ID,  seqNumber SEQ\_NUMBER,  euiccResult : {  listEim : { LIST\_EIM\_RESULT }  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| UPDATE\_EIM\_RES\_ERR\_COMMAND\_ERROR | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  updateEimResult : 7  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| UPDATE\_EIM\_RES\_ERR\_COUNTER\_OUT\_OF\_RANGE | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  updateEimResult : 6  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| UPDATE\_EIM\_RES\_ERR\_EIM\_NOT\_FOUND | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  updateEimResult : 1  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| UPDATE\_EIM\_RES\_ERR\_UNKNOWN\_EUICC\_CI\_PKID | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  updateEimResult : 3  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| UPDATE\_EIM\_RES\_OK\_1 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  seqNumber SEQ\_NUMBER,  euiccResult : {  updateEimResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR |
| UPDATE\_EIM\_RES\_OK\_2 | euiccPackageResultDataSigned {  eimId paramEIMId,  counterValue paramCounterValue,  eimEimTransactionId S\_EIM\_TRANSACTION\_ID,  seqNumber SEQ\_NUMBER,  euiccResult : {  updateEimResult : 0  }  },  euiccSignEPR EUICC\_SIGN\_EPR |

Annex E Profiles

Those profiles are defined in SGP.23 [8].

Annex F IUT Settings

F.1 eUICC Settings

In order to execute the test cases defined in this document, the eUICC Manufacturer SHALL deliver following settings:

|  |  |
| --- | --- |
| eUICC Setting name | Description |
| IUT\_DLOA\_URL | Discovery Base URL of the SE default DLOA Registrar as defined in GlobalPlatform DLOA specification [19] (optional) |
| IUT\_EUICC\_ADD\_PP\_VERSIONS | The expected content, if any, of the additionalEuiccProfilePackageVersions field in EUICCInfo2, coded as binary value without tag and length.  This setting is only applicable for eUICCs supporting SGP.22 v2.3 or later.  NOTE: the version(s) indicated in this field shall be version(s) listed in section 7.1 in the “Allowed values for #IUT\_EUICC\_ADD\_PP\_VERSIONS” column. |
| IUT\_EUICC\_CATEGORY | The category, if provided, SHALL be either not present or:   other(0)   or basicEuicc(1)   or mediumEuicc(2)   or contactlessEuicc(3) |
| IUT\_EUICC\_FIRMWARE\_VER | eUICC Firmware version coded as binary value (3 bytes representing major/minor/revision). |
| IUT\_GLOBALPLATFORM\_VERSION | GlobalPlatform version coded as binary value (3 bytes representing major/minor/revision, 2.3.0 or higher). The support of GlobalPlatform is considered as mandatory in the scope of this specification. |
| IUT\_PLATFORM\_LABEL | Platform\_Label as defined in GlobalPlatform DLOA specification [19] (optional) |
| IUT\_PP\_VERSION | Protection Profile version coded as binary value (3 bytes representing major/minor/revision). |
| IUT\_SAS\_ACCREDITATION\_NUMBER | SAS Accreditation Number, coded as ASN.1 UTF8String |
| IUT\_SIMA\_VERSION | Version of eUICC Profile Package Specification [4] supported by the eUICC (3 bytes representing major/minor/revision) e.g. 0x020100 |
| IUT\_TS102241\_VERSION | The ts102241 version field is coded as binary value (3 bytes representing major/minor/revision, 9.0.0 or higher). The support of Java Card is considered as mandatory in the scope of this specification.  The ts102241 Version field indicates the latest version of ETSI TS102 241[17] supported by the eUICC |
| IUT\_UICC\_CAPABILITY | Sequence is derived from ServicesList[] defined in eUICC Profile Package PEDefinitions, coded as ASN.1 BIT STRING. |

F.4 Common Settings

In order to execute the test cases defined in this document, the IUT provider SHALL deliver following settings:

|  |  |
| --- | --- |
| IUT Setting name | Description |
| IUT\_RSP\_VERSION | Version of SGP.22 supported by the IUT encoded as a string of three integers separated with dots (for example: 2.1.0).  In the scope of this specification, this value SHALL indicate one of the versions of SGP.22 for which this specification contains test cases, as specified in section 1.2. |

Annex G Initial States

Unless it is defined differently in a particular test case, the IUTs SHALL be set in the following initial state before the test case execution.

G.1 eUICC

Depending on the test cases and on the supported options, the EUM SHALL configure the eUICC according to the following Initial States. The initial conditions SHALL be restored, as described in the following subsections, after each test sequence.

G.1.1 Common Initial States

The following initial states apply for all test cases defined in this Test Plan whatever the options supported by the eUICC:

 The eUICC is configured with the ISD-R AID #ISD\_R\_AID and the EID #EID1.

 The eUICC does not contain any Profile.

 The default files system does not contain EF\_ICCID.

 The eUICC’s Pending Notifications List is empty.

 No RSP session is ongoing.

 The eUICC is configured with the default SM-DS address #TEST\_ROOT\_DS\_ADDRESS.

 The eUICC is configured without Default SM-DP+ address.

 No CRL is loaded on the eUICC.

 The ECASD is configured as defined in section G.2.2 and/or G.2.3 depending on the support of the options O\_E\_NIST and O\_E\_BRP.

o If the eUICC only supports O\_E\_NIST, the ECASD is configured as defined in section G.2.2.

o If the eUICC only supports O\_E\_BRP, the ECASD is configured as defined in section G.2.3.

o If the eUICC supports O\_E\_NIST and O\_E\_BRP, the ECASD is configured as defined in sections G.2.2 and G.2.3 (i.e. several EUM / eUICC Certificates and Keys are configured in the eUICC).

The CI, identified as highest priority in euiccCiPKIdListForSigning, is also selectable in the euiccCiPKIdListForVerification (i.e. all EUM and eUICC Certificates lead to a Root CI certificate linked to a #PK\_CI\_ECDSA contained in the eUICC).

This CI corresponds to the SubjectKeyIdentifier of one of the #CERT\_CI\_ECDSA defined in sections G.2.2 and G.2.3.

The default RAT configuration defined in section G.2.4 applies for all test sequences except if the Test Case overrides it. Particular RAT configurations for those specific Test Cases are defined in section G.2.5.

G.1.2 For eUICC supporting NIST P-256

If the eUICC supports O\_E\_NIST, the ECASD contains at least:

 The eUICC’s Private Key #SK\_EUICC\_ECDSA (for creating ECDSA signatures) based on NIST P-256 [11]

 The eUICC’s Certificate #CERT\_EUICC\_ECDSA (for eUICC authentication) containing the eUICC’s Public Key #PK\_EUICC\_ECDSA based on NIST P-256 [11]

 The GSMA Certificate Issuer’s Public Key #PK\_CI\_ECDSA (for verifying off-card entities certificates) based on NIST P-256 [11]

 The Certificate of the EUM #CERT\_EUM\_ECDSA based on NIST P-256 [11]

Other Certificates and Keys MAY be present.

G.1.3 For eUICC supporting BrainpoolP256r1

If the eUICC supports O\_E\_BRP, the ECASD contains at least:

 The eUICC’s Private Key #SK\_EUICC\_ECDSA (for creating ECDSA signatures) based on brainpoolP256r1 [8]

 The eUICC’s Certificate #CERT\_EUICC\_ECDSA (for eUICC authentication) containing the eUICC’s Public Key #PK\_EUICC\_ECDSA based on brainpoolP256r1 [8]

 The GSMA Certificate Issuer’s Public Key #PK\_CI\_ECDSA (for verifying off-card entities certificates) based on brainpoolP256r1 [8]

 The Certificate of the EUM #CERT\_EUM\_ECDSA based on brainpoolP256r1 [8]

 Other Certificates and Keys MAY be present.

G.1.4 With default RAT configuration

The eUICC’s RAT is configured as detailed in SGP.21 Annex H:

 Only one PPAR authorizing PPR1 and PPR2 for all MNOs with End User consent required i.e. #PPRS\_ALLOWED

G.1.5 With Additional PPARs in the RAT

The eUICC’s RAT is configured as below (following this order):

 Additional PPARs representing custom agreements between MNOs and OEMs:

o #PPR1\_WITH\_OWNER\_GID

o #PPR1\_WITHOUT\_GID

o #PPR2\_WITHOUT\_CONSENT

 The last PPAR authorizes PPR1 and PPR2 for all MNOs with End User consent required i.e. #PPRS\_ALLOWED

G.1.6 Clean-up procedure

Unless differently specified in the test case, the following procedure SHALL be executed after each test sequence to bring the eUICC back to its Common Initial State:

* eUICC Memory Reset to delete all profiles and reset the SM-DP+ Address
* Retrieve and Remove all pending notifications

Where necessary, in addition to the above, other steps may be executed to restore the initial state specified in this Annex.

Annex J Integrated eUICC Testing (Normative)

**J.1 Overview (Informative)**

An Integrated eUICC hardware resides in an SoC along with other subsystems such as general processing and mobile broadband modem, all connected through a proprietary SoC interconnect channel. Alternatively, an Integrated eUICC may communicate with a mobile broadband modem external to the SoC via an external interface, which may be proprietary or based on a standard not associated with UICC. As such, Integrated eUICC may not include a physical UICC-Terminal interface [5].

In order to test the functionality and compliancy of an Integrated eUICC, hardware and OS, Integrated eUICC manufacturers need to provide and support a test interface to which testing equipment can be connected to. Having a standardized testing interface, will increase interoperable and reusability between different manufacturer of Integrated eUICC and test equipment.

For Integrated eUICC with a USB CCID [29] test interface, this annex describes its properties. In cases where a USB interface is not available in a device containing an Integrated eUICC, an adapter to USB CCID needs to be provided, e.g. Bluetooth to USB CCID. The functionality needed to provide and support the test interface, shall be considered part of the test environment and not the IUT.



**Figure 1 Integrated eUICC with USB CCID [29] Test Interface**

Note: The mechanism providing USB CCID to the RSP eUICC Test System, and described in this Annex, is implementation specific. As such, it may be implemented in the SoC, on-Device, off-Device or any combination thereof.

**J.2 Integrated eUICC test requirements**

An Integrated eUICC manufacturer shall provide a USB CCID test interface implementing the functionality specified in J.3.

The test interface shall maintain the integrity and order of the data between the Integrated eUICC and the test system.

The Integrated eUICC manufacturer shall ensure that during testing no other clients or SoC subsystems interfere with the testing.

The Integrated eUICC may use any physical or logical interface between the Integrated eUICC and the test system, as long as a USB CCID is provided to the test system and the channel is reliable (i.e. maintain integrity and order).

**J.3 USB CCID test interface**

The Integrated eUICC USB CCID test interface shall operate in a card reader mode.

The Integrated eUICC USB CCID test interface shall support the following [29] section 6 messages:

* [29] section 6.1 Messages:
  + PC\_to\_RDR\_IccPowerOn
  + PC\_to\_RDR\_IccPowerOff
  + PC\_to\_RDR\_GetSlotStatus
  + PC\_to\_RDR\_Escape
  + PC\_to\_RDR\_XfrBlock
  + PC\_to\_RDR\_T0APDU
  + PC\_to\_RDR\_Secure
  + PC\_to\_RDR\_Abort
* [29] section 6.2 Messages:
  + RDR\_to\_PC\_SlotStatus
  + RDR\_to\_PC\_Escape
  + RDR\_to\_PC\_DataBlock

Note: For test systems using wincard.h/PCSC lite APIs to connect to the Integrated eUICC USB CCID test interface, the following APIs are expected to be used:

* SCardEstablishContext
* SCardListReaders[A|W]
* SCardConnect[A|W]
* SCardControl
* SCardTransmit
* SCardDisconnect
* SCardStatus[A|W]
* SCardReleaseContext
* SCardReconnect
* SCardBeginTransaction
* SCardEndTransaction
* SCardGetStatusChange
* SCardFreeMemory
* SCardGetAttrib

Annex K Document Management

J.1 Document History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Version | Date | CR Number | Brief Description of Change | Approval Authority | Editor / Company |
| SGP.33-1 v1.0 | 26 January 2024 | NA | Initial draft of SGP.33-1 | ISAG | Yolanda Sanz, GSMA |
| CR0002R00 | UpdateEim test cases |
| CR0003R00 | DeleteEim test cases |
| CR0004R00 | ListEim test cases |
| CR0005R00 | AddEim test cases |
| CR0006R01 | Adding test cases to cover Enable function. |
| CR0007R01 | Adding test cases to cover Disable function. |
| CR0008R02 | Adding test cases to cover Delete function. |
| CR0009R01 | Adding test cases to cover Disable Error Cases. |
| CR0010R01 | Adding test cases to cover Delete Error Cases. |
| CR0011R01 | Adding test cases to cover EnableError Cases. |
| CR0001R02 | Constants and Methods for eUICC Package |
| CR00012R01 | AddInitialEim test cases |
| CR00013R00 | GetEimConfigData test cases |
| CR00014R01 | ES10b\_ProfileRollback |
| CR0016R01 | Aligning AddEim test cases with PSMO test cases |
| CR000017R01 | Aligning UpdateEim test cases with PSMO test cases |
| CR000018R01 | Aligning DeleteEim test cases with PSMO test cases |
| CR000019R01 | Aligning ListEim test cases with PSMO test cases |
| CR00020R03 | LoadEUICCPackage test cases |
| CR00021R02 | GetCerts test cases |
| CR0022R01 | ESep\_ListProfileInfo |
| CR0023R01 | ESep\_GetRat |
| CR0024R01 | ES10\_ConfigureAutomaticProfileEnabling |
| CR0025R01 | ESep\_Enable\_ErrorCasesScenario5 |
| CR0026R01 | ES10bEnablingUsingDD |
| NA | Remove editor’s note  Delete Annex D commands and reponses table with empty values  Fix the version of the SGP.xxx specifications according to the latest version published.  Remove section 5 test cases as they are for further study. |
| CR0027 | ApplicabilityTableForES10x |
| CR0028 | IPAe testing ES9+ and ES11 |
| NA | Mark eSIM – eUICC Test Caes Mandatory  Mark eSIM – eIM Test Cases Mandatory  Integreted eUICC Test cases – voidd and mark for FFS  Section 5 Procedures deleted and mark as FFS.  Adjust the Reference sections based on the versions agreed during eSIMWG3.101Bis  Reference the test cases in SGP.23 for the eUICC memory reset nominal and error cases |

J.2 Other Information

|  |  |
| --- | --- |
| Type | Description |
| Document Owner | Yolanda Sanz, GSMA |
| Editor / Company | Yolanda Sanz, GSMA |

It is our intention to provide a quality product for your use. If you find any errors or omissions, please contact us with your comments. You may notify us at [prd@gsma.com](mailto:prd@gsma.com).

Your comments or suggestions & questions are always welcome.