IoT Security Applet Interface Test Specification

Version 1.0

27 June 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

Disclaimer

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

Compliance Notice

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

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

Table of Contents

1 Introduction

1.1 Overview

This document defines the test specification for the IoT Security Applet Interface as defined in [1].

1.2 Scope

The elements within the scope are described in the section 2.4.

1.3 Document Purpose

This specification has the objective of testing if the provided IoT SAFE application is correctly interpreting and executing commands sent according to the [1].

This document is agnostic upon the Secure Element platform on which the IoT SAFE application is loaded under condition that it is respecting system requirements described within this specification and in [1]. Below test cases are created in a way to remove any context side effects that could impacted the IoT SAFE application functions.

1.4 Intended Audience

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

1.5 Definition of Terms

Term	Description
(e)UICC	A removable or non-removable UICC which enables the remote and/or local management of Profiles in a secure way. NOTE: The term originates from "embedded UICC"
UICC	A secure element platform specified in ETSI TS 102 221 [2]

1.6 Abbreviations

Abbreviation	Description
AID	Application Identifier
APDU	Application Protocol Data Unit
CLA	Class
DH	Diffie-Hellman
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral
ECDSA	Elliptic Curve Digital Signature Algorithm
ECKA	Elliptic Curve Key Agreement
EF	Elementary File

GSMA	GSM Association
HKDF	Hash-based Key Derivation Function
HMAC	Hash-based Message Authentication Code
ID	Identity
INS	Instruction
IoT	Internet of Things
ISD	Issuer Security Domain
ISD-R	Issuer Security Domain Root
IUT	Implementation Under Test
NIST	National Institute for Science and Technology
PKCS	Public-Key Cryptography Standards
PRF	Pseudorandom Function
PSK	Pre-Shared Key
RSA	Rivest / Shamir / Adleman
SCP	Secure Channel Protocol
SHA	Secure Hash Algorithm
T	Test Tool
TLS	Transport Layer Security

1.7 References

Ref	Document Number	Title
[1]	IoT.05	GSMA IoT Security Applet Interface Description
[2]	ETSI TS 102 221	Smart Cards; UICC-Terminal interface; Physical and logical characteristics
[3]	RFC 2119	Key words for use in RFCs to Indicate Requirement Levels, S. Bradner
[4]	RFC 8174	Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words

1.8 Conventions

“The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALLNOT”, “SHOULD ”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119 [3] and clarified by RFC 8174 [4], when, and only when, they appear in all capitals, as shown here.”

2 Test Environment

2.1 Table of optional features

The supplier of the implementation SHALL state the support of possible options in Table 1.

Item	Option	Support	Mnemonic
1	Support of Applet Type 1 (See Note1)		O_TYPE_1
2	Support of Key Derivation Cryptographic Function (see Note2 and Note4)		O_KEY_DER
3	Support of Key derivation PRF SHA-256 (see Note2 and Note3)		O_PRF
4	Support of Hash Algorithm SHA-384		O_SHA384
5	Support of Hash Algorithm SHA-512		O_SHA512
6	Support of Applet Type 2 (See Note1)		O_TYPE_2
Note1: The implementation SHALL support either Item 1, or Item 6. Note2: O_PRF and O_KEY_DER are mandatory if O_TYPE_2 is supported Note3: O_PRF is applicable only if O_KEY_DER is supported Note4: If the O_KEY_DER is supported then the HMAC KEYS and Compute HKDF SHALL be supported.			

Table 1 Options

2.2 Applicability table

2.2.1 Applet type 1

The table bellow lists the test cases applicable for Applet Type 1.

Test case	Test case title	GSMA Version 1.0	Applet Store Configuration
4.1.1.1	Compute DH – Test Case 1	M	DEF_NIST DEF_BRP
4.1.1.2	Compute DH – Test Case 2	M	DEF_NIST DEF_BRP
4.1.1.3	Compute DH – Test Case 3	M	DEF_NIST DEF_BRP
4.1.1.4	Compute DH – Test Case 4	M	DEF_NIST DEF_BRP
4.1.1.5	Compute DH – Test Case 5	M	DEF_NIST
4.1.1.6	Compute DH – Test Case 6	M	DEF_NIST
4.1.1.7	Compute DH – Test Case 7	M	DEF_NIST
4.1.1.8	Compute DH – Test Case 8	M	DEF_NIST DEF_BRP
4.1.1.9	Compute DH – Test Case 9	M	DEF_NIST
4.1.1.10	Compute DH – Test Case 10	M	DEF_COMB
4.1.1.11	Compute DH – Test Case 11	M	DEF_NIST
4.1.1.12	Compute DH – Test Case 12	M	DEF_NIST DEF_BRP
4.1.1.13	Compute DH – Test Case 13	M	DEF_NIST DEF_BRP
4.1.1.14	Compute DH – Test Case 14	M	DEF_NIST DEF_BRP
4.1.1.15	Compute DH – Test Case 15	M	DEF_NIST DEF_BRP
4.1.2.1	Compute HKDF – Test Case 1	C004	DEF_SEC_KEY

Test case	Test case title	GSMA Version 1.0	Applet Store Configuration
4.1.2.2	Compute HKDF – Test Case 2	C002	DEF_SEC_KEY
4.1.2.3	Compute HKDF – Test Case 3	C003	DEF_SEC_KEY
4.1.2.4	Compute HKDF – Test Case 4	C004	DEF_SEC_KEY
4.1.2.5	Compute HKDF – Test Case 5	C002	DEF_SEC_KEY
4.1.2.6	Compute HKDF – Test Case 6	C003	DEF_SEC_KEY
4.1.2.7	Compute HKDF – Test Case 7	C004	DEF_SEC_KEY
4.1.2.8	Compute HKDF – Test Case 8	C004	DEF_SEC_KEY
4.1.2.9	Compute HKDF – Test Case 9	C004	DEF_SEC_KEY
4.1.2.10	Compute HKDF – Test Case 10	C004	DEF_SEC_KEY
4.1.2.11	Compute HKDF – Test Case 11	C004	DEF_SEC_KEY
4.1.2.12	Compute HKDF – Test Case 12	C004	DEF_SEC_KEY
4.1.3.1	Compute PRF – Test Case 1	C001	DEF_SEC_KEY
4.1.3.2	Compute PRF – Test Case 2	C001	DEF_SEC_KEY
4.1.3.3	Compute PRF – Test Case 3	C001	DEF_SEC_KEY
4.1.3.4	Compute PRF – Test Case 4	C001	DEF_SEC_KEY
4.1.3.5	Compute PRF – Test Case 5	C001	DEF_SEC_KEY
4.1.3.6	Compute PRF – Test Case 6	C001	DEF_SEC_KEY
4.1.3.7	Compute PRF – Test Case 7	C001	DEF_SEC_KEY
4.1.3.8	Compute PFR – Test Case 8	C001	DEF_SEC_KEY
4.1.3.9	Compute PRF – Test Case 9	C001	DEF_SEC_KEY
4.1.3.10	Compute PRF – Test Case 10	C001	DEF_SEC_KEY
4.1.3.11	Compute PRF – Test Case 11	C001	DEF_SEC_KEY
4.1.3.12	Compute PRF – Test Case 12	C001	DEF_SEC_KEY
4.1.4.1	Compute Signature – Test Case 1	M	DEF_NIST DEF_BRP
4.1.4.2	Compute Signature – Test Case 2	M	DEF_NIST DEF_BRP

Test case	Test case title	GSMA Version 1.0	Applet Store Configuration
4.1.4.3	Compute Signature – Test Case 3	M	DEF_NIST DEF_BRP
4.1.4.4	Compute Signature – Test Case 4	M	DEF_NIST DEF_BRP
4.1.4.5	Compute Signature – Test Case 5	M	DEF_NIST DEF_BRP
4.1.4.6	Compute Signature – Test Case 6	M	DEF_NIST DEF_BRP
4.1.4.7	Compute Signature – Test Case 7	M	DEF_NIST DEF_BRP DEF_COMB
4.1.4.8	Compute Signature – Test Case 8	M	DEF_NIST DEF_BRP DEF_COMB
4.1.4.9	Compute Signature – Test Case 9	M	DEF_NIST DEF_BRP
4.1.4.10	Compute Signature – Test Case 10	M	DEF_NIST DEF_BRP
4.1.4.11	Compute Signature – Test Case 11	M	DEF_NIST DEF_BRP
4.1.4.12	Compute Signature – Test Case 12	M	DEF_NIST DEF_BRP
4.1.4.13	Compute Signature – Test Case 13	M	DEF_NIST DEF_BRP
4.1.5.1	Generate Key Pair – Test Case 1	M	DEF_NIST DEF_BRP DEF_COMB
4.1.5.2	Generate Key Pair – Test Case 2	M	DEF_NIST DEF_BRP DEF_COMB
4.1.5.3	Generate Key Pair – Test Case 3	M	DEF_NIST
4.1.5.4	Generate Key Pair – Test Case 4	M	DEF_NIST
4.1.5.5	Generate Key Pair – Test Case 5	M	DEF_NIST
4.1.5.6	Generate Key Pair – Test Case 6	M	DEF_NIST
4.1.5.7	Generate Key Pair – Test Case 7	M	DEF_NIST
4.1.5.8	Generate Key Pair – Test Case 8	M	DEF_NIST
4.1.5.9	Generate Key Pair – Test Case 9	M	DEF_NIST
4.1.5.10	Generate Key Pair – Test Case 10	M	DEF_NIST

Test case	Test case title	GSMA Version 1.0	Applet Store Configuration
4.1.6.1	Get Data-Application – Test Case 1	M	DEF_NIST
4.1.6.2	Get Data- Application – Test Case 2	M	DEF_NIST
4.1.6.3	Get Data- Application – Test Case 3	M	DEF_NIST
4.1.6.4	Get Data- Application – Test Case 4	M	DEF_NIST
4.1.7.1	Get data – file Test Case 1	M	DEF_NIST
4.1.7.2	Get data – file Test Case 2	M	DEF_NIST
4.1.7.3	Get data – file Test Case 3	M	DEF_NIST
4.1.7.4	Get data – file Test Case 4	M	DEF_NIST
4.1.7.5	Get data – file Test Case 5	M	DEF_NIST
4.1.8.1	Get data – object list Test Case 1	M	DEF_NIST DEF_BRP DEF_COMB DEF_SEC_KEY
4.1.8.2	Get data – object list Test Case 2	M	DEF_NIST
4.1.8.3	Get data – object list Test Case 3	M	DEF_NIST
4.1.8.4	Get data – object list Test Case 4	M	DEF_NIST
4.1.9.1	Get Data-Private Key Information – Test Case 1	M	DEF_NIST DEF_BRP DEF_COMB
4.1.9.2	Get Data-Private Key Information – Test Case 2	M	DEF_NIST DEF_BRP DEF_COMB
4.1.9.3	Get Data-Private Key Information – Test Case 3	M	DEF_NIST
4.1.9.4	Get Data-Private Key Information – Test Case 4	M	DEF_NIST
4.1.9.5	Get Data-Private Key Information – Test Case 5	M	DEF_NIST
4.1.9.6	Get Data-Private Key Information – Test Case 6	M	DEF_NIST
4.1.10.1	Get data – Public key Information - Test Case 1	M	DEF_NIST DEF_BRP DEF_COMB
4.1.10.2	Get data – Public key Information - Test Case 2	M	DEF_NIST DEF_BRP DEF_COMB
4.1.10.3	Get data – Public key Information -Test Case 3	M	DEF_NIST

Test case	Test case title	GSMA Version 1.0	Applet Store Configuration
4.1.10.4	Get data – Public key Information - Test Case 4	M	DEF_NIST
4.1.10.5	Get data – Public key Information -Test Case 5	M	DEF_NIST
4.1.10.6	Get data – Public key Information - Test Case 6	M	DEF_NIST
4.1.11.1	Get data secret key information – Test Case 1	M	DEF_SEC_KEY
4.1.11.2	Get data secret key information – Test Case 2	M	DEF_SEC_KEY
4.1.11.3	Get data secret key information – Test Case 3	M	DEF_SEC_KEY
4.1.11.4	Get data secret key information – Test Case 4	M	DEF_SEC_KEY
4.1.11.5	Get data secret key information – Test Case 5	M	DEF_SEC_KEY
4.1.12.1	Get Random - Test Case 1	M	DEF_NIST
4.1.12.2	Get Random - Test Case 2	M	DEF_NIST
4.1.12.3	Get Random - Test Case 3	M	DEF_NIST
4.1.12.4	Get Random - Test Case 4	M	DEF_NIST
4.1.13.1	Put Public Key - Test Case 1	M	DEF_NIST DEF_BRP DEF_COMB
4.1.13.2	Put Public Key - Test Case 2	M	DEF_NIST DEF_BRP DEF_COMB
4.1.13.3	Put Public Key - Test Case 3	M	DEF_NIST DEF_BRP
4.1.13.4	Put Public Key - Test Case 4	M	DEF_NIST DEF_BRP
4.1.13.5	Put Public Key - Test Case 5	M	DEF_NIST DEF_BRP
4.1.13.6	Put Public Key - Test Case 6	M	DEF_NIST DEF_BRP
4.1.13.7	Put Public Key - Test Case 7	M	DEF_NIST DEF_BRP
4.1.13.8	Put Public Key - Test Case 8	M	DEF_NIST DEF_BRP
4.1.13.9	Put Public Key - Test Case 9	M	DEF_NIST DEF_BRP
4.1.13.10	Put Public Key - Test Case 10	M	DEF_NIST DEF_BRP
4.1.13.11	Put Public Key - Test Case 11	M	DEF_NIST DEF_BRP
4.1.13.12	Put Public Key - Test Case 12	M	DEF_NIST DEF_BRP
4.1.13.13	Put Public Key - Test Case 13	M	DEF_NIST DEF_BRP
4.1.14.1	Read File - Test Case 1	M	DEF_NIST

Test case	Test case title	GSMA Version 1.0	Applet Store Configuration
			DEF_BRP
4.1.14.2	Read File - Test Case 2	M	DEF_NIST DEF_BRP
4.1.14.3	Read File - Test Case 3	M	DEF_NIST DEF_BRP
4.1.14.4	Read File - Test Case 4	M	DEF_NIST DEF_BRP
4.1.14.5	Read File - Test Case 5	M	DEF_NIST DEF_BRP
4.1.14.6	Read File - Test Case 6	M	DEF_NIST DEF_BRP
4.1.15.1	Read Public Key - Test Case 1	M	DEF_NIST DEF_BRP DEF_COMB
4.1.15.2	Read Public Key - Test Case 2	M	DEF_NIST DEF_BRP
4.1.15.3	Read Public Key - Test Case 3	M	DEF_NIST DEF_BRP
4.1.15.4	Read Public Key - Test Case 4	M	DEF_NIST DEF_BRP
4.1.15.5	Read Public Key - Test Case 5	M	DEF_NIST DEF_BRP
4.1.15.6	Read Public Key - Test Case 6	M	DEF_NIST DEF_BRP
4.1.15.7	Read Public Key - Test Case 7	M	DEF_NIST DEF_BRP
4.1.15.8	Read Public Key - Test Case 8	M	DEF_NIST DEF_BRP
4.1.16.1	Verify Signature Test Case 1	M	DEF_NIST DEF_BRP
4.1.16.2	Verify Signature Test Case 2	M	DEF_NIST DEF_BRP
4.1.16.3	Verify Signature Test Case 3	M	DEF_NIST DEF_BRP
4.1.16.4	Verify Signature Test Case 4	M	DEF_NIST DEF_BRP
4.1.16.5	Verify Signature Test Case 5	M	DEF_NIST DEF_BRP
4.1.16.6	Verify Signature Test Case 6	M	DEF_NIST DEF_BRP
4.1.16.7	Verify Signature Test Case 7	M	DEF_NIST DEF_BRP
4.1.16.8	Verify Signature Test Case 8	M	DEF_NIST
4.1.16.9	Verify Signature Test Case 9	M	DEF_NIST
4.1.16.10	Verify Signature Test Case 10	M	DEF_NIST DEF_BRP
4.1.16.11	Verify Signature Test Case 11	M	DEF_NIST DEF_BRP
4.1.16.12	Verify Signature Test Case 12	M	DEF_NIST
4.1.16.13	Verify Signature Test Case 13	M	DEF_NIST DEF_BRP

Table 2: Applicability of tests for Applet type 1

2.2.2 Applet type 1

The table bellow lists the test cases applicable for Applet Type 2.

Test case	Test case title	GSMA Version 1.0	Applet Store Configuration
4.2.1.1	Compute HKDF – Test Case 1	M	DEF_SEC_KEY
4.2.1.2	Compute HKDF – Test Case 2	C005	DEF_SEC_KEY
4.2.1.3	Compute HKDF – Test Case 3	C006	DEF_SEC_KEY
4.2.1.4	Compute HKDF – Test Case 4	M	DEF_SEC_KEY
4.2.1.5	Compute HKDF – Test Case 5	C005	DEF_SEC_KEY
4.2.1.6	Compute HKDF – Test Case 6	C006	DEF_SEC_KEY
4.2.1.7	Compute HKDF – Test Case 7	M	DEF_SEC_KEY
4.2.1.8	Compute HKDF – Test Case 8	M	DEF_SEC_KEY
4.2.1.9	Compute HKDF – Test Case 9	M	DEF_SEC_KEY
4.2.1.10	Compute HKDF – Test Case 10	M	DEF_SEC_KEY
4.2.1.11	Compute HKDF – Test Case 11	M	DEF_SEC_KEY
4.2.2.1	Compute PRF – Test Case 1	M	DEF_SEC_KEY
4.2.2.2	Compute PRF – Test Case 2	M	DEF_SEC_KEY
4.2.2.3	Compute PRF – Test Case 3	M	DEF_SEC_KEY
4.2.2.4	Compute PRF – Test Case 4	M	DEF_SEC_KEY
4.2.2.5	Compute PRF – Test Case 5	M	DEF_SEC_KEY

Test case	Test case title	GSMA Version 1.0	Applet Store Configuration
4.2.2.6	Compute PRF – Test Case 6	M	DEF_SEC_KEY
4.2.2.7	Compute PRF – Test Case 7	M	DEF_SEC_KEY
4.2.2.8	Compute PRF – Test Case 8	M	DEF_SEC_KEY
4.2.2.9	Compute PRF – Test Case 9	M	DEF_SEC_KEY
4.2.2.10	Compute PRF – Test Case 10	M	DEF_SEC_KEY
4.2.2.11	Compute PRF – Test Case 11	M	DEF_SEC_KEY
4.2.2.12	Compute PRF – Test Case 12	M	DEF_SEC_KEY
4.2.3.1	Get data Application – Test Case 1	M	DEF_SEC_KEY
4.2.3.2	Get data Application – Test Case 2	M	DEF_SEC_KEY
4.2.3.3	Get data Application – Test Case 3	M	DEF_SEC_KEY
4.2.3.4	Get data Application – Test Case 4	M	DEF_SEC_KEY
4.2.3.5	Get data Application – Test Case 5	M	DEF_SEC_KEY
4.2.4.1	Get data File – Test Case 1	M	DEF_SEC_KEY
4.2.4.2	Get data File – Test Case 2	M	DEF_SEC_KEY
4.2.4.3	Get data File – Test Case 3	M	DEF_SEC_KEY
4.2.4.4	Get data File – Test Case 4	M	DEF_SEC_KEY

Test case	Test case title	GSMA Version 1.0	Applet Store Configuration
4.2.4.5	Get data File – Test Case 5	M	DEF_SEC_KEY
4.2.5.1	Get data – object list Test Case 1	M	DEF_SEC_KEY
4.2.5.2	Get data – object list Test Case 2	M	DEF_SEC_KEY
4.2.5.3	Get data – object list Test Case 3	M	DEF_SEC_KEY
4.2.5.4	Get data – object list Test Case 4	M	DEF_SEC_KEY
4.2.6.1	Get data secret key information – Test Case 1	M	DEF_SEC_KEY
4.2.6.2	Get data secret key information – Test Case 2	M	DEF_SEC_KEY
4.2.6.3	Get data secret key information – Test Case 3	M	DEF_SEC_KEY
4.2.6.4	Get data secret key information – Test Case 4	M	DEF_SEC_KEY
4.2.6.5	Get data secret key information – Test Case 5	M	DEF_SEC_KEY
4.2.7.1	Get Random - Test Case 1	M	DEF_SEC_KEY
4.2.7.2	Get Random - Test Case 2	M	DEF_SEC_KEY
4.2.7.3	Get Random - Test Case 3	M	DEF_SEC_KEY
4.2.7.4	Get Random - Test Case 4	M	DEF_SEC_KEY
4.2.8.1	Read File - Test Case 1	M	DEF_SEC_KEY
4.2.8.2	Read File - Test Case 2	M	DEF_SEC_KEY
4.2.8.3	Read File - Test Case 3	M	DEF_SEC_KEY
4.2.8.4	Read File - Test Case 4	M	DEF_SEC_KEY
4.2.8.5	Read File - Test Case 5	M	DEF_SEC_KEY
4.2.8.6	Read File - Test Case 6	M	DEF_SEC_KEY

Table 1: Applicability of tests for Applet type 2

Conditional item	Condition
C001	IF O_PRF SUPPORTED THEN M ELSE N/A
C002	IF O_SHA384 SUPPORTED AND 0_KEY_DER SUPPORTED THEN M ELSE N/A
C003	IF O_SHA512 SUPPORTED AND 0_KEY_DER SUPPORTED THEN M ELSE N/A
C004	IF O_KEY_DER SUPPORTED THEN M ELSE N/A
C005	IF O_SHA384 SUPPORTED THEN M ELSE N/A
C006	IF O_SHA512 SUPPORTED THEN M ELSE N/A

Table 2: Conditional items referenced by Table 2 and Table 3

2.3 Optional features and applicability tables formatting

2.3.1 Format of the table of optional features

The columns in Table 1 have the following meaning.

Column	Meaning
Option:	The optional feature supported or not by the implementation.
Support:	The support columns are to be filled in by the supplier of the implementation. The following common notations are used for the support column in table 1. • Y or y supported by the implementation; • N or n not supported by the implementation; • N/A, or n/a - no answer required (allowed only if the status is N/A, directly or after evaluation of a conditional status).
Mnemonic:	The mnemonic column contains mnemonic identifiers for each item.

2.3.2 Format of the applicability table

The applicability of every test in Table 2 and Table 3 Table 1is formally expressed by the use of Boolean expressions defined in the following section 2.3.3.

The columns in Table 2 and Table 3 have the following meaning:

Column	Meaning
Test case:	The “Test case” column gives a reference to the test case number(s) detailed in the present document.
Test case title:	The “Test case title” column gives the title of the test case.
GSMA Version X:	The “Version X” column indicates which test cases are applicable for the given Technical Specification version. Several different status notifications can be used in this column. They are defined in section 2.3.3.
Applet Store Configuration	The “Applet Store Configuration” column indicates with which configurations the test case SHALL be ran. It is mandated to run the test case with all the configurations indicated. The configurations are described in section 2.8.

2.3.3 Status and Notations

The “GSMA Version X” columns show the status of the entries as follows:
The following notations are used for the status column:

M mandatory – the capability is required to be supported.

O optional – the capability may be supported or not.

N/A not applicable – in the given context, it is impossible to use the capability.

Ci conditional – the requirement on the capability (“M”, “O”, “X” or “N/A”) depends on the support of other optional or conditional items. For nested conditional expressions, the syntax “IF .. THEN (IF .. THEN .. ELSE..) ELSE ..” is to be used to avoid ambiguities.

2.4 Test environment description

The general architecture for the test environment is:

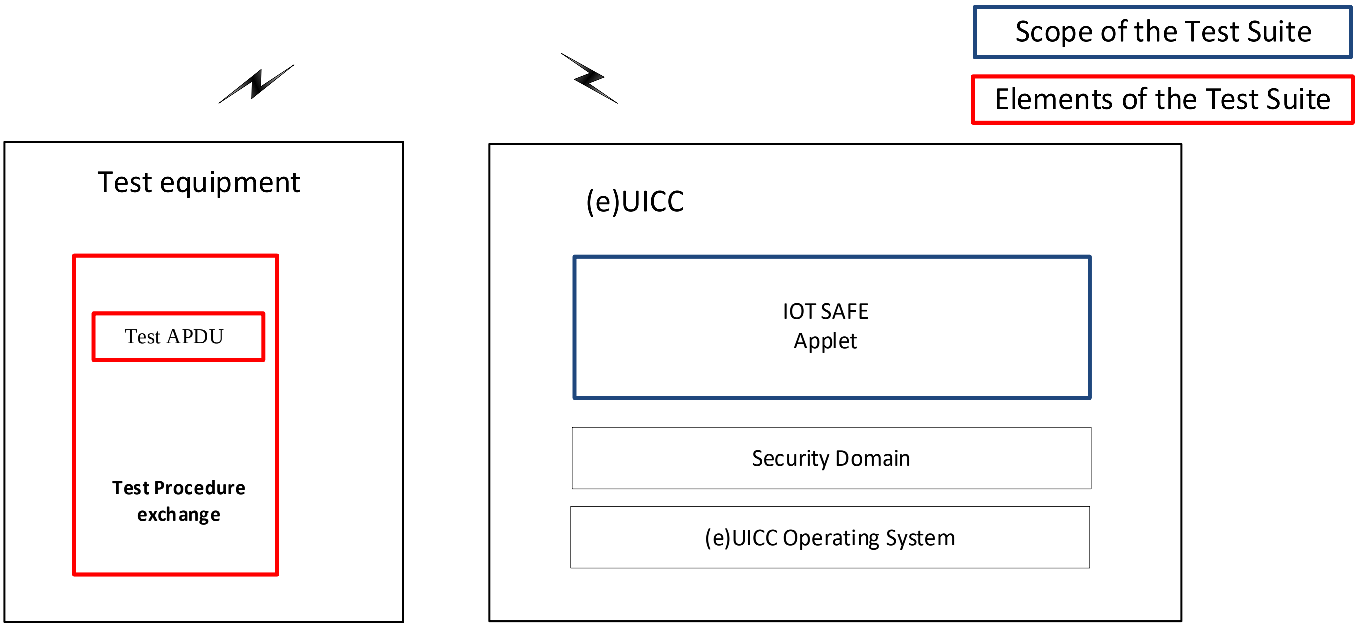

Figure 1: Test environment description

2.5 Test equipment

The test equipment SHALL meet the following requirements:

- It SHALL be able to provide results of the tests.
- It SHALL be able to accept all valid status codes returned.
- It SHALL be able to send and receive commands to/from the (e)UICC on the ISO 7816 interface.
- It SHALL be able to handle the installation and personalization of the IOT SAFE Applet based on the .ldr script of the TCA Loader.

2.6 Test execution

2.6.1 General Initial Conditions

The General Initial Conditions are a set of general prerequisites for the IUT prior to the execution of testing. For each test procedure described in the present document, the following rules apply to the Initial Conditions:

- Unless otherwise stated, the (e)UICC SHALL be reset before each test procedure.
- Unless otherwise stated, the IOT SAFE Applet SHALL be installed and personalized before each test procedure. The IOT SAFE Applet vendor SHALL provide scripts for the installation and the personalization of the IOT SAFE Applet in .ldr format of the TCA Loader. The following Secure Channel Protocols MAY be used in these scripts: SCP03, SCP80, SCP81.

2.6.2 General Post Conditions

For each test procedure described in the present document, the following rules apply to the Post Conditions:

- After each test procedure the IOT SAFE applet instance SHALL be deleted.

2.7 Pass criterion

A test SHALL be considered successful, only if the test procedure was carried out successfully respecting all conformance requirements referenced in the test procedure.

2.8 Applet Type 1 configurations

2.8.1 Default configuration

- SIM Alliance version: **01**
- Applet proprietary identifier: **47534d4120496f542053414645204170706c657420547970652031FFFFFFFFFF** (hex format) – ‘GSMA IoT SAFE Applet Type **1**’ **-** filled with FFs to reach the expected 32 bytes.
- Max number of files: **05**
- Max number of private keys: **05**
- Max number of public keys: **05**
- Max number of secret keys: **05**
- Cryptographic functions: **0F (Signature / Key generation / Key agreement / key derivation)**
- Supported algorithms for hash: **01 (SHA-256)**
- Supported algorithms for signature: **04 (ECDSA)**
- Supported algorithms for key agreement: **01 (ECKA)**
- Supported algorithms for key derivation: **02 (HKDF)**
- Maximum number of sessions: **03**

  Note: For the scope of this document the Key generation is mandatory.

2.8.1.1 Applet Store

2.8.1.1.1 Applet Store Default Configuration NIST keys (DEF_NIST)

The key pair defined below with label as ‘GSMA EPH CL Key Pair LB 01’ is a NIST p256r1 Ephemeral’s (Volatile) to be used for Key Generation (Generate Key Pair command), Key Agreement (Compute DH command) and Get Data related test cases.

Public Key :
Label: ‘GSMA EPH CL Key Pair LB 01’
ID: **‘GSMA EPH CL PKID 01’**
Object Access Condition: **Read**
Object State: **deactivated**
Key Type: NIST p256r1 (volatile)
Key Specific Usage: **general purpose**
Cryptographic functions: **Key generation | Key agreement**
Algorithm for key agreement: ECKA

Private Key :
Label: ‘GSMA EPH CL Key Pair LB 01’
ID: **‘GSMA EPH CL SKID 01’**
Object Access Condition: None
Object State: deactivated
Key Type: NIST p256r1 (volatile)
Key Specific Usage: **general purpose**
Cryptographic functions: **Key generation | Key agreement**
Algorithm for key agreement: ECKA

The key pair defined below with label as ‘GSMA KPLB 01’ is a NIST p256r1 Persistent to be used in ComputeDH, Generate Key Pair, Put Public Key, Verify Signature and Get Data related test cases.

Private Key :
Label: **‘GSMA KPLB 01’**
ID: **‘GSMA SKID 01’**
Object Access Condition: **None**
Object State: **activated**
Key Type: **NIST p256r1 (persistent)**
Key Specific Usage: **General purpose file**
Cryptographic functions: **Key agreement | Signature**
Algorithm for key agreement: **ECKA**
Key Value: **‘880df809bc68fe11ddfb2676bed8a102cbf89f1c8d67480e668525b643f5b551’**

Public Key :
Label: **‘GSMA KPLB 01’**
ID: ‘GSMA PKID 01’
Object Access Condition: **Read**
Object State: **activated**
Key Type: **NIST p256r1 (persistent)**
Key Specific Usage: **General purpose file**

Cryptographic functions: **Key agreement | Signature**
Algorithm for key agreement: **ECKA**
Key Value:
[‘271ab56b70cd46f65b6617e4f3b78d7d6f39da0ed2575675da50f4c6db47a95’, ‘e3b99f1a471da2255fca287586466826add3f20f7724cf5047d8d604147f2ec7’]

The Public key defined below with ID as ‘GSMA PKID 02’ is a NIST p256r1 Persistent to be used in Compute DH and Get Data related test cases.

Public Key :
Label: **‘GSMA KPLB 02’**
ID: **‘GSMA PKID 02’**
Object Access Condition: **Read**
Object State: **activated**
Key Type: **NIST p256r1 (persistent)**
Key Specific Usage: **General purpose file**
Cryptographic functions: **Key agreement**
Algorithm for key agreement: **ECKA**
Key Value:
[‘4950d9f3ac76bc1dd214d61f254059b44bd464e3e6bdc58f9cd4b1921a59af3d’, ‘e926907fb27c1f31cbcb0294b3d047b0feacd1c5b2695c5e11fad45704598ddf’]

The Private key defined below with ID ‘GSMA CL CRED SKID 02’ Persistent to be used in Compute DH, Compute Signature and Get Data commands. It can also be used for a negative testcase of Generate Key Pair command since it does not have key generation cryptographic function granted.

Private Key:
Label: ‘**GSMA CL CRED SKLB 02’**
ID: **‘GSMA CL CRED SKID 02’**
Object Access Condition: **None**
Object State: **activated**
Key Type: NIST p256r1 (persistent)
Key Specific Usage: **TLS-handshake Certificate Verify message**
Cryptographic functions: **Signature**
Algorithm for Signature: ECDSA
Algorithm for Hash: SHA256
Value: **‘f2267e6475a69a057deb80a6d4f541fa0a1828724577d4c4a55f471dfb04adb8’**

The file defined below with label ‘GSMA CL CRED FLB 01’ SHALL contain the Public Key associated to the Private Key above (with label “GSMA CL CRED SKLB 02”). This can be used in Read File, Get Data – File and Get Data – object list related test cases.

File 1:
Label: **‘GSMA CL CRED FLB 01’**
ID: **‘GSMA CL CRED FID 01’**
Object Access Condition: **Read**
Object State: **activated**
File Specific Usage: **X509v3 certificate**
Value: **['dac1eafad2909577cbc83b78d07f205d2c61d17fa56a08cc92d447da011d44ab', '16242f4ff655589d5f9c890614b2df7781a52985381f68557eed893a672a9bdb']**

The Public Key defined below with ID ‘GSMA SV EPH PKID 02’ is a NIST p256r1 Ephemeral (Volatile) to be used in Get data – Object List, Put Public Key, Read Public Key and Get Data related test cases.

Public Key:
Label: **‘GSMA SV EPH PKLB 02’**
ID: **‘GSMA SV EPH PKID 02’**
Object Access Condition: **Update**
Object State: **deactivated**
Key Type: **NIST p256r1 (volatile)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Signature**
Algorithm for Signature: **ECDSA**
Algorithm for Hash: **SHA256**

The Public key defined below with ID ‘GSMA ISSUER PKID 03’ is a NIST p256r1 Persistent to be used in Read Public Key, Verify Signature and Put Public Key related test cases.

Public Key:
Label: **‘GSMA ISSUER PKLB 03’**
ID: **‘GSMA ISSUER PKID 03’**
Object Access Condition: **Read**
Object State: **activated**
Key Type: **NIST p256r1 (persistent)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Signature**
Algorithm for Signature: **ECDSA**
Algorithm for Hash: **SHA256**
Value:
['052fe46694427e98e41418b52c99dd17b2bb6f625b4e46fd2815ff62f1d2f20c', '096e3f4c170d0e107a7c5a67db6401dc5b96a68f10a627637ac3655f44106ccd']

The file defined below with label ‘GSMA ISSUER CERT FLB 02’ SHALL contain the Public Key defined above (with label “GSMA ISSUER PKLB 03”). This can be used for Read File and Get Data related test cases.

File 2:
Label: **‘GSMA ISSUER CERT FLB 02’**
ID: **‘GSMA ISSUER CERT FID 02’**
Object Access Condition: **Read**
Object State: **activated**
File Specific Usage: **X509v3 certificate**
Value:
['052fe46694427e98e41418b52c99dd17b2bb6f625b4e46fd2815ff62f1d2f20c', '096e3f4c170d0e107a7c5a67db6401dc5b96a68f10a627637ac3655f44106ccd']

The File defined below with ID ‘GSMA FID 01_01’ does not have ‘Read’ as Object Access Condition. To be used in Read File negative testcase. It can also be used in Get Data related commands.

File 1_a:
Label: **‘GSMA FLB 01_01’**
ID: **‘GSMA FID 01_01’**
Object Access Condition: **NONE**
Object State: **activated**
File Specific Usage: **X509v3 certificate**

The File defined below with ID ‘GSMA FID 01_02’ is ‘Deactivated’ as Object State. To be used in Read File negative testcase. It can also be used in Get Data related commands.

File 1_b:
Label: **‘GSMA FLB 01_02’**
ID: **‘GSMA FID 01_02’**
Object Access Condition: **Read**
Object State: **deactivated**
File Specific Usage: **X509v3 certificate**

2.8.1.1.2 Applet Store Default Configuration Keys for Negative test (DEF_NIST_NEG)

Public Key :
Label: **‘GSMA EPH CL Key Pair LB 03’**
ID: **‘GSMA EPH CL PKID 03’**
Object Access Condition: **None**
Object State: **deactivated**
Key Type: **NIST p256r1 (volatile)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Key generation | Key agreement**
Algorithm for key agreement: **ECKA**

Private Key :
Label: **‘GSMA EPH CL Key Pair LB 03’**
ID: **‘GSMA EPH CL SKID 03’**
Object Access Condition: **None**
Object State: **deactivated**
Key Type: **NIST p256r1 (volatile)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Key generation | Key agreement**
Algorithm for key agreement: **ECKA**

The Public key defined below with ID ‘GSMA SERVER PKLB 04’ is a NIST p256r1 Persistent to be used in Read Public Key negative testcase, since it does not have ‘Read’ as Object Access Condition. It can also be used for Get Data related test cases.

Public Key :
Label: **‘GSMA SERVER PKLB 04’**
ID: **‘GSMA SERVER PKID 04’**

Object Access Condition: **none**
Object State: **activated**
Key Type: **NIST p256r1 (persistent)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Signature**
Algorithm for Signature: **ECDSA**
Algorithm for Hash: **SHA256**

2.8.1.1.3 Applet Store Default Configuration Brainpool keys (DEF_BRP)

The Key Pair defined below with Label ‘GSMA EPH CL Key Pair LB 01’ is a Brainpool p256r1 Persistent to be used in Compute Signature, Get Data – Private Key Information, Get Data – Public Key Information, Compute DH, Generate Key Pair, Put Public Key and Verify Signature test cases.

Public Key :
Label: **‘GSMA EPH CL Key Pair LB 01’**
ID: **‘GSMA EPH CL PKID 01’**
Object Access Condition: **Read**
Object State: **deactivated**
Key Type: **Brainpool p256r1 (volatile)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Key generation | Key agreement**
Algorithm for key agreement: **ECKA**

Private Key :
Label: **‘GSMA EPH CL Key Pair LB 01’**
ID: **‘GSMA EPH CL SKID 01’**
Object Access Condition: **None**
Object State: **deactivated**
Key Type: **Brainpool p256r1 (volatile)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Key generation | Key agreement**
Algorithm for key agreement: **ECKA**

The Private Key defined below with ID ‘GSMA CL CRED SKID 02’ is Brainpool p256r1 Persistent to be used in Compute Signature, Compute DH Generate Key Pair and Get Data related test cases.

Private Key :
Label: **‘GSMA CL CRED SKLB 02’**
ID: **‘GSMA CL CRED SKID 02’**
Object Access Condition: **None**
Object State: **activated**
Key Type: **Brainpool p256r1 (persistent)**
Key Specific Usage: **TLS-handshake Certificate Verify message (recommendation is TLS v1.2)**

Cryptographic functions: **Signature**
Algorithm for Signature: **ECDSA**
Algorithm for Hash: **SHA256**

The file defined below with label ‘GSMA CL CRED FLB 01’ SHALL contain the Public Key associated to the Private Key above (with label “GSMA CL CRED SKLB 02”). This can be used in Read File, Get Data – File and Get Data – object list related test cases.

File 1:
Label: **‘GSMA CL CRED FLB 01’**
ID: **‘GSMA CL CRED FID 01’**
Object Access Condition: **Read**
Object State: **activated**
File Specific Usage: **X509v3 certificate**

The Public key defined below with ID ‘GSMA SV EPH PKID 02’ is Brainpoolp256r1 Ephemeral (Volatile) to be used in Get Data – Object List, Put Public Key and Read Public Key test cases.

Public Key :
Label: **‘GSMA SV EPH PKLB 02’**
ID: **‘GSMA SV EPH PKID 02’**
Object Access Condition: **Update**
Object State: **deactivated**
Key Type: **Brainpool p256r1 (volatile)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Signature**
Algorithm for Signature: **ECDSA**
Algorithm for Hash: **SHA256**

The Public Key defined below with ID ‘GSMA ISSUER PKID 03’ is Brainpool p256r1 Persistent to be used in Put Public Key, Read Public Key, Verify Signature and Get Data related test cases.

Public Key :
Label: **‘GSMA ISSUER PKLB 03’**
ID: **‘GSMA ISSUER PKID 03’**
Object Access Condition: **READ**
Object State: **activated**
Key Type: **Brainpool p256r1 (persistent)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Signature**
Algorithm for Signature: **ECDSA**
Algorithm for Hash: **SHA256**

The file defined below with label ‘GSMA ISSUER CERT FLB 02’ SHALL contain the Public Key defined above (with label “GSMA ISSUER PKLB 03”). This can be used for Read File and Get Data related test cases.

File 2:
Label: **‘GSMA ISSUER CERT FLB 02’**
ID: **‘GSMA ISSUER CERT FID 02’**
Object Access Condition: **Read**
Object State: **activated**
File Specific Usage: **X509v3 certificate**

The Public Key defined below with ID ‘GSMA PKID 02’ is Brainpool p256r1 Persistent to be used in Compute DH and Get Data related test cases.

Public Key :
Label: **‘GSMA KPLB 02’**
ID: ‘**GSMA PKID 02’**
Object Access Condition: **Read**
Object State: **activated**
Key Type: **Brainpoolp256r1 (persistent)**
Key Specific Usage: **General purpose file**
Cryptographic functions: **Key agreement**
Algorithm for key agreement: **ECKA**
Key Value:
[‘8D2D688C6CF93E1160AD04CC4429117DC2C41825E1E9FCA0ADDD34E6F1B39F7B’,
‘990C57520812BE512641E47034832106BC7D3E8DD0E4C7F1136D7006547CEC6A’]

The Key Pair defined below with label ‘GSMA KPLB 01’ is Brainpool p256r1 Persistent to be used in Compute DH, Generate Key Pair, Verify Signature and Get Data related commands.

Private Key :
Label: **‘GSMA KPLB 01’**
ID: **‘GSMA SKID 01’**
Object Access Condition: **None**
Object State: **activated**
Key Type: **Brainpoolp256r1 (persistent)**
Key Specific Usage: **General purpose file**
Cryptographic functions: **Key agreement | Signature**
Algorithm for key agreement: **ECKA**
Key Value:
‘1A8ED90F8DBC48CA8BAB9201C4B7F52B63357A97830437ADD360867E58C1A079’

Public Key :
Label: **‘GSMA KPLB 01’**
ID: **‘GSMA PKID 01’**
Object Access Condition: **Read**
Object State: **activated**
Key Type: **Brainpoolp256r1 (persistent)**
Key Specific Usage: **General purpose file**
Cryptographic functions: **Key agreement | Signature**
Algorithm for key agreement: **ECKA**
Key Value:
[’045643DB0FC7E6B4AEEC3496466BBBD3FB6EC028034C8DBFFB863F401527D676’,
’3A71C31C96DEC5CF8F2306255474A5658F9234F1EA6278F1A159D9542F5424D0DB]

This file defined bellow with label ‘GSMA FLB 01’ SHALL contain the Public Key defined above (with label “GSMA KPLB 01”).

File 1: (Note: It SHALL contain the above Private Key)
Label: **‘GSMA FLB 01’**
ID: **‘GSMA FID 01’**
Object Access Condition: **Read**
Object State: **activated**
File Specific Usage: **X509v3 certificate**

The Public Key defined below with ID ‘GSMA SERVER PKID 04’ is Brainpool p256r1 Persistent to be used in Read Public Key and Get Data related test cases.

Public Key :
Label: **‘GSMA SERVER PKLB 04’**
ID: **‘GSMA SERVER PKID 04’**
Object Access Condition: **none**
Object State: **activated**
Key Type: **Brainpool p256r1 (persistent)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Signature**
Algorithm for Signature: **ECDSA**
Algorithm for Hash: **SHA256**
Value:
['052fe46694427e98e41418b52c99dd17b2bb6f625b4e46fd2815ff62f1d2f20c', '096e3f4c170d0e107a7c5a67db6401dc5b96a68f10a627637ac3655f44106ccd']

This file SHALL contain the Public Key defined above (with label “GSMA SERVER PKLB 04”).

File 2: (Note: It SHALL contain the above Public Key – Server public key)
Label: **‘GSMA SERVER FLB 01’**
ID: **‘GSMA SERVER FID 01’**
Object Access Condition: **Read**
Object State: **activated**
File Specific Usage: **X509v3 certificate**

2.8.1.1.4 Applet Store Default Configuration Combined keys [NIST and Brainpool] (DEF_COMB)

The key pair defined below with label as ‘GSMA EPH CL Key Pair LB 01’ is a NIST p256r1 Ephemeral’s (Volatile) to be used for Key Generation (Generate Key Pair command), Key Agreement (Compute DH command) and Get Data related test cases.

Public Key :
Label: **‘GSMA EPH CL Key Pair LB 01’**
ID: **‘GSMA EPH CL PKID 01’**
Object Access Condition: **Read**
Object State: **deactivated**
Key Type: **NIST p256r1 (volatile)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Key generation | Key agreement**
Algorithm for key agreement: **ECKA**

Private Key:

Label: **‘GSMA EPH CL Key Pair LB 01’**
ID: **‘GSMA EPH CL SKID 01’**
Object Access Condition: **None**
Object State: **deactivated**
Key Type: **NIST p256r1 (volatile)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Key generation | Key agreement**
Algorithm for key agreement: **ECKA**

The key pair defined below with label as ‘GSMA EPH CL Key Pair LB 02’ is a Brainpool p256r1 Ephemeral’s (Volatile) to be used for Key Generation (Generate Key Pair command), Key Agreement (Compute DH command) and Get Data related test cases.
Note: This key is not used in this version of the specification.

Private Key :
Label: **‘GSMA EPH CL Key Pair LB 02’**
ID: **‘GSMA EPH CL SKID 02’**
Object Access Condition: **None**
Object State: **activated**
Key Type: **Brainpool p256r1 (volatile)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Key generation | Key agreement**
Algorithm for key agreement: **ECKA**

Public Key :
Label: **‘GSMA EPH CL Key Pair LB 02’**
ID: **‘GSMA EPH CL PKID 02’**
Object Access Condition: **Read**
Object State: **deactivated**
Key Type: **Brainpool p256r1 (volatile)**
Key Specific Usage: **general purpose**
Cryptographic functions: **Key generation | Key agreement**
Algorithm for key agreement: **ECKA**

The file defined below with ID ‘GSMA CL CRED FLB 01’ can be used for Read File and Get Data related test cases.

File 1:– Client Public Key)
Label: **‘GSMA CL CRED FLB 01’**
ID: **‘GSMA CL CRED FID 01’**
Object Access Condition: **Read**
Object State: **activated**
File Specific Usage: **X509v3 certificate**
Value: **X509_CERT** (Located in the Annex – Under Data Values used in Test Cases)

2.8.1.1.5 Applet Store Default Configuration secret keys (DEF_SEC_KEY)

- Secret Key :
  Label: **‘GSMA SECRET KEY LB 01’**
  ID: **‘GSMA SEC KEY ID 01’**

Object Access Condition: **none**
Object State: **activated**
Key Type: **A0** (HMAC capable key)
Cryptographic functions: **Key derivation**
Supported algorithms for key derivation:**03** (PRF SHA-256,HKDF)(NOTE 1)
Value: **5345435245544b455931**

- Secret Key :
  ID: **'GSMA SEC KEY ID 02'**
  Object Access Condition: **none**
  Object State: **activated**
  Key Type: **A0** (HMAC capable key)
  Cryptographic functions: **Key derivation**
  Supported algorithms for key derivation: **02** (HKDF)
  Value: **5345435245544b455932**

Secret Key :
Label: **'GSMA SECRET KEY LB 03'**
ID: **'GSMA SEC KEY ID 03'**
Object Access Condition: **none**
Object State: **activated**
Key Type: **A0** (HMAC capable key)
Cryptographic functions: **Key derivation**
Supported algorithms for key derivation:**01** (PRF SHA-256)
Value:
404142434445464748494A4B4C4D4E4F404142434445464748494A4B4C4D4E4F

NOTE 1: In case of the PRF SHA-256 (rfc5246) is not supported the Supported algorithms for key derivation value SHALL be set to 0x02

2.8.2 Optional configuration

2.8.2.1.1 Applet Configuration Algorithm For Hash - sha384 (suffix: _+SHA384)

The configuration defined in the section 2.8.1 SHALL apply, with the exception listed below:
Supported algorithms for hash: **03 (SHA-256, SHA-384)**

2.8.2.1.2 Applet Configuration Algorithm For Hash - sha512 (suffix: _+SHA384-512)

The configuration defined in the section 2.8.1 SHALL apply, with the exception listed below:
Supported algorithms for hash: **07 (SHA-256, SHA-384,SHA-512)**

2.8.2.1.3 PRF configuration

The configuration defined in the section 2.8.1 SHALL apply, with the exception listed below: Supported algorithms for key derivation: **03 (HKDF/PRF)**

2.9 Applet Type 2 configurations

2.9.1 Default configuration

- SIM Alliance version: **01**
- Applet proprietary identifier: **47534d4120496f54205341464520417070706c65742054797065203 2FFFFFFFFF** – 'GSMA IoT SAFE Applet Type **2**' **-** filled with FFs up to 32 bytes.
- Max number of files: **05**
- Max number of secret keys: **05**
- Cryptographic functions: **08 (Key derivation)**
- Supported algorithms for hash: **01 (SHA-256)**
- Supported algorithms for key derivation: **03 (HKDF / PRF)**

2.9.1.1 Applet Store Default Configuration secret keys (DEF_SEC_KEY)

Secret Key:
Label: **'GSMA SECRET KEY LB 01'**
ID: **'GSMA SEC KEY ID 01'**
Object Access Condition: **none**
Object State: **activated**
Key Type: **A0** (HMAC capable key)
Cryptographic functions: **Key derivation**
Supported algorithms for key derivation: **02** (HKDF)
Value: **5345435245544b455931**

Secret Key:
ID: **'GSMA SEC KEY ID 02'**
Object Access Condition: **none**
Object State: **activated**
Key Type: **A0** (HMAC capable key)
Cryptographic functions: **Key derivation**
Supported algorithms for key derivation: **02** (HKDF)
Value: **5345435245544b455932**

Secret Key:
Label: **'GSMA SECRET KEY LB 03'**
ID: **'GSMA SEC KEY ID 03'**
Object Access Condition: **none**
Object State: **activated**
Key Type: **A0** (HMAC capable key)
Cryptographic functions: **Key derivation**
Supported algorithms for key derivation:**01** (PRF SHA-256)
Value:
404142434445464748494A4B4C4D4E4F404142434445464748494A4B4C4D4E4F

2.9.2 Optional configuration

2.9.2.1 Applet Store Default Configuration secret keys - sha384

The configuration defined in the section 2.9.1 SHALL apply, with the exception listed below:
Supported algorithms for hash: 03 (SHA-256, SHA-384)

2.9.2.2 Applet Store Default Configuration secret keys - sha512

The configuration defined in the section 2.9.1 SHALL apply, with the exception listed below:
Supported algorithms for hash: 07 (SHA-256, SHA-384,SHA-512)

2.10 Information to be provided by the Applet Vendor

The Applet Vendor SHALL provide information:

Item	Description	Value
1	Maximum number of sessions allowed	

3 Conformance Requirements

3.1 Conformance requirements for Applet Type 1

Req ID	Requirement description
RQ.2.6.1	The Compute DH command is used to generate a shared secret from a public key and a private key.
RQ.2.6.2	The command executes the Diffie-Hellman key agreement scheme with both keys and provides the caller with the shared secret.
RQ.2.6.3	Both public and private keys must be activated
RQ.2.6.4	Both public and private keys must come from a different key pair.
RQ.2.6.5	Both public and private keys must have the same key type
RQ.2.6.6	Both public and private keys must be granted with a key agreement cryptographic function and with same ECKA key agreement algorithm.
RQ.2.6.7	Command header and data field for Compute DH as defined in sections 2.6.3.1 and 2.6.3.2 of [1].
RQ.2.7.1	The Compute HKDF command is used to generate key material based on the HMAC-based key derivation function.
RQ.2.7.2	HKDF-Extract general mode: the command is used to generate a pseudo random key. The input key material (IKM) comes as an input in command data.
RQ.2.7.3	HKDF-Extract PSK-based: the command is used to generate a pseudo random key. The input key material (the secret) comes from a secret key of the applet store.
RQ.2.7.4	The hash function to use with the HKDF algorithm is indicated in input data
RQ.2.7.5	The length of the salt must match with the length of the digest of the hash algorithm
RQ.2.7.6	Processing state returned in response message for Compute HKDF as reported in section 2.7.4.2 of [1]
RQ.2.7.7	Data Field in the command message for Compute HKDF as reported in section 2.7.3.2 of [1]
RQ.2.7.8	Data Field returned in response message for Compute HKDF as reported in section 2.7.4.1 of [1]
RQ.2.8.1	The Compute PRF command generates pseudo-random numbers based on the PRF function
RQ.2.8.2	General mode: the secret comes as an input data in command data.
RQ.2.8.3	PSK-plain pre-master secret mode: the applet builds a pre-master secret in conformance with rfc4279, chapter 2, and uses it as the secret in the PRF computation. The PSK used for that pre-master secret computation comes from a secret key of the applet store.
RQ.2.8.4	PSK-ECDHE pre-master secret mode: the applet builds a pre-master secret in conformance with rfc5489, chapter 2, and uses it as the secret in the PRF computation. The PSK used for that pre-master secret computation comes from a secret key of the applet store and the ECDH computation result comes as input in command data
RQ.2.8.5	The length of the salt must match with the length of the digest of the hash algorithm
RQ.2.8.6	Processing state returned in response message for Compute PRF as reported in section 2.8.4.2 of [1]
RQ.2.8.7	Data Field in the command message for Compute PRF as reported in section 2.8.3.2 of [1]
RQ.2.8.8	Data Field returned in response message for Compute PRF as reported in section 2.8.4.1 of [1]
RQ.2.9.1	The Compute Signature – Init command opens a session to compute a signature. The command can also be used to cancel a signature computation session.
RQ.2.9.2	Full text processing: the full text is hashed by the applet before padding and signature computation.
RQ.2.9.3	Last block processing: the last block of the text is hashed by the applet before padding and signature computation.
RQ.2.9.4	Pad and sign processing: the hash on the full text is computed externally then transferred to the applet for padding and comparison with the value in the reference signature.
RQ.2.9.5	The session number to open must be closed and the maximum number of sessions must not be reached (request for session opening).
RQ.2.9.6	The private key must exist, be activated and must be granted for signature with requested hash and signature algorithms (request for session opening).
RQ.2.9.7	The session to close must be opened and related to a Compute Signature operation (request for session cancellation)
RQ.2.9.8	Command header and data field for Compute Signature – Init as defined in sections 2.9.3.1 and 2.9.3.2 of [1].
RQ.2.9.9	Processing state returned in response message for Compute Signature – Init as defined in section 2.9.4.1 of [1].
RQ.2.10.1	The Compute Signature – Update command is used to provide the applet with reference data and return a signature to the caller.

RQ.2.10.2	First APDU with more incoming data are indicated with P1.b8=0 & P1.b0=0. The last APDU with last incoming data is indicated with P1.b8=1 & P1.b0. This APDU returns output data and a 6300SW whenever further output data can be fetched.
RQ.2.10.3	The session must be opened.
RQ.2.10.4	Command header and data field for Compute Signature – Update as defined in sections 2.10.3.1 and 2.10.3.2 of [1].
RQ.2.10.5	Data field returned in response message for Compute Signature - Update as defined in section 2.10.4.1 of [1].
RQ.2.10.6	Processing state returned in response message for Compute Signature – Update as defined in section 2.10.4.2 of [1].
RQ.2.11.1	The Generate key pair command is used to generate public/private key pair.
RQ.2.11.2	Processing state returned in response message for Generate key pair as reported in section 2.11.4.2 of [1]
RQ.2.11.3	Data Field in the command message for Generate key pair as reported in section 2.11.3.2 of [1]
RQ.2.11.4	Data Field returned in response message for Generate key pair as reported in section 2.11.4.1 of [1]
RQ.2.12.1	The Get Data - Application command lists information about the applet and it's capacity
RQ. 2.12.2	Processing state returned in response message for Get Data - Application as reported in section 2.12.4.2 of [1]
RQ.2.12.3	Command Header in the command message for Get Data - Application as reported in section 2.12.3.1 of [1]
RQ. 2.12.4	Data Field returned in response message for Get Data - Application as reported in section 2.12.4.1 of [1]
RQ.2.13.1	Get Data - File command retrieves all file information from the applet store
RQ.2.13.2	Processing state returned in response message for Get Data - File as reported in section 2.13.4.2 of [1]
RQ.2.13.3	Data Field in the command message for Get Data - File as reported in section 2.13.3.2 of [1]
RQ.2.13.4	Data Field returned in response message for Get Data - File as reported in section 2.13.4.1 of [1]
RQ.2.14.1	The Get Data – Object List command list all objects (files, key pairs, secret keys) and their attributes present in the applet store.
RQ.2.14.2	Objects are listed one after the other. They are grouped together in the same response message. A response message only contains the complete object information structures.
RQ.2.14.3	The 'data outgoing mode' indicates where the first objects SHALL be fetched or where the next objects SHALL be fetched. The presence of objects to retrieve is indicated with the SW6300h. From that SW fetching subsequent objects requires the next command to be provided with the exact same CLA, INS and P1 bytes but P2.
RQ.2.14.4	Command header for Get Data – Object List as defined in section 2.14.3.1 of [1].
RQ.2.14.5	Data field returned in the message for Get Data – Object List as defined in section 2.14.4.1 of [1].
RQ.2.14.6	Processing state returned in response message for Get Data – Object List as defined in section 2.14.4.6 of [1].
RQ.2.15.1	The Get Data – Private Key command lists Private key Information of specified Private key.
RQ.2.15.2	Processing state returned in response message for Get Data – Private Key as reported in section 2.15.4.2 of [1]
RQ.2.15.3	Data Field in the command message for Get Data – Private Key as reported in section 2.15.3.2 of [1]
RQ.2.15.4	Data Field returned in response message for Get Data – Private Key as reported in section 2.15.4.1 of [1]
RQ.2.16.1	The Get Data – Public Key command lists Pubic key Information of specified Public key.
RQ.2.16.2	Processing state returned in response message for Get Data – Public Key as reported in section 2.16.4.2 of [1]
RQ.2.16.3	Data Field in the command message for Get Data – Public Key as reported in section 2.16.3.2 of [1]
RQ.2.16.4	Data Field returned in response message for Get Data – Public Key as reported in section 2.16.4.1 of [1]
RQ.2.17.1	The Get Data – Secret Key command retrieves all information associated to a secret key.

RQ.2.17.2	Processing state returned in response message as reported in section 2.17.4.2 of [1]
RQ.2.17.3	Data Field in the command message for as reported in section 2.17.3.2 of [1]
RQ.2.17.4	Data Field returned in response message as reported in section 2.17.4.1 of [1]
RQ.2.18.1	The Get Random command gets a random number of specified length.
RQ.2.18.2	Processing state returned in response message for Get Data – Private Key as reported in section 2.18.4.2 of [1]
RQ.2.18.3	Data Field in the command message for Get Data – Private Key as reported in section 2.18.3.2 of [1]
RQ.2.18.4	Data Field returned in response message for Get Data – Private Key as reported in section 2.18.4.1 of [1]
RQ.2.19.1	Put public key – init command opens a session to update a public key
RQ.2.19.2	Put public key – init command can be used to cancel a put public key session
RQ.2.19.3	Processing state returned in response message for Put public key – init command as reported in section 2.19.4.1 of [1]
RQ.2.19.4	Data Field in the command message for Put public key – init as reported in section 2.19.3.1 of [1]
RQ.2.20.1	Put public key – update command is used as many times as necessary to load a public key in applet store
RQ.2.20.2	Processing state returned in response message for Put public key – update as reported in section 2.18.4.1 of [1]
RQ.2.20.3	Data Field in the command message for Put public key – update as reported in section 2.20.3.1 of [1]
RQ.2.20.4	Data Field returned in response message for Put public key – update as reported in section 2.20.3.2 of [1]
RQ.2.21.1	Read File command retrieves file data from the applet store
RQ.2.21.2	Processing state returned in response message for Read file as reported in section 2.21.4.1 of [1]
RQ.2.21.3	Data Field in the command message for Read file as reported in section 2.21.3.1 of [1]
RQ.2.21.4	Data Field returned in response message for Read file as reported in section 2.21.3.2 of [1]
RQ.2.22.1	Read public key command retrieves a public key from the applet store
RQ.2.22.2	Processing state returned in response message for Read public key as reported in section 2.22.4.1 of [1]
RQ.2.22.3	Data Field in the command message for Read public key as reported in section 2.22.3.1 of [1]
RQ.2.22.4	Data Field returned in response message for Read public key as reported in section 2.22.3.2 of [1]
RQ.2.23.1	The Verify Signature – Init command opens a session to verify a signature.
RQ.2.23.2	Full text processing: the full text is hashed by the applet before padding and comparison with the value in the reference signature.
RQ.2.23.3	Pan and sign processing: the hash of the full text is computed externally then transferred to the applet for padding and comparison with the value in the reference signature.
RQ.2.23.4	The session number to open must be closed and the maximum number of sessions must not be reached (request for session opening).
RQ.2.23.5	The public key must exist, be activated, and must be granted for signature with requested hash and signature algorithm (request for session opening).
RQ.2.23.6	The session to close must be opened and related to a Verify Signature operation (request for session cancellation).
RQ.2.23.7	Command header and data for Verify Signature – Init as defined in sections 2.23.3.1 and 2.23.3.2 of [1].
RQ.2.23.8	Processing state returned in response message for Verify Signature – Init as defined in section 2.23.4.1 of [1].
RQ.2.24.1	The Verify Signature – Update command is used to provide the applet with reference data to verify a signature and get the comparison result.
RQ.2.24.2	With ‘full text’ mode several APDU might be necessary to provide reference data. In that case it is expected that all but the last data block are filled with 255 bytes of data.
RQ.2.24.3	The session must be opened.
RQ.2.24.4	Command header and data field for Verify Signature – Update as defined in sections 2.24.3.1 and 2.24.3.2 of [1].
RQ.2.24.5	Processing state returned in response message for Verify Signature – Update as defined in section 2.24.4.1 of [1].

3.2 Conformance requirements for Applet Type 2

Req ID	Requirement description
RQ.3.6.1	The Compute HKDF command is used to generate key material based on the HMAC-based key derivation function.
RQ.3.6.2	HKDF-Extract general mode: the command is used to generate a pseudo random key. The input key material (IKM) comes as an input in command data.
RQ.3.6.3	HKDF-Extract PSK-based: the command is used to generate a pseudo random key. The input key material (the secret) comes from a secret key of the applet store.
RQ.3.6.4	The hash function to use with the HKDF algorithm is indicated in input data
RQ.3.6.5	The length of the salt must match with the length of the digest of the hash algorithm
RQ.3.6.6	Processing state returned in response message for Compute HKDF as reported in section 3.6.4.2 of [1]
RQ.3.6.7	Data Field in the command message for Compute HKDF as reported in section 3.6.3.2 of [1]
RQ.3.6.8	Data Field returned in response message for Compute HKDF as reported in section 3.6.4.1 of [1]
RQ.3.7.1	The Compute PRF command generates pseudo-random numbers based on the PRF function
RQ.3.7.2	General mode: the secret comes as an input data in command data.
RQ.3.7.3	PSK-plain pre-master secret mode: the applet builds a pre-master secret in conformance with rfc4279, chapter 2, and uses it as the secret in the PRF computation. The PSK used for that pre-master secret computation comes from a secret key of the applet store.
RQ.3.7.4	PSK-ECDHE pre-master secret mode: the applet builds a pre-master secret in conformance with rfc5489, chapter 2, and uses it as the secret in the PRF computation. The PSK used for that pre-master secret computation comes from a secret key of the applet store and the ECDH computation result comes as input in command data
RQ.3.7.5	The length of the salt must match with the length of the digest of the hash algorithm
RQ.3.7.6	Processing state returned in response message for Compute PRF as reported in section 2.8.4.2 of [1]
RQ.3.7.7	Data Field in the command message for Compute PRF as reported in section 2.8.3.2 of [1]
RQ.3.7.8	Data Field returned in response message for Compute PRF as reported in section 2.8.4.1 of [1]
RQ.3.8.1	The Get Data - Application command lists information about the applet and it's capacity
RQ.3.8.2	Processing state returned in response message for Get Data - Application as reported in section 3.8.4.2 of [1]
RQ.3.8.3	Command Header in the command message for Get Data - Application as reported in section 3.8.3.1 of [1]
RQ. 3.8.4	Data Field returned in response message for Get Data - Application as reported in section 3.8.4.1 of [1]
RQ.3.9.1	Get Data - File command retrieves all file information from the applet store
RQ.3.9.2	Processing state returned in response message for Get Data - File key as reported in section 3.9.4.2 of [1]
RQ.3.9.3	Data Field in the command message for Get Data - File as reported in section 3.9.3.2 of [1]
RQ.3.10.1	The Get Data – Object List command list all objects (files, secret keys) and their attributes present in the applet store.
RQ.3.10.2	Objects are listed one after the other. They are grouped together in the same response message. A response message only contains complete object information structures.
RQ.3.10.3	The 'data outgoing mode' indicates where the first objects SHALL be fetched of where the next objects SHALL be fetched. The presence of objects to retrieve is indicated with the SW6300h. From that SW fetching subsequent objects requires the next command to be provided with exact same CLA, INS and P1 bytes but P2.
RQ.3.10.4	Command header for Get Data – Object List as defined in section 3.10.3.1 of [1].
RQ.3.10.5	Data field returned in the message for Get Data – Object List as defined in section 3.10.4.1 of [1].
RQ.3.10.6	Processing state returned in response message for Get Data – Object List as defined in section 3.10.4.4 of [1].
RQ.3.11.1	The Get Data – Secret Key command retrieves all information associated to a secret key.
RQ.3.11.2	Processing state returned in response message as reported in section 2.17.4.2 of [1]
RQ.3.11.3	Data Field in the command message for as reported in section 2.17.3.2 of [1]
RQ.3.11.4	Data Field returned in response message as reported in section 2.17.4.1 of [1]

RQ.3.12.1	The Get Random command gets a random number of specified length.
RQ.3.12.2	Processing state returned in response message for Get Random as reported in section 3.12.4.2 of [1]
RQ.3.12.3	Data Field in the command message for Get Random as reported in section 3.12.3.2 of [1]
RQ.3.12.4	Data Field returned in response message for Get Random as reported in section 3.12.4.1 of [1]
RQ.3.13.1	The Read File command reads contents of an EF from start offset and returns 256 bytes or less if it reaches to the end.
RQ.3.13.2	Processing state returned in response message for Read File Key as reported in section 3.13.4.2 of [1]
RQ.3.13.3	Data Field in the command message for Read File as reported in section 3.13.3.2 of [1]
RQ.3.13.4	Data Field returned in response message for Read File as reported in section 3.13.4.1 of [1]

4 Test Cases

4.1 Test Cases for Applet Type 1

4.1.1 Compute DH

4.1.1.1 Test Case 1

4.1.1.1.1 Test Case Description

Successful generation of shared secret from a Public Key and a Private Key present in applet store using Key ID. The target Private and Public key are:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and Brainpool Keys section (2.8.1.1.2)
- referenced by Key ID.

4.1.1.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.1.3 Test Procedure IoT SAFE

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key ID = ‘GSMA SKID 01’. • Public Key ID = ‘GSMA PKID 02’.	RQ2.6.1 RQ2.6.2 RQ2.6.3 RQ2.6.4 RQ2.6.5 RQ2.6.6
3	T ← IoT SAFE	Expected response: DATA_1 2923BE84E16CD6AE529049F1F1BBE9EBB3A6DB3C870C3E99245E0D1C06B747D EB3124DC843BB8BA61F035A7D0938251F5DD4CBFC96F5453B130D890A1CDBAE 32209A50EE407836FD124932F69E7D49DCAD4F14F2444066D06BC430B7323BA12 2F622919DE18B1FDAB0CA9902B9729D492C807EC599D5E980B2EAC9CC53BF67 D6 HASH EC0A89849484E8B9A988E10AD448D65DCA39D48CC823404C362E08896D7B3364 DATA_2 2923BE84E16CD6AE529049F1F1BBE9EBB3A6DB3C870C3E99245E0D1C06B747D EB3124DC843BB8BA61F035A7D0938251F5DD4CBFC96F5453B130D890A1CDBAE 32209A50EE407836FD124932F69E7D49DCAD4F14F2444066D06BC430B7323BA12 2F622919DE18B1FDAB0CA9902B9729D492C807EC599D5E980B2EAC9CC53BF67 D6BF14D67E2DDC8E6683EF574961FF698F61CDD11E9D9C167272E61DF0844F4A 7702D7E8392C53CBC9121E33749E0CF4D5D49FD4A4597E35CF3222F4CCCFD390 2D48D38F75E6D91D2AE5C0F72B788187440E5F5000D4618DBE7B0515073B33821 F187092DA6454CEB1853E6915F8466A0496730ED9162F6768D4F74A4AD0576876F A16BB11ADAE248879FE52DB2543E53CF445D3D828CE0BF5C560593D97278A597 62DD0C2C9CD68D4496A792508614014B13B6AA51128C18CD6A90B87978C2FF11 51D9A95C19BE1C07EE9A89AA786C2B554BF9AE7D923D155903828D1D96CA1665 E4EE1309CFED9719FE2A5E20C9BB44765382A4689A982797A7678C263B126DF	RQ2.6.7

		DATA_3 89AFC39D41D3B327814B80940B042590F96556EC91E6AE7939BCE31F3A18BF2B SIGNATURE_1 Generated using KeyPair (GSMA KPLB 01) and DATA_1: NIST-256r1: 6B17D1F2E12C4247F8BCE6E563A440F277037D812DEB33A0F4A13945D898C2963D3FD758AA446847EB1100ABCA4ED4D428C897280763201397B1D0683F28B223 Brainpool-256r1: 3044022008923B8A5504B262AE7B9DE4B6F8E34F68C520CD45C2EC53BF39959437A79BAE022005CF0A3BEB15ED9C78F5E0D4D463CD48FB43A0265495F4640EFFAE4371568C8D SIGNATURE_2 Generated using KeyPair (GSMA KPLB 01) and DATA_2: NIST-256r1: 6B17D1F2E12C4247F8BCE6E563A440F277037D812DEB33A0F4A13945D898C2961095769228E4E84122AF901D56E7F6492D9D4A9FEEAE8D10DF36B72D7FE56645 Brainpool-256r1: 3044022062B45E8FB99E633A08D7D2F8B7FFDF3546DD1751CD6511B472D5552F81187566022009134184B27CA8C490AAC1154ADF65249597B4E7073E055F957BEEF2F2110188 SECRET_DATA_KEY_1 41 SECRET_DATA_KEY_1	

4.1.1.2 Test Case 2

4.1.1.2.1 Test Case Description

Successful generation of shared secret from a Public Key and a Private Key present in applet store using Key Label. The target Private and Public key are:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,

- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and Brainpool Keys section (2.8.1.1.3)
- referenced by Key Label.

4.1.1.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key Label = **‘GSMA KPLB 01’** • Public Key Label = **‘GSMA KPLB 02’**	RQ2.6.1 RQ2.6.2 RQ2.6.3 RQ2.6.4 RQ2.6.5 RQ2.6.6
3	T ← IoT SAFE	Expected response: DATA_2.	RQ2.6.7

4.1.1.3 Test Case 3

4.1.1.3.1 Test Case Description

Successful generation of shared secret from a Public Key and a Private Key present in applet store using Key Label or Key ID combination. The target Private and Public key are:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and Brainpool Keys section (2.8.1.1.3)
- referenced by Key Label.

4.1.1.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key ID = **‘GSMA SKID 01’** • Public Key Label = **‘GSMA KPLB 02’**	RQ2.6.1 RQ2.6.2 RQ2.6.3 RQ2.6.4 RQ2.6.5 RQ2.6.6
3	T ← IoT SAFE	Expected response: DATA_2.	RQ2.6.7

4	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key Label = **'GSMA KPLB 01'** • Public Key ID = **'GSMA PKID 02'**	RQ2.6.1 RQ2.6.2 RQ2.6.3 RQ2.6.4 RQ2.6.5 RQ2.6.6
5	T ← IoT SAFE	Expected response: DATA_2.	RQ2.6.7

4.1.1.4 Test Case 4

4.1.1.4.1 Test Case Description

Successful generation of shared secret from a Public Key and a Private Key using ephemeral keys:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1).
- referenced by Key ID.

4.1.1.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.4.3 Test Procedure IoT SAFE

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate Key Pair command to generate the key with the following information : • Private Key ID = **'GSMA EPH CL SKID 01'.**	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key ID = **'GSMA EPH CL SKID 01'.** • Public Key ID = **'GSMA PKID 01'.**	RQ2.6.1 RQ2.6.2 RQ2.6.3 RQ2.6.4 RQ2.6.5 RQ2.6.6
3	T ← IoT SAFE	Expected response: correct length of the shared secret	RQ2.6.7

4.1.1.5 Test Case 5

4.1.1.5.1 Test Case Description

Failure when Compute DH command is sent with incorrect parameter P1. The target Private and Public key are:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and
- referenced by Key ID.

4.1.1.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 = EEh • P2 = 00h • Private Key ID = **'GSMA SKID 01'.** • Public Key ID = **'GSMA PKID 02'.**	RQ2.6.7
3	T ← IoT SAFE	Expected response: SW6A86h.	RQ2.6.7

4.1.1.6 Test Case 6

4.1.1.6.1 Test Case Description

Failure when Compute DH command is sent with incorrect parameter P2. The target Private and Public key are:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and
- referenced by Key ID.

4.1.1.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 = 00h • P2 = 1Fh • Private Key ID = **'GSMA SKID 01'.** • Public Key ID = **'GSMA PKID 02'.**	RQ2.6.7
3	T ← IoT SAFE	Expected response: SW6A86h.	RQ2.6.7

4.1.1.7 Test Case 7

4.1.1.7.1 Test Case Description

Failure while deactivated private key is used for generation of shared secret. Private and Public key:

- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,

- are according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and
- referenced by Key ID.

4.1.1.7.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.7.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key ID = **'GSMA EPH CL SKID 01'.** • Public Key ID = **'GSMA PKID 02'.**	RQ2.6.3
3	T ← IoT SAFE	Expected response: SW6985h.	RQ2.6.7

4.1.1.8 Test Case 8

4.1.1.8.1 Test Case Description

Failure while deactivated public key is used for generation of shared secret . Private and Public key:

- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and
- The key is referenced by Key ID.

4.1.1.8.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.8.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key ID = **'GSMA CL CRED SKID 02'.** • Public Key ID = **'GSMA EPH CL PKID 01'.**	RQ2.6.3
3	T ← IoT SAFE	Expected response: SW6985h.	RQ2.6.7

4.1.1.9 Test Case 9

4.1.1.9.1 Test Case Description

Failure while Public Key and Private Key belong to same key pair for session key generation. The target Private key and Public Key:

- are activated,

- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm and
- are according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1).
- The key is referenced by Key ID.

4.1.1.9.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.9.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key ID = **'GSMA SKID 01'.** • Public Key ID = **'GSMA PKID 01'.**	RQ2.6.4
3	T ← IoT SAFE	Expected response: SW6985h.	RQ2.6.7

4.1.1.10 Test Case 10

4.1.1.10.1 Test Case Description

Failure when Private and Public keys are of different type, and it is according to the Applet Store Default Configuration Combined Key section 2.8.1.1.3. are used to generate the shared secret. The target Private key and Public Key are:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm and
- referenced by Key ID.

4.1.1.10.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.10.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate Key Pair command to generate the key with the following information : • Private Key ID = **'GSMA EPH CL SKID 01'.**	
3	T → IoT SAFE	Send a Generate Key Pair command to generate the key with the following information : • Private Key ID = **'GSMA EPH CL SKID 02'.**	

2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key ID = **'GSMA EPH CL SKID 01'.** • Public Key ID = **'GSMA EPH CL PKID 02'.**	RQ2.6.5
3	T ← IoT SAFE	Expected response: SW6985h.	RQ2.6.7

4.1.1.11 Test Case 11

4.1.1.11.1 Test Case Description

Failure when Private key don't have Key Agreement usage while Public key have Key Agreement usage. The target Private key and Public Key are:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,
- according to the Applet Store Default Configuration Brainpool Key section (2.8.1.1.1) and
- referenced by Key ID.

4.1.1.11.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.11.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key ID = **'GSMA CL CRED SKID 02'.** • Public Key ID = **'GSMA PKID 02'.**	RQ2.6.6
3	T ← IoT SAFE	Expected response: SW6985h.	RQ2.6.7

4.1.1.12 Test Case 12

4.1.1.12.1 Test Case Description

Failure while generation of shared secret from a Public Key ID not present in app store and a Private Key present in applet store. The target Private key is:

- activated,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and Brainpool Keys section NIST keys (2.8.1.1.2)
- referenced by Key ID.

4.1.1.12.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.12.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key ID = **'GSMA SKID 01'.** • Public Key ID = **'GSMA PKID 08'.**	RQ2.6.7
3	T ← IoT SAFE	Expected response: SW6985h.	RQ2.6.7

4.1.1.13 Test Case 13

4.1.1.13.1 Test Case Description

Failure while generation of shared secret from a Public Key present in app store and a Private Key ID not present in applet store. The target Public Key is:

- activated,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and Brainpool Keys section (2.8.1.1.2)
- referenced by Key ID.

4.1.1.13.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.13.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Private Key ID = **'GSMA SKID 08'.** • Public Key ID = **'GSMA PKID 01'.**	RQ2.6.7
3	T ← IoT SAFE	Expected response: SW6985h.	RQ2.6.7

4.1.1.14 Test Case 14

4.1.1.14.1 Test Case Description

Failure while generation of shared secret with two Private Keys instead of one public key and one private key. Both Private Keys are:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and Brainpool Keys section (2.8.1.1.2)
- referenced by Key ID.

4.1.1.14.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.14.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate Key Pair command to generate the key with the following information : • Private Key ID = **‘GSMA EPH CL SKID 01’.**	
3	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 84: Private Key ID = **‘GSMA SKID 01’** • Tag 74: Private Key Label = **‘GSMA EPH CL SKID 01’**	RQ2.6.7
4	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7

4.1.1.15 Test Case 15

4.1.1.15.1 Test Case Description

Failure while generation of shared secret with two Public Key’s instead of one public and one private key. Both Public Keys are:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and Brainpool Keys section (2.8.1.1.2)
- referenced by Key ID / Key Label

4.1.1.15.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.15.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 85: Public Key ID = **‘GSMA PKID 01’** • Tag 75: Public Key Label = **‘GSMA KPLB 02’**	RQ2.6.7
3	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7

4.1.1.16 Test Case 16

4.1.1.16.1 Test Case Description

Failure while generation of shared secret from a Public Key and a Private Key in applet store. But the Key ID and Key Label tags having additional tag(s). Private and Public Keys are:

- activated,
- belongs to different key pair,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and Brainpool Keys section (2.8.1.1.2)
- referenced by Key ID / Key Label

4.1.1.16.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.16.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 84: Private Key ID = **'GSMA SKID 01'** • Tag 85: Public Key ID = **'GSMA PKID 02'** • Tag 74: Private Key Label = **'GSMA KPLB 02'**	RQ2.6.7
3	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7
4	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 84: Private Key ID = **'GSMA SKID 01'** • Tag 85: Public Key ID = **'GSMA PKID 02'** • Tag 75: Public Key Label = **'GSMA KPLB 02'**	RQ2.6.7
5	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7
6	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 84: Private Key ID = **'GSMA SKID 01'** • Tag 74: Private Key Label = **'GSMA KPLB 01'** • Tag 75: Public Key Label = **'GSMA KPLB 02'**	RQ2.6.7
7	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7
8	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 85: Public Key ID = **'GSMA PKID 02'** • Tag 74: Private Key Label = **'GSMA KPLB 01'** • Tag 75: Public Key Label = **'GSMA KPLB 02'**	RQ2.6.7
9	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7

10	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 84: Private Key ID = **‘GSMA SKID 01’** • Tag 85: Public Key ID = **‘GSMA PKID 02’** • Tag 74: Private Key Label = **‘GSMA KPLB 01’** • Tag 75: Public Key Label = **‘GSMA KPLB 02’**	RQ2.6.7
11	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7

4.1.1.17 Test Case 17

4.1.1.17.1 Test Case Description

Failure while generation of shared secret from a Public Key and a Private Key in applet store. But the Key ID or Key Label tags having same tags multiple times. Private and Public Keys are:

- activated,
- granted with a key agreement cryptographic function,
- granted with ECKA key agreement algorithm,
- according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) and Brainpool Keys section (2.8.1.1.2)
- referenced by Key ID / Key Label

4.1.1.17.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.1.17.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 84: Private Key ID = **‘GSMA SKID 01’** • Tag 85: Public Key ID = **‘GSMA PKID 02’** • Tag 84: Private Key ID = **‘GSMA SKID 01’**	RQ2.6.7
3	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7
4	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 84: Private Key ID = **‘GSMA SKID 01’** • Tag 85: Public Key ID = **‘GSMA PKID 02’** • Tag 85: Public Key ID = **‘GSMA PKID 02’**	RQ2.6.7
5	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7
6	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 74: Private Key Label = **‘GSMA KPLB 01’** • Tag 74: Private Key Label = **‘GSMA KPLB 01’** • Tag 75: Public Key Label = **‘GSMA KPLB 02’**	RQ2.6.7
7	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7

8	T → IoT SAFE	Send a Compute DH command with the following information: • P1 and P2 parameter = 00h • Tag 74: Private Key Label = **'GSMA KPLB 01'** • Tag 75: Public Key Label = **'GSMA KPLB 02'** • Tag 75: Public Key Label = **'GSMA KPLB 02'**	RQ2.6.7
9	T ← IoT SAFE	Expected response: SW6A80h.	RQ2.6.7

-
-

4.1.2 Compute HKDF

4.1.2.1 Test Case 1

4.1.2.1.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract general mode is set, and the Hash algorithm is SHA-256

4.1.2.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x00 (extract general mode) • Secret equal to SECRET_DATA_KEY_16 • Salt equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.2.7.1 RQ.2.7.2 RQ.2.7.4 RQ.2.7.5 RQ.2.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_1	RQ2.7.8

4.1.2.2 Test Case 2

4.1.2.2.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract general mode is set, and the Hash algorithm is SHA-384

4.1.2.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x00 (extract general mode) • Secret equal to SECRET_DATA_KEY_16 • Salt length equal to SALT_DATA_48 • Hash algorithm equal 0x0002 (sha384)	RQ.2.7.1 RQ.2.7.2 RQ.2.7.4 RQ.2.7.5 RQ.2.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_2	RQ2.7.8

4.1.2.3 Test Case 3

4.1.2.3.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract general mode is set, and the Hash algorithm is SHA-512

4.1.2.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x00 (extract general mode) • Secret equal to SECRET_DATA_KEY_16 • Salt length equal to SALT_DATA_64 • Hash algorithm equal 0x0004 (sha512)	RQ.2.7.1 RQ.2.7.2 RQ.2.7.4 RQ.2.7.5 RQ.2.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_3	RQ2.7.8

4.1.2.4 Test Case 4

4.1.2.4.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract PSK-based mode is set, the Hash algorithm is SHA-256 and the search by Secret key ID is used

4.1.2.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x01 (extract PSK-based mode) • Tag 86 as specified in section 2.8.1.1.4 for Key ID = **GSMA SEC KEY ID 01’** • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.2.7.1 RQ.2.7.3 RQ.2.7.4 RQ.2.7.5 RQ.2.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_4	RQ2.7.8

4.1.2.5 Test Case 5

4.1.2.5.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract PSK-based mode is set, the Hash algorithm is SHA-384 and the search by Secret key ID is used

4.1.2.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x01 (extract PSK-based mode) • Tag 86 as specified in section 2.8.1.1.4 for Key ID = **GSMA SEC KEY ID 01** • Salt length equal to SALT_DATA_48 • Hash algorithm equal 0x0002 (sha384)	RQ.2.7.1 RQ.2.7.3 RQ.2.7.4 RQ.2.7.5 RQ.2.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_5	RQ2.7.8

4.1.2.6 Test Case 6

4.1.2.6.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract PSK-based mode is set, the Hash algorithm is SHA-512 and the search by Secret key Label is used

4.1.2.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x01 (extract PSK-based mode) • Tag 76 as specified in section 2.8.1.1.4 for Key ID = **GSMA SEC KEY ID 02** • Salt length equal to SALT_DATA_64 • Hash algorithm equal 0x0004 (sha512)	RQ.2.7.1 RQ.2.7.3 RQ.2.7.4 RQ.2.7.5 RQ.2.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_6	RQ2.7.8

4.1.2.7 Test Case 7

4.1.2.7.1 Test Case Description

Successful execution of Compute HKDF command to generate key material with the key length as small as possible when extract general mode is used.

4.1.2.7.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.7.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x00 (extract general mode) • Secret equal to SECRET_DATA_KEY_1 • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.2.7.1 RQ.2.7.2 RQ.2.7.4 RQ.2.7.5 RQ.2.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_7	RQ2.7.8

4.1.2.8 Test Case 8

4.1.2.8.1 Test Case Description

Compute HKDF command with an incorrect Salt length fails with an error status word.

4.1.2.8.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.8.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x00 (extract general mode) • Secret equal to SECRET_DATA_KEY16 • Salt length equal to 16 (0x10) • Hash algorithm equal 0x0001 (sha256)	RQ.2.7.1 RQ.2.7.3 RQ.2.7.4 RQ.2.7.5 RQ.2.7.7
3	T ← IoT SAFE	Execution failure with SW6A80h	RQ2.7.8

4.1.2.9 Test Case 9

4.1.2.9.1 Test Case Description

Compute HKDF command with an incorrect P1 fails with the defined status word.

4.1.2.9.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.9.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0xFF • Tag 86 as specified in section 2.8.1.1.4 for Key ID = **GSMA SEC KEY ID 01** • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.2.7.1 RQ.2.7.4 RQ.2.7.5 RQ.2.7.7
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ2.7.6

4.1.2.10 Test Case 10

4.1.2.10.1 Test Case Description

Compute HKDF command with an incorrect P2 fails with the defined status word.

.

4.1.2.10.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.10.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P2 set to 0xFF • Tag 86 as specified in section 2.8.1.1.4 for Key ID = **GSMA SEC KEY ID 01** • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.2.7.1 RQ.2.7.4 RQ.2.7.5 RQ.2.7.7

3	T ← IoT SAFE	Execution failure with SW6A86h	RQ2.7.6

4.1.2.11 Test Case 11

4.1.2.11.1 Test Case Description

The Compute HKDF command fails when the key is not granted for HKDF key derivation algorithm.

4.1.2.11.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.11.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x01 (extract PSK-based mode) • Tag 86 as specified in section 2.8.1.1.4 for key with ID = **GSMA SEC KEY ID 03** • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.2.7.1 RQ.2.7.2 RQ.2.7.4 RQ.2.7.7
3	T ← IoT SAFE	Execution failure with SW6985h	RQ2.7.6

4.1.2.12 Test Case 12

4.1.2.12.1 Test Case Description

Check that Compute HKDF command fails when the Label is not present.

4.1.2.12.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.2.12.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x01 (extract PSK-based mode) • Tag 76 with value: 0xFFFFFFFF • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.2.7.1 RQ.2.7.2 RQ.2.7.4 RQ.2.7.7
3	T ← IoT SAFE	Execution failure with SW6985h	RQ2.7.6

4.1.3 Compute PRF

4.1.3.1 Test Case 1

4.1.3.1.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the general mode is set, the secrete length is equal to 16 bytes and the length result value is set to 8.

4.1.3.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.3.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x00 (general mode) • Secret equal to SECRET_DATA_PRF_KEY16 • Label and Seed equal to LABEL_AND_SEED_16 • Pseudo-random length set to 8	RQ.2.8.1 RQ.2.8.2 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_1	RQ.2.8.8

4.1.3.2 Test Case 2

4.1.3.2.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the general mode is set, the secrete length is equal to 20 bytes and the length result value is set to 32.

4.1.3.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.3.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x00 (general mode) • Secret equal to SECRET_DATA_PRF_KEY20 • Label and Seed equal to LABEL_AND_SEED_11 • Pseudo-random length set to 32	RQ.2.8.1 RQ.2.8.2 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_2	RQ.2.8.8

4.1.3.3 Test Case 3

4.1.3.3.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the general mode is set, the secrete length is equal to 64 bytes , the length of concatenation of the label and seed parameters is 123 and the length result value is set to 32.

4.1.3.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.3.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x00 (general mode) • Secret equal to SECRET_DATA_PRF_KEY64 • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.2.8.1 RQ.2.8.2 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_3	RQ.2.8.8

4.1.3.4 Test Case 4

4.1.3.4.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-Plain pre-master secret mode is set, the key is searched by ID, the length of concatenation of the label and seed parameters is 16and the length result value is set to 8.

4.1.3.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.3.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x01 (PSK-Plain mode) • Tag 86 as specified in section 2.8.1.1.4 for key with ID: **'GSMA SEC KEY ID 03'** • Label and Seed equal to LABEL_AND_SEED_16 • Pseudo-random length set to 8	RQ.2.8.1 RQ.2.8.3 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_4	RQ.2.8.8

4.1.3.5 Test Case 5

4.1.3.5.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-Plain pre-master secret mode is set, the key is searched by Label and the length result value is set to 8.

4.1.3.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.3.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x01 (PSK-Plain mode) • Tag 76 as specified in section 2.8.1.1.4 for key with ID: **'GSMA SECRET KEY LB 03'** • Label and Seed equal to LABEL_AND_SEED_16 • Pseudo-random length set to 8	RQ.2.8.1 RQ.2.8.3 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_4	RQ.2.8.8

4.1.3.6 Test Case 6

4.1.3.6.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-Plain pre-master secret mode is set, the key is searched by Label, the length of concatenation of the label and seed parameters is 123 and the length result value is set to 32.

4.1.3.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.3.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x01 (PSK-Plain mode) • Tag 76 as specified in section 2.8.1.1.4 for key with ID: **'GSMA SECRET KEY LB 03'** • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.2.8.1 RQ.2.8.3 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_6	RQ.2.8.8

4.1.3.7 Test Case 7

4.1.3.7.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-ECDHE pre-master secret mode is set, the key is searched by ID, the length of concatenation of the label and seed parameters is 16 and the length result value is set to 8.

4.1.3.7.2 Initial Conditions

Applet installed according 2.8.2.1.3 with Applet Store containing the objects defined at 2.8.1.1.4

4.1.3.7.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x02 (PSK-ECDHE mode) • Tag 86 as specified in section 2.8.1.1.4 for key with ID: **‘GSMA SEC KEY ID 03’** • ECDH result equal to ECDH_COMP_RESULT • Label and Seed equal to LABEL_AND_SEED_16 • Pseudo-random length set to 8	RQ.2.8.1 RQ.2.8.4 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_7	RQ.2.8.8

4.1.3.8 Test Case 8

4.1.3.8.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-ECDHE pre-master secret mode is set, the key is searched by Label, the length of concatenation of the label and seed parameters is 16 and the length result value is set to 8.

4.1.3.8.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.3.8.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x02 (PSK-ECDHE mode) • Tag 76 as specified in section 2.8.1.1.4 for key with ID: **‘GSMA SECRET KEY LB 03’** • ECDH result equal to ECDH_COMP_RESULT • Label and Seed equal to LABEL_AND_SEED_16 • Pseudo-random length set to 8	RQ.2.8.1 RQ.2.8.4 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_7	RQ.2.8.8

4.1.3.9 Test Case 9

4.1.3.9.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-ECDHE pre-master secret mode is set, and the length of concatenation of the label and seed parameters is 123 and the length result value is set to 32.

4.1.3.9.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.3.9.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x02 (PSK-ECDHE mode) • Tag 76 as specified in section 2.8.1.1.4 for key with ID: **'GSMA SECRET KEY LB 03'** • ECDH result equal to ECDH_COMP_RESULT • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.2.8.1 RQ.2.8.4 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_9	RQ.2.8.8

4.1.3.10 Test Case 10

4.1.3.10.1 Test Case Description

Check that Compute PRF command fails when the Label is not present.

.

4.1.3.10.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.3.10.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x01 • Tag 76 with value: 0xFFFFFFFF • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.2.8.1 RQ.2.8.3 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	Execution failure with SW6985h	RQ.2.8.6

4.1.3.11 Test Case 11

4.1.3.11.1 Test Case Description

Check that Compute PRF command fails with incorrect P1.

.

4.1.3.11.2 Initial Conditions

Applet installed according 2.8.2.13 with Applet Store containing the objects defined at 2.8.1.1.4

4.1.3.11.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0xFF • Tag 76 as specified in section 2.8.1.1.4 for key with ID: **'GSMA SECRET KEY LB 03'** • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.2.8.1 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.2.8.6

Test Case 12

4.1.3.11.4 Test Case Description

Check that Compute PRF command fails since the secret key is not granted with PRF SHA-256 key derivation algorithm.

4.1.3.11.5 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.3.11.6 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x01 • Tag 86 as specified in section 2.8.1.1.4 for key with ID: **'GSMA SEC KEY ID 02'** • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.2.8.1 RQ.2.8.3 RQ.2.8.5 RQ.2.8.7
3	T ← IoT SAFE	Execution failure with SW6985h	RQ.2.8.6

4.1.4 Compute Signature

4.1.4.1 Test Case 1

4.1.4.1.1 Test Case Description

Successful signature generation using Full text processing mode without requiring chaining. The target key to be used is the Private Key with Label 'GSMA CL CRED SKLB 02' according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID and the data used for signature is DATA_1 (5.1).

4.1.4.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with session number 1 and the following information: • Private Key ID of the target key. • Mode of operation: Full Text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
4	T → IoT SAFE	Send a Compute Signature – Update command for the session number 1, requesting a signature generation for DATA_1.	RQ.2.10.1 RQ.2.10.2 RQ.2.10.3 RQ.2.10.4
5	T ← IoT SAFE	Expected response: ECDSA signature generated with the correct length and SW9000h (Successful execution).	RQ.2.10.5 RQ.2.10.6

4.1.4.2 Test Case 2

4.1.4.2.1 Test Case Description

Successful signature generation using Full text processing mode requiring chaining. The target key to be used is the Private Key with Label 'GSMA CL CRED SKLB 02' according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by Label and the data used for signature is DATA_2 (5.2).

4.1.4.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with session number 1 and the following information: • Private Key Label of the target key. • Mode of operation: Full Text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8

3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
4	T → IoT SAFE	Send two Compute Signature – Update commands forming the chaining sequence for requesting the signature generation of DATA_2 for session number 1.	RQ.2.10.1 RQ.2.10.2 RQ.2.10.3 RQ.2.10.4
5	T ← IoT SAFE	Expected response: ECDSA signature generated with the correct length and SW9000h (Successful execution).	RQ.2.10.5 RQ.2.10.6

4.1.4.3 Test Case 3

4.1.4.3.1 Test Case Description

- Successful signature generation using Last block processing mode. The target key to be used is the Private Key with Label 'GSMA CL CRED SKLB 02' according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by Label and the data used for signature is DATA_1 (5.1).

4.1.4.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with session number 1 and the following information: • Private Key Label of the target key. • Mode of operation: Last block • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.3 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
4	T → IoT SAFE	Send a Compute Signature – Update command for the session number 1, requesting a signature generation for DATA_1. The Last block to hash, Intermediate hash and Number of bytes already hashed fields are correctly given.	RQ.2.10.1 RQ.2.10.2 RQ.2.10.3 RQ.2.10.4
5	T ← IoT SAFE	Expected response: ECDSA signature generated with the correct length and SW9000h (Successful execution).	RQ.2.10.5 RQ.2.10.6

4.1.4.4 Test Case 4

4.1.4.4.1 Test Case Description

Successful signature generation using Pad and sign processing mode. The target key to be used is the Private Key with Label 'GSMA CL CRED SKLB 02' according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID and the data used for signature is DATA_2 (5.20).

4.1.4.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with session number 1 and the following information: • Private Key ID of the target key. • Mode of operation: Pad and sign • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.4 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
4	T → IoT SAFE	Send a Compute Signature – Update command for the session number 1, requesting a signature generation for DATA_2. The Final hash field is correctly given.	RQ.2.10.1 RQ.2.10.2 RQ.2.10.3 RQ.2.10.4
5	T ← IoT SAFE	Expected response: ECDSA signature generated with the correct length and SW9000h (Successful execution).	RQ.2.10.5 RQ.2.10.6

4.1.4.5 Test Case 5

4.1.4.5.1 Test Case Description

Successful signature generation when wo different Compute Signature sessions are opened and a signature is generated for each one in order to ensure a correct behaviour when having concurrency between sessions:

- For the first session, Full text processing mode is used and the target key to be used is the Private Key with Label ‘GSMA CL CRED SKLB 02’ according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID and the data used for signature is DATA_1 (5.1).
- For the second session, Pad and sign processing mode is used and the target key is the same one used before. The key is referenced by Label and the data used for signature is DATA_1 (5.1).

4.1.4.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening the first session with number 1 and the following information: • Private Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
4	T → IoT SAFE	Send a Compute Signature – Init command opening the second session with number 2 and the following information: • Private Key ID of the target key. • Mode of operation: Pad and sign • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.4 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
5	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
6	T → IoT SAFE	Send a Compute Signature – Update command for the session number 1, requesting a signature generation for DATA_1.	RQ.2.10.1 RQ.2.10.2 RQ.2.10.3 RQ.2.10.4
7	T ← IoT SAFE	Expected response: ECDSA signature generated with the correct length and SW9000h (Successful execution).	RQ.2.10.5 RQ.2.10.6

8	T → IoT SAFE	Send a Compute Signature – Update command for the session number 2, requesting a signature generation for DATA_1. The Final hash field is correctly given.	RQ.2.10.1 RQ.2.10.2 RQ.2.10.3 RQ.2.10.4
9	T ← IoT SAFE	Expected response: ECDSA signature generated with the correct length and SW9000h (Successful execution).	RQ.2.10.5 RQ.2.10.6

4.1.4.6 Test Case 6

4.1.4.6.1 Test Case Description

Successful Compute Signature session opened, closed and opened again to ensure that the close functionality works as expected. The target key to be used is Private Key with Label 'GSMA CL CRED SKLB 02' according the Applet Store Default Configuration NIST key section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by Label.

4.1.4.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 1 and the following information: • Private Key Label of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
4	T → IoT SAFE	Send a Compute Signature – Init command closing a session with number 1.	RQ.2.9.7
5	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
6	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 1 and the following information: • Private Key Label of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
7	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9

4.1.4.7 Test Case 7

4.1.4.7.1 Test Case Description

Failure when trying to open a session with a number corresponding to another session already opened. The target key to be used is the Private Key with Label 'GSMA CL CRED SKLB 02' according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID.

4.1.4.7.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.7.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 1 and the following information: • Private Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
4	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 1 and the following information: • Private Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.6 RQ.2.9.8
5	T ← IoT SAFE	Expected response: SW6A86h (Incorrect P1, P2).	RQ.2.10.6

4.1.4.8 Test Case 8

4.1.4.8.1 Test Case Description

Failure when trying to close a session that has not been opened previously.

4.1.4.8.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.8.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 1.	RQ.2.9.1 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW9000h.	RQ.2.9.9
2	T → IoT SAFE	Send a Compute Signature – Init command closing a session with number 2.	RQ.2.9.1 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW6A86h (Incorrect P1, P2).	RQ.2.10.6

4.1.4.9 Test Case 9

4.1.4.9.1 Test Case Description

Failure when trying to open a session referencing a Private Key that does not exists in the Applet Store. For this case, no target key is indicated as it is the focus of the test case.

4.1.4.9.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.9.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 1 and the following information: • ID of a non-existing Private key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW6985h (Conditions of use not satisfied).	RQ.2.10.6

4.1.4.10 Test Case 10

4.1.4.10.1 Test Case Description

Failure when the target key used does not satisfy some of the required conditions such as being Activated or having Signature permissions. The target key to be used is the Private Key with Label ‘GSMA EPH CL Key Pair LB 01’ according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by Label.

4.1.4.10.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.10.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 1 and the following information: • Private Key Label of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW6985h (Conditions of use not satisfied).	RQ.2.10.6

4.1.4.11 Test Case 11

4.1.4.11.1 Test Case Description

Failure when trying to open a session after having reached the maximum number of sessions allowed. The target key to be used is the Private Key with Label ‘GSMA CL CRED SKLB 02’ according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID.

4.1.4.11.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.11.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 1 and the following information: • Private Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
4	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 2 and the following information: • Private Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
5	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
6	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 3 and the following information: • Private Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
7	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
8	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 4 and the following information: • Private Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.6 RQ.2.9.8
9	T ← IoT SAFE	Expected response: SW6989h (Maximum number of sessions reached).	RQ.2.9.9

4.1.4.12 Test Case 12

4.1.4.12.1 Test Case Description

Failure when requesting a data signature with the Compute Signature – Update command without having opened the correspondent session previously. The data used for signature is DATA_1 (5.1).

4.1.4.12.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.12.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Update command for the session number 1, requesting a signature generation for DATA_1.	RQ.2.10.1 RQ.2.10.2 RQ.2.10.4

3	T ← IoT SAFE	Expected response: SW6A86h (Incorrect P1, P2).	RQ.2.10.6

4.1.4.13 Test Case 13

4.1.4.13.1 Test Case Description

Failure when opening a session with Full text processing mode and requesting a data signature indicating the Final hash field, which does not correspond with the operation mode indicated at the beginning. The target key to be used is the Private Key with Label 'GSMA CL CRED SKLB 02' according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by Label and the data used for signature is DATA_2 (5.2).

4.1.4.13.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.4.13.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command opening a session with number 1 and the following information: • Private Key Label of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.9.1 RQ.2.9.2 RQ.2.9.5 RQ.2.9.6 RQ.2.9.8
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.9.9
4	T → IoT SAFE	Send a Compute Signature – Update command for the session number 1, requesting a signature generation for DATA_2. The Final hash field is correctly given.	RQ.2.10.1 RQ.2.10.2 RQ.2.10.3
5	T ← IoT SAFE	Expected response: SW6A80h (Incorrect data).	RQ.2.9.9

4.1.5 Generate Key Pair

4.1.5.1 Test Case 1

4.1.5.1.1 Test Case Description

Successful execution of Generate key pair command to check that applet returns Private/Public Key ID and public key data targeting the private key by ID.

4.1.5.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.5.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair command with value of Tag 84 for Private Key ID = **GSMA PKID 01**	RQ2.11.1 RQ2.11.3
3	T ← IoT SAFE	Expected response: Command response SHOULD contain Private key ID, public key ID and public key. Verify that Private and Public key ID length must be between 1-14h bytes and Public key data length must be 45h long.	RQ2.11.4

4.1.5.2 Test Case 2

4.1.5.2.1 Test Case Description

Successful execution of Generate key pair command to check that applet returns Private/Public Key ID and public key data targeting the private key by Label.

4.1.5.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.5.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair command with value of Tag 74 for Private Key Label = **GSMA KPLB 01**	RQ2.11.1 RQ2.11.3
3	T ← IoT SAFE	Expected response: Command response SHOULD contain Private key ID, public key ID and public key. Verify that Private and Public key ID length must be between 1-14h bytes and Public key data length must be 45h long.	RQ2.11.4

4.1.5.3 Test Case 3

4.1.5.3.1 Test Case Description

Generate key pair command SHOULD fail with SW6985h in case the target private key ID is missing.

4.1.5.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.5.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair command with value of Tag 74 as 11111111111111	RQ2.11.3
3	T ← IoT SAFE	SW6985h	RQ2.11.2

4.1.5.4 Test Case 4

4.1.5.4.1 Test Case Description

Generate key pair SHOULD fail with SW6985h in case the private key does not have key generation cryptographic function granted.

4.1.5.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.5.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair command with value of Tag 74 for Private Key LABEL = **GSMA CL CRED SKLB 02**	RQ2.11.3
3	T ← IoT SAFE	SW6985h	RQ2.11.2

4.1.5.5 Test Case 5

4.1.5.5.1 Test Case Description

Generate key pair command SHOULD fail with SW6985h in case the public key does not have READ access condition.

4.1.5.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.5.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair command with value of Tag 74 for Private Key LABEL = **GSMA EPH CL** Key Pair LB 03	RQ2.11.3
3	T ← IoT SAFE	SW6985h	RQ2.11.2

4.1.5.6 Test Case 6

4.1.5.6.1 Test Case Description

Generate key pair command SHOULD fail with SW6985h in case the Public Key is in another session.

4.1.5.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.5.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Put public key init command with value of Tag 85 as **GSMA EPH CL PKID 01** Send a Generate key pair command with value of Tag 74 as **GSMA EPH CL SKID 01**	RQ2.11.3
3	T ← IoT SAFE	SW6985h	RQ2.11.2

4.1.5.7 Test Case 7

4.1.5.7.1 Test Case Description

Generate key pair command SHOULD fail with SW6A80h in case the command data is wrong.

4.1.5.7.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.5.7.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair command with value of Tag values as FFh	RQ2.11.3
3	T ← IoT SAFE	SW6A80h	RQ2.11.2

4.1.5.8 Test Case 8

4.1.5.8.1 Test Case Description

Execution of Generate key pair command failed with SW6A86h in case the command parameter P1 is wrong.

4.1.5.8.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.5.8.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair with P1 as FFh	RQ2.11.3
3	T ← IoT SAFE	SW6A86h	RQ2.11.2

4.1.5.9 Test Case 9

4.1.5.9.1 Test Case Description

Execution of Generate key pair command failed with SW6A86h in case the command parameter P2 is wrong.

4.1.5.9.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.5.9.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair with P2 as FFh	RQ2.11.3
3	T ← IoT SAFE	SW6A86h	RQ2.11.2

4.1.6 Get data – Application

4.1.6.1 Test Case 1

4.1.6.1.1 Test Case Description

Successful execution of Get Data – Application command to check the information about the applet and its capacity.

4.1.6.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.6.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Application command	RQ2.12.1 RQ2.12.3
3	T ← IoT SAFE	Expected response: A list of all the data field contained by the Applet and codified according to section 2.8.1 NOTE: Response SHOULD be equivalent to section 2.8.1	RQ2.12.2 RQ2.12.4

4.1.6.2 Test Case 2

4.1.6.2.1 Test Case Description

Check that Get Data – Application command with an incorrect P1 fails with the defined status word.

4.1.6.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.6.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data - Application command with a P1 value as FFh	RQ2.12.3
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ2.12.2

4.1.6.3 Test Case 3

4.1.6.3.1 Test Case Description

Check that Get Data – Application command with an incorrect P2 fails with the defined status word.

4.1.6.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.6.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Application command with a P2 value as FFh	RQ2.12.3
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ2.12.2

4.1.6.4 Test Case 4

4.1.6.4.1 Test Case Description

Check that Get Data – Application command with wrong Le fails with the defined status word.

4.1.6.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.6.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Application with Le as FFh.	RQ2.12.3
3	T ← IoT SAFE	Execution failure with SW6700h	RQ2.12.2

4.1.7 Get data – File

4.1.7.1 Test Case 1

4.1.7.1.1 Test Case Description

Successful execution of Get Data – File command to check the File information of a specific file based on File ID/Label from the Applet Store. The target file to be used is the File 1.

4.1.7.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.7.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – File command sequence using File ID values from all files present within configurations defined in the initial conditions section.	RQ.2.13.1 RQ.2.13.3
3	T ← IoT SAFE	Expected response: A list of File information structure of the targeted File ID contained by the Applet Store and codified according [1] section 2.14.4.	RQ.2.13.2 RQ.2.13.4
4	T → IoT SAFE	Send a Get Data – File command sequence using File Label values from all files present within configurations defined in the initial conditions section.	RQ.2.13.1 RQ.2.13.3
5	T ← IoT SAFE	Expected response: A list of File information structure of the targeted File Label contained by the Applet Store and codified according [1] section 2.14.4.	RQ.2.13.2 RQ.2.13.4

-

4.1.7.2 Test Case 2

4.1.7.2.1 Test Case Description

Check that Get Data – File command with an incorrect P1 fails with the defined status word.

4.1.7.2.2 Initial Conditions

Applet installed according 2.8.1 with Applet Store containing the objects defined at 2.8.1.1.1, or 2.8.1.1.2 or 2.8.1.1.3.

4.1.7.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – File command with incorrect P1 value = FF	RQ.2.13.1
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.2.13.2

4.1.7.3 Test Case 3

4.1.7.3.1 Test Case Description

Check that Get Data – File command with an incorrect P2 fails with the defined status word.

4.1.7.3.2 Initial Conditions

Applet installed according 2.8.1 with Applet Store containing the objects defined at 2.8.1.1.1, or 2.8.1.1.2 or 2.8.1.1.3.

4.1.7.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – File command with an incorrect P2 value = FF	RQ.2.13.1
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.2.13.2

4.1.7.4 Test Case 4

4.1.7.4.1 Test Case Description

Check that Get Data – File command with a non-existent File ID/Label with the defined status word.

4.1.7.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.7.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – File command sequence using File ID "**GSMA FID 03**"	RQ.2.13.1
3	T ← IoT SAFE	Execution failure with SW6A82h	RQ.2.13.2
4	T → IoT SAFE	Send a Get Data – File command sequence using File ID "**GSMA FLB 03**"	RQ.2.13.1
5	T ← IoT SAFE	Execution failure with SW6A82h	RQ.2.13.2

4.1.7.5 Test Case 5

4.1.7.5.1 Test Case Description

Check that Get Data – File command with Incorrect data with the defined status word.

4.1.7.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.7.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – File command with an incorrect tag value in the data field. DataIn= FFh.	RQ.2.13.1
3	T ← IoT SAFE	Execution failure with SW6A80h	RQ.2.13.2

-

4.1.8 Get data – Object List

4.1.8.1 Test Case 1

4.1.8.1.1 Test Case Description

Successful execution of Get Data – Object List command to check the object list into the Applet Store.

4.1.8.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.8.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Object List command sequence	RQ.2.14.1 RQ.2.14.2 RQ.2.14.3 RQ.2.14.4
3	T ← IoT SAFE	Expected response: A list of all the objects (and their attributes) contained by the Applet Store and codified according [1] section 2.14.4.	RQ.2.14.5 RQ.2.14.6

4.1.8.2 Test Case 2

4.1.8.2.1 Test Case Description

Check that Get Data – Object List command with an incorrect P1 fails with the defined status word.

4.1.8.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.8.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data command with a P1 = 0xFF.	RQ.2.14.1 RQ.2.14.4
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.2.14.6

4.1.8.3 Test Case 3

4.1.8.3.1 Test Case Description

Check that Get Data – Object List command with an incorrect P2 fails with the defined status word.

4.1.8.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.8.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Object List command with a P2 = 0xFF.	RQ.2.14.1 RQ.2.14.4
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.2.14.6

4.1.8.4 Test Case 4

4.1.8.4.1 Test Case Description

Check that Get Data – Object List command first command with P2 indicating More Outgoing Data fails with the defined status word.

4.1.8.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.8.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Object List command with More Outgoing Data mode (P2 = 01h) for the first APDU.	RQ.2.14.1 RQ.2.14.4
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.2.14.6

4.1.8.5 Test Case 5

4.1.8.5.1 Test Case Description

Check that Get Data – Object List command sequence fails with the defined status word if there is some update in the Applet Store.

4.1.8.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.8.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Object List command with First Outgoing Data mode (P2 = 00h).	RQ.2.14.1 RQ.2.14.2 RQ.2.14.3 RQ.2.14.4
3	T ← IoT SAFE	Expected response: A list of all the objects (and their attributes) that fill in an APDU responses and codified according [1] section 2.14.4.	RQ.2.14.5 RQ.2.14.6

4	T → IoT SAFE	Send a Put Public Key - Init command (with targeted container ID **GSMA SV EPH PKID 02**.
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

5	T ← IoT SAFE	Response SW9000h	

6	T → IoT SAFE	Send a Put Public Key – Update command using Public Key value **PUBLIC_KEY_VALUE_002_ECC**
		
		P1 = 80h (Last incoming data)
		P2 = 01h (Session No.1)
		DataIn = **PUBLIC_KEY_VALUE_002_ECC**
			RQ.2.20.1
			RQ.2.20.3
			RQ.2.20.4

7	T → IoT SAFE	Send the next Get Data – Object List command with More Outgoing Data mode (P2 = 01h),	RQ.2.14.1 RQ.2.14.4
8	T ← IoT SAFE	Execution failure with SW6989h.	RQ.2.14.6
9	T → IoT SAFE	Send a Get Data – Object List command sequence	RQ.2.14.1 RQ.2.14.2 RQ.2.14.3 RQ.2.14.4
10	T ← IoT SAFE	Expected response: A list of all the objects (and their attributes) contained by the Applet Store and codified according [1] section 2.14.4.	RQ.2.14.5 RQ.2.14.6

4.1.9 Get data – Private Key Information

4.1.9.1 Test Case 1

4.1.9.1.1 Test Case Description

Successful execution of Get Data – Private key Information command to check that applet returns Private key Information from applet store when searching by key ID.

4.1.9.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.9.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Private key Information command with value of Tag 85 as specified in section 2.8.1.1.1 or 2.8.1.1.2, or 2.8.1.1.3 for Private Key ID = **GSMA EPH CL PKID 01** (depending on which Applet Store Configuration is used)	RQ2.15.1 RQ2.15.3
3	T ← IoT SAFE	Expected response: Command response with the values according section 2.8.1.1.1, or 2.8.1.1.2, or 2.8.1.1.3 for Private Key ID = **GSMA EPH CL PKID 01** (depending on which Applet Store Configuration is used)	RQ2.15.2 RQ2.15.4

4.1.9.2 Test Case 2

4.1.9.2.1 Test Case Description

Successful execution of Get Data Private key Information command to check that applet returns Private key Information from applet store searching by key label.

4.1.9.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.9.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Private key Information command with value of Tag 75 as specified in section 2.8.1.1.1, or 2.8.1.1.2, or 2.8.1.1.3 for Private Key Label = **GSMA EPH CL** Key Pair LB 01 (depending on which Applet Store Configuration is used)	RQ2.15.1 RQ2.15.3
3	T ← IoT SAFE	Expected response: Command response with the values according section 2.8.1.1.1, or 2.8.1.1.2, or 2.8.1.1.3 for Private Key Label = **GSMA EPH CL** Key Pair LB 01 (depending on which Applet Store Configuration is used)	RQ2.15.2 RQ2.15.4

4.1.9.3 Test Case 3

4.1.9.3.1 Test Case Description

Check that Get Data – Private key Information command fails when ID sent in a command does not match with any private key ID present in the applet store.

4.1.9.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.9.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Private key Information command with value of Tag 85 as 11111111111111	RQ2.15.3
3	T ← IoT SAFE	SW6985h	RQ2.15.2

4.1.9.4 Test Case 4

4.1.9.4.1 Test Case Description

Check that Get Data – Private key Information command fails when label sent in a command does not match with any private key label present in the applet store.

4.1.9.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.9.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Private key Information command with value of Tag 75 as 11111111111111	RQ2.15.3
3	T ← IoT SAFE	SW6985h	RQ2.15.2

4.1.9.5 Test Case 5

4.1.9.5.1 Test Case Description

Check that Get Data – Private key Information with an incorrect P1 fails with the defined status word.

4.1.9.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.9.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Private key information command with a P1 value as FFh	RQ2.15.3
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ2.15.2

4.1.9.6 Test Case 6

4.1.9.6.1 Test Case Description

Check that Get Data – Private key Information with an incorrect P2 fails with the defined status word.

4.1.9.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.9.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Private key information command with a P2 value as FFh	RQ2.15.3
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ2.15.2

4.1.10 Get data – Public Key Information

4.1.10.1 Test Case 1

4.1.10.1.1 Test Case Description

Successful execution of Get Data – public key command to retrieve the public key information from the Applet Store, when the search by Public key ID is used.

4.1.10.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.10.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data –public key command with value of Tag 85 as specified in section 2.8.1.1.1, or 2.8.1.1.2, or 2.8.1.1.3 for Public Key ID = **GSMA EPH CL PKID 01** (depending on which Applet Store Configuration is used)	RQ2.16.1 RQ2.16.3
3	T ← IoT SAFE	Expected response: Command response as reported in 2.16.4.1 of [1] with the values according section 2.8.1.1.1, or 2.8.1.1.3, or 2.8.1.1.4 for Public Key ID = **GSMA EPH CL PKID 01** (depending on which Applet Store Configuration is used)	RQ2.16.2 RQ2.16.4

4.1.10.2 Test Case 2

4.1.10.2.1 Test Case Description

Successful execution of Get Data – public key command to retrieve the public key information from the Applet Store when the search by Public key Label is used.

4.1.10.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.10.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – public key command sequence with value of Tag 75 as specified in section 2.8.1.1.1, or 2.8.1.1.3, or 2.8.1.1.4 for Public Key Label = **GSMA EPH CL Key Pair LB 01** (depending on which Applet Store Configuration is used)	RQ2.16.1 RQ2.16.3
3	T ← IoT SAFE	Expected response: Command response as reported in 2.16.4.1 of [1] ,with the values according section 2.8.1.1.1, or 2.8.1.1.3, or 2.8.1.1.4 for Public Key Label = **GSMA EPH CL Key Pair LB 01** (depending on which Applet Store Configuration is used)	RQ2.16.2 RQ2.16.4

4.1.10.3 Test Case 3

4.1.10.3.1 Test Case Description

Check that Get Data– public key command fails when the label sent in a command does not match with any public key label present in the applet store.

4.1.10.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.10.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – public key command with value of Tag 85 = 3030303030303030	RQ2.16.3
3	T ← IoT SAFE	Execution failure with SW6985h	RQ2.16.2

4.1.10.4 Test Case 4

4.1.10.4.1 Test Case Description

Check that Get Data– public key command fails when the ID sent in a command does not match with any public key ID present in the applet store.

4.1.10.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.10.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – public key command with value of Tag 75 = 3030303030303030	RQ2.16.3
3	T ← IoT SAFE	Execution failure with SW6985h	RQ2.16.2

4.1.10.5 Test Case 5

4.1.10.5.1 Test Case Description

Check that Get Data – Public key Information with an incorrect P1 fails with the defined status word.

4.1.10.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.10.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Public key information command with a P1 value as FFh	RQ2.16.3
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ2.16.2

4.1.10.6 Test Case 6

4.1.10.6.1 Test Case Description

Check that Get Data – Public key Information with an incorrect P2 fails with the defined status word.

4.1.10.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.10.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Public key information command with a P2 value as FFh	RQ2.16.3
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ2.16.2

4.1.11 Get data – Secret key information

4.1.11.1 Test Case 1

4.1.11.1.1 Test Case Description

Successful execution of Get Data – secret key information command to check the secret key information into the Applet Store, when the search by Secret key ID is used

4.1.11.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.11.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – secret key information command with value of Tag 86 for secret key ID = **GSMA SEC KEY ID 01**	RQ2.17.1
3	T ← IoT SAFE	Expected response: Command response as reported in 2.17.4.1 of [1] ,with the values according to section 2.8.1.1.4 for key ID = **GSMA SEC KEY ID 01**	RQ2.17.1

4.1.11.2 Test Case 2

4.1.11.2.1 Test Case Description

Successful execution of Get Data – secret key information command to check the secret key information into the Applet Store, when the search by Secret key Label is used

4.1.11.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.11.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – secret key information command sequence with value of Tag 76 as specified in section 2.8.1.1.4 for key Label = **GSMA SECRET KEY LB 01**	RQ2.17.1
3	T ← IoT SAFE	Expected response: Command response as reported in 2.17.4.1 of [1] ,with the values according section 2.8.1.1.4 for key Label = **GSMA SECRET KEY LB 01**	RQ2.17.1

4.1.11.3 Test Case 3

4.1.11.3.1 Test Case Description

Successful execution of Get Data – secret key information command to check the secret key information into the Applet Store, when the search by Secret key ID is used and the secret key does not have a label.

4.1.11.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.11.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – secret key information command with value of Tag 86 as specified in section 2.8.1.1.4 for key ID = **GSMA SEC KEY ID 02**	RQ2.17.1
3	T ← IoT SAFE	Expected response: Command response as reported in 2.17.4.1 of [1] ,with the values section 2.8.1.1.4 for key ID = **GSMA SEC KEY ID 02**	RQ2.17.1

4.1.11.4 Test Case 4

4.1.11.4.1 Test Case Description

Check that Get Data– secret key information command fails when the Label is not present.

4.1.11.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.11.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – secret key information command with value of Tag 86 = 3030303030303030	RQ2.17.1
3	T ← IoT SAFE	Execution failure with SW6985h	RQ2.17.1

4.1.11.5 Test Case 5

4.1.11.5.1 Test Case Description

Check that Get Data –secret key information command fails when the ID is not present.

4.1.11.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.11.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – secret key information command with value of Tag 76 = 3030303030303030	RQ2.17.1
3	T ← IoT SAFE	Execution failure with SW6985h	RQ2.17.1

4.1.12 Get Random

4.1.12.1 Test Case 1

4.1.12.1.1 Test Case Description

Successful execution of Get Random command with expected length of 256 bytes and get a random number back.

4.1.12.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.12.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Random command with expected length of 256 bytes	RQ2.18.1 RQ2.18.3
3	T ← IoT SAFE	Expected response: Random number of 256 bytes	RQ2.18.2 RQ2.18.4

4.1.12.2 Test Case 2

4.1.12.2.1 Test Case Description

Successful execution of Get Random command with expected length of 20 bytes and get a random number back.

4.1.12.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.12.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Random command with expected length of 20 bytes	RQ2.18.1 RQ2.18.3
3	T ← IoT SAFE	Expected response: Random number of 20 bytes	RQ1.18.2 RQ2.18.4

4.1.12.3 Test Case 3

4.1.12.3.1 Test Case Description

Execution of Get Random command failed with SW6A86h.

4.1.12.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.12.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair with P1 as FFh	RQ2.18.3
3	T ← IoT SAFE	SW6A86h	RQ2.18.2

4.1.12.4 Test Case 4

4.1.12.4.1 Test Case Description

Execution of Get Random command failed with SW6A86h.

4.1.12.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.12.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Generate key pair with P2 as FFh	RQ2.18.3
3	T ← IoT SAFE	SW6A86h	RQ2.18.2

4.1.13 Put Public Key

4.1.13.1 Test Case 1

4.1.13.1.1 Test Case Description

Successful execution of Put Public Key - Init and update command to test a put public key command.

4.1.13.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key - Init command (with targeted container ID **GSMA SV EPH PKID 02**.
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

3	T ← IoT SAFE	Response SW9000h	

4	T → IoT SAFE	Send a Put Public Key – Update command using Public Key value **PUBLIC_KEY_VALUE_002_ECC**
		P1 = 80h (Last incoming data)
		P2 = 01h (Session No.1)
		DataIn = **PUBLIC_KEY_VALUE_002_ECC**
			RQ.2.20.1
			RQ.2.20.3
			RQ.2.20.4

5	T ← IoT SAFE	Response SW9000h	

6	T → IoT SAFE	Send a Read Public Key command with the target being **ECC_PUB_ID_02**
		P1 = 00h
		P2 = 00h (Last outgoing command)
		DataIn = **ECC_PUB_ID_02**
			RQ.2.22.1
			RQ.2.22.3

7	T ← IoT SAFE	- ECC Public Key **PUBLIC_KEY_VALUE_002_ECC** - Response SW9000h	RQ.2.22.4

4.1.13.2 Test Case 2

4.1.13.2.1 Test Case Description

Check that Put Public Key – Init command can process Open Session, Cancel Session and Open Session parameters again to ensure that the close functionality works as expected.

4.1.13.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with targeted container ID **GSMA SV EPH PKID 02**.
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

3	T ← IoT SAFE	Response SW9000h	

4	T → IoT SAFE	Send an Put Public Key – Init(Cancel Session) command
		
		P1 = 01h (Cancel Session)
		P2 = 01h (Session No.1)
			RQ.2.19.2

5	T ← IoT SAFE	Response SW9000h	

6	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command again (with targeted container ID **GSMA SV EPH PKID 02** using same session number).
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

7	T ← IoT SAFE	Response SW9000h	

4.1.13.3 Test Case 3

4.1.13.3.1 Test Case Description

Check that Put Public Key – Init(Open Session) command cannot open the same session twice

4.1.13.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with targeted container ID **GSMA SV EPH PKID 02**.
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

3	T ← IoT SAFE	Response SW9000h	

4	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with targeted container ID **GSMA EPH CL PKID 01**) again with the same session number.
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA EPH CL PKID 01**
			RQ.2.19.3

5	T ← IoT SAFE	Response SW6A86h	

4.1.13.4 Test Case 4

4.1.13.4.1 Test Case Description

Check that Put Public Key – Init(Cancel Session) command with Lc not 00h.

4.1.13.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with targeted container ID **GSMA SV EPH PKID 02**.
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

3	T ← IoT SAFE	Response SW9000h	

4	T → IoT SAFE	Send an Put Public Key – Init(Cancel Session) command
		
		P1 = 01h (Cancel Session)
		P2 = 01h (Session No.1)
		Lc = 01h
			RQ.2.19.3

5	T ← IoT SAFE	Response SW6700h	

4.1.13.5 Test Case 5

4.1.13.5.1 Test Case Description

Check that Put Public Key – Init(Cancel Session) command cannot close unopened sessions

4.1.13.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with targeted container ID **GSMA SV EPH PKID 02**.
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

3	T ← IoT SAFE	Response SW9000h	

4	T → IoT SAFE	Send an Put Public Key – Init(Cancel Session) command with no sessions opened yet.
		
		P1 = 01h (Cancel Session)
		P2 = 02h (Session No.2)
			RQ.2.19.3

5	T ← IoT SAFE	Response SW6A86h	

6	T → IoT SAFE	Send a Put Public Key – Update command using Public Key value **PUBLIC_KEY_VALUE_002_ECC**) with same session number 01h
		
		P1 = 80h (Last incoming data)
		P2 = 01h (Session No.1)
		DataIn = **PUBLIC_KEY_VALUE_002_ECC**
			RQ.2.20.1
			RQ.2.20.3
			RQ.2.20.4

7	T ← IoT SAFE	Response SW9000h	

4.1.13.6 Test Case 6

4.1.13.6.1 Test Case Description

Check that Put Public Key - Init command with an incorrect P1 fails with the defined status word.

4.1.13.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Init command with Invalid P1 value. P1 SHOULD be either 00h (Open Session) or 01h (Cancel Session) only.
		
		P1 = 02h (Invalid P1)
		P2 = 01h (Session No.1)
			RQ.2.19.3

3	T ← IoT SAFE	Response SW6A86h	

4.1.13.7 Test Case 7

4.1.13.7.1 Test Case Description

Check that Put Public Key – Init command with a non-existent Key ID/Label fails with the defined status word.

4.1.13.7.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.7.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with targeted container ID **ECC_PUB_ID_09_NONEXISTENT**
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **ECC_PUB_ID_09_NONEXISTENT**
			RQ.2.19.3

1	T ← IoT SAFE	Response SW6985h	

4.1.13.8 Test Case 8

4.1.13.8.1 Test Case Description

Check that Put Public Key - Init command with open session up to maximum limit of sessions allowed fails with the defined status word.

4.1.13.8.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.8.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command with targeted container ID **GSMA SV EPH PKID 02**.
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

3	T ← IoT SAFE	Response SW9000h	
4	T → IoT SAFE T ← IoT SAFE	Repeat Step 2 and Step 3 n times, where n is the maximum number of sessions allowed. P2 values is increased by 1 in each Put Public Key – Init command.	RQ.2.19.1 RQ.2.19.4

5	T → IoT SAFE	Send a Put Public Key – Init(Open n+1th Session) command with targeted container ID **GSMA SV EPH PKID 02**.
		
		P1 = 00h (Open Session)
		P2 = n+1h (Session No.n+1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.3

6	T ← IoT SAFE	Response SW6989h	

4.1.13.9 Test Case 9

4.1.13.9.1 Test Case Description

Check that Put Public Key – Update command with no sessions opened previously fails with the defined status word.

4.1.13.9.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.9.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Update command using Public Key value **PUBLIC_KEY_VALUE_002_ECC**
		
		P1 = 80h (Last incoming data)
		P2 = 01h (Session No.1)
		DataIn = **PUBLIC_KEY_VALUE_002_ECC**
			RQ.2.20.1
			RQ.2.20.2

3	T ← IoT SAFE	Response SW6A86h	

4.1.13.10 Test Case 10

4.1.13.10.1 Test Case Description

Check that Put Public Key – Init command with Public Key not granted for update fails with the defined status word.

4.1.13.10.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.10.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with targeted container ID **GSMA ISSUER PKID 03**)
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **PUBLIC_KEY_VALUE_002_ECC**
			RQ.2.19.3

3	T ← IoT SAFE	Response SW6985h	

4.1.13.11 Test Case 11

4.1.13.11.1 Test Case Description

Check that Put Public Key – Init(Open Session) command cannot use same Public Key in another Put Public Key cryptographic command session with the defined status word.

4.1.13.11.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.11.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with targeted container ID **GSMA SV EPH PKID 02**).
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

3	T ← IoT SAFE	Response SW9000h	

4	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with same targeted container ID **GSMA SV EPH PKID 02** but different session number).
		
		P1 = 00h (Open Session)
		P2 = 02h (Session No.2)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.3

5	T ← IoT SAFE	Response SW6A85h	

4.1.13.12 Test Case 12

4.1.13.12.1 Test Case Description

Check that Put Public Key – Init(Open Session) command cannot use same Public Key in different cryptographic command session with the defined status word.

4.1.13.12.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.12.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Init (Open Session) command (with targeted container ID **GSMA SV EPH PKID 02**). P1 = 00h (Open Session) P2 = 01h (Session No.1) DataIn: Public Key ID = **GSMA SV EPH PKID 02** Mode of operation = Full text Hash algorithm = SHA-256 Signature algorithm = ECDSA	RQ1.1.3
3	T ← IoT SAFE	Response SW9000h	

4	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with same targeted container ID **GSMA SV EPH PKID 02**).
		
		P1 = 00h (Open Session)
		P2 = 02h (Session No.2)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.3

5	T ← IoT SAFE	Response SW6A85h	

4.1.13.13 Test Case 13

4.1.13.13.1 Test Case Description

Check that Put Public Key – Init command with Private Key (associated to a Public Key) not granted for update fails with the defined status word.

4.1.13.13.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.13.13.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key – Init(Open Session) command (with targeted container ID **GSMA PKID 01**)
		
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **PUBLIC_KEY_VALUE_002_ECC**
			RQ.2.19.3

1	T ← IoT SAFE	Response SW6985h	

4.1.14 Read file

4.1.14.1 Test Case 1

4.1.14.1.1 Test Case Description

Successful execution of Read file command with expected data.

4.1.14.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.14.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Execute get data file with value of Tag 83 as specified in section 2.8.1.1.1 for file1 to retrieve the size of file. Send a Read File command with value of Tag 83	RQ2.21.1 RQ2.21.3
3	T ← IoT SAFE	Response SHOULD contain data as mentioned in File 1	RQ2.21.2 RQ2.21.4

4.1.14.2 Test Case 2

4.1.14.2.1 Test Case Description

Execution of Read file command failed with expected SW.

4.1.14.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.14.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Read File command with value of Tag 83 as specified in section 2.8.1.1.1 for file1 and offset outside file data	RQ2.21.3
3	T ← IoT SAFE	SW6981h	RQ2.21.2

4.1.14.3 Test Case 3

4.1.14.3.1 Test Case Description

Execution of Read file command failed with expected SW since READ condition is not granted.

4.1.14.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.14.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Read File command with value of Tag 83 as specified in section 2.8.1.1.1 for File 1_a.	RQ2.21.3
3	T ← IoT SAFE	SW6985h	RQ2.21.2

4.1.14.4 Test Case 4

4.1.14.4.1 Test Case Description

Execution of Read file command failed with expected SW since FILE is deactivated.

4.1.14.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.14.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Read File command with value of Tag 83 as specified in section 2.8.1.1.1 for File 1_b .	RQ2.21.3
3	T ← IoT SAFE	SW6985h	RQ2.21.2

4.1.14.5 Test Case 5

4.1.14.5.1 Test Case Description

Execution of Read file command failed with expected SW.

4.1.14.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.14.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send Read File command with Incorrect tag value as FFh	RQ2.21.3
3	T ← IoT SAFE	SW6A80h	RQ2.21.2

4.1.14.6 Test Case 6

4.1.14.6.1 Test Case Description

Execution of Read file command failed with expected SW.

4.1.14.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.14.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Read File command with value of Tag 83 as FFFF.FF.	RQ2.21.3
3	T ← IoT SAFE	SW6A82h	RQ2.21.2

4.1.15 Read Public Key

4.1.15.1 Test Case 1

4.1.15.1.1 Test Case Description

Successful execution of Read Public Key command on an ECC Key Pair

4.1.15.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.15.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Read Public Key command with the target being **ECC_PUB_ID_02**
		
		P1 = 00h
		P2 = 00h (Last outgoing command)
		DataIn = **ECC_PUB_ID_02**
			RQ.2.22.1
			RQ.2.22.3

3	T ← IoT SAFE	- ECC Public Key **PUBLIC_KEY_VALUE_002_EC** - Response SW9000h	RQ.2.22.4

4.1.15.2 Test Case 2

4.1.15.2.1 Test Case Description

Check that Read Public Key command with an invalid P1 parameter fails with the defined status word.

4.1.15.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.15.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send Read Public Key command with invalid P1 value. P1 SHOULD only be 00h.
		P1 = 01h (Invalid value)
		P2 = 00h (Last outgoing command)
		DataIn = **ECC_PUB_ID_02**
			RQ.2.22.2

3	T ← IoT SAFE	Response SW6A86h	

4.1.15.3 Test Case 3

4.1.15.3.1 Test Case Description

Check that Read Public Key command fails on key containers that are deactivated after Put Public Key – Init open session command with the defined status word.

4.1.15.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.15.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Put Public Key - Init command (with targeted container ID **GSMA SV EPH PKID 02**.
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

3	T ← IoT SAFE	Response SW9000h	

4	T → IoT SAFE	Send a Put Public Key – Update command using Public Key value **PUBLIC_KEY_VALUE_002_ECC**
		P1 = 80h (Last incoming data)
		P2 = 01h (Session No.1)
		DataIn = **PUBLIC_KEY_VALUE_002_ECC**
			RQ.2.20.1
			RQ.2.20.3
			RQ.2.20.4

5	T ← IoT SAFE	Response SW9000h	

6	T → IoT SAFE	Send a Read Public Key command with the target being **ECC_PUB_ID_02**
		P1 = 00h
		P2 = 00h (Last outgoing command)
		DataIn = **ECC_PUB_ID_02**
			RQ.2.22.1
			RQ.2.22.3

7	T ← IoT SAFE	- ECC Public Key **PUBLIC_KEY_VALUE_002_ECC** Response SW9000h	RQ.2.22.4

8	T → IoT SAFE	Send a Put Public Key - Init command (with targeted container ID **GSMA SV EPH PKID 02**.
		P1 = 00h (Open Session)
		P2 = 01h (Session No.1)
		DataIn = **GSMA SV EPH PKID 02**
			RQ.2.19.1
			RQ.2.19.4

9	T ← IoT SAFE	Response SW9000h	

10	T → IoT SAFE	Send a Read Public Key command on a non-activated Key Container Label : **ECC_PUB_ID_02**
		P1 = 00h
		P2 = 00h (Last outgoing command)
		DataIn = **ECC_PUB_ID_02**
			RQ.2.22.1
			RQ.2.22.2

11	T ← IoT SAFE	Response SW6985h	

4.1.15.4 Test Case 4

4.1.15.4.1 Test Case Description

Check that Read Public Key command with a non-existent Key ID/Label fails with the defined status word.

4.1.15.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.15.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Read Public Key command on a non-existent Key Container ID **ECC_PUB_ID_03**
		
		P1 = 00h
		P2 = 00h (Last outgoing command)
		DataIn = **ECC_PUB_ID_03**
			RQ.2.22.1
			RQ.2.22.2

3	T ← IoT SAFE	Response SW6985h	

4.1.15.5 Test Case 5

4.1.15.5.1 Test Case Description

Check that Read Public Key command with an invalid P2 parameter fails with the defined status word.

4.1.15.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.15.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send Read Public Key command with invalid P2 value. P2 SHOULD only be 00h or 01h.
		
		P1 = 00h
		P2 = 03h (Invalid value)
		DataIn = **ECC_PUB_ID_02**
			RQ.2.22.2

3	T ← IoT SAFE	Response SW6A86h	

4.1.15.6 Test Case 6

4.1.15.6.1 Test Case Description

Check that Read Public Key command with Public Key not granted for read access condition fails with the defined status word.

4.1.15.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.15.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Read Public Key command with the target being **GSMA SERVER PKID 04**
		
		P1 = 00h
		P2 = 00h (Last outgoing command)
		DataIn = **GSMA SERVER PKID 04**
			RQ.2.22.2

3	T ← IoT SAFE	Response SW6985h	

4.1.15.7 Test Case 7

4.1.15.7.1 Test Case Description

Check that Read Public Key command with a single public key (not in a key pair).

4.1.15.7.2 Initial Condition

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.15.7.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Read Public Key command with the target being **GSMA ISSUER PKLB 03**
		
		P1 = 00h
		P2 = 00h (Last outgoing command)
		DataIn = **GSMA ISSUER PKLB 03**
			RQ.2.22.2

3	T ← IoT SAFE	Response SW9000h	

4.1.15.8 Test Case 8

4.1.15.8.1 Test Case Description

Check that Read Public Key command with a volatile ECC key before and after key generation.

4.1.15.8.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.15.8.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Read Public Key command with the target being **GSMA EPH CL Key Pair LB 03**
		
		P1 = 00h
		P2 = 00h (Last outgoing command)
		DataIn = **GSMA EPH CL Key Pair LB 03**
			RQ2.22.2

3	T ← IoT SAFE	Response SW6985h	RQ2.22.4
4	T → IoT SAFE	Send a Generate key pair command with value of Tag 84 for **GSMA EPH CL Key Pair LB 03**	RQ2.11.1 RQ2.11.3
5	T ← IoT SAFE	Expected response: Command response SHOULD contain Private key ID, public key ID and public key. Verify that Private and Public key ID length must be between 1-14h bytes and Public key data length must be 45h long.	RQ2.11.4

4.1.16 Verify Signature

4.1.16.1 Test Case 1

4.1.16.1.1 Test Case Description

Successful signature verification using Full text processing mode without requiring chaining. The target key to be used is the Public Key with Label GSMA KPLB 01 according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID. The data and signature used are DATA_1 (5.1) and SIGNATURE_1.

4.1.16.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
4	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, requesting a signature verification for DATA_1 and SIGNATURE_1.	RQ.2.24.1 RQ.2.24.2 RQ.2.24.3 RQ.2.24.4
5	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.24.5

4.1.16.2 Test Case 2

4.1.16.2.1 Test Case Description

Successful signature verification using Full text processing mode requiring chaining. The target key to be used is the Public Key with Label GSMA KPLB 01 according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by Label. The data and signature used are DATA_2 (5.2) and SIGNATURE_2.

4.1.16.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key Label of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
4	T → IoT SAFE	Send two Verify Signature – Update commands forming the chaining sequence for requesting the signature verification of DATA_2 and SIGNATURE_2, for session number 1.	RQ.2.24.1 RQ.2.24.2 RQ.2.24.3 RQ.2.24.4
5	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.24.5

4.1.16.3 Test Case 3

4.1.16.3.1 Test Case Description

Successful signature verification using Pad and sign processing mode. The target key to be used is the Public Key with Label GSMA ISSUER PKLB 03 according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID. The data and signature used are DATA_1 (5.1) and SIGNATURE_1.

4.1.16.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key ID of the target key. • Mode of operation: Pad and sign • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.3 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
4	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, requesting a signature verification for DATA_1(HASH) and SIGNATURE_1. The Final hash field is correctly given.	RQ.2.24.1 RQ.2.24.2 RQ.2.24.3 RQ.2.24.4
5	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.24.5

4.1.16.4 Test Case 4

4.1.16.4.1 Test Case Description

Two different Verify Signature sessions are opened, and a signature is verified for each one in order to ensure a correct behaviour when having concurrency between sessions:
For the first session, Full text processing mode is used and the target key to be used is the Public Key with Label GSMA KPLB 01 according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID and the data used for signature is DATA_1 (5.1).
For the second session, Pad and sign processing mode is used and the target key is the same one used before. The key is referenced by Label and the data used for signature is DATA_1(HASH) (5.1).

4.1.16.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Init command opening the first session with number 1 and the following information: • Public Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
4	T → IoT SAFE	Send a Verify Signature – Init command opening the second session with number 2 and the following information: • Public Key Label of the target key. • Mode of operation: Pad and sign • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.3 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
5	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
6	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, requesting a signature verification for DATA_1 and SIGNATURE_1.	RQ.2.24.1 RQ.2.24.2 RQ.2.24.3 RQ.2.24.4
7	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.24.5
8	T → IoT SAFE	Send a Verify Signature – Update command for the session number 2, requesting a signature verification for DATA_1 and SIGNATURE_1. The Final hash field is correctly given	RQ.2.24.1 RQ.2.24.2 RQ.2.24.3 RQ.2.24.4
9	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.24.5

4.1.16.5 Test Case 5

4.1.16.5.1 Test Case Description

Successful Verify Signature session opened, closed and opened again to ensure that the close functionality works as expected. The target key to be used is the Public Key with Label GSMA ISSUER PKLB 03 according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID.

4.1.16.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7

3	T← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
4	T → IoT SAFE	Send a Verify Signature – Init command closing a session with number 1.	RQ.2.23.6
5	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
6	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
7	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8

4.1.16.6 Test Case 6

4.1.16.6.1 Test Case Description

Failure on signature verification when the given signature does not match with the provided data. Use full Text processing mode. The target key to be used is the Public Key with Label GSMA ISSUER PKLB 03 according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by Label. The data and signature used are DATA_1 (5.1) and a non-matching signature.

4.1.16.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key Label of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
4	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, requesting a signature verification for DATA_1 and a non matching signature, DATA_2(HASH)	RQ.2.24.1 RQ.2.24.2 RQ.2.24.3
5	T ← IoT SAFE	Expected response: SW6D01h (Provided signature does not match).	RQ.2.24.5

4.1.16.7 Test Case 7

4.1.16.7.1 Test Case Description

Failure when trying to open a session with a number corresponding to another session already opened. The target key to be used is the Public Key with Label GSMA ISSUER PKLB 03 according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID.

4.1.16.7.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.7.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
4	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.5 RQ.2.23.7
5	T ← IoT SAFE	Expected response: SW6A86h (Incorrect P1, P2).	RQ.2.23.8

4.1.16.8 Test Case 8

4.1.16.8.1 Test Case Description

Failure when trying to close a session that has not been opened previously.

4.1.16.8.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.8.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute Signature – Init command closing a session with number 1.	RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW6A86h (Incorrect P1, P2).	RQ.2.23.8

4.1.16.9 Test Case 9

4.1.16.9.1 Test Case Description

Failure when trying to open a session referencing a Public Key that does not exist in the Applet Store. For this case, no target key is indicated as it is the focus of the test case.

4.1.16.9.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.9.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • ID of a non existing Public Key. (**ECC_PUB_ID_09_NONEXISTENT**) • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW6985h (Conditions of use not satisfied).	RQ.2.23.8

4.1.16.10 Test Case 10

4.1.16.10.1 Test Case Description

Failure when the target key used does not satisfy some of the required conditions such as being Activated or having Signature permissions. The target key to be used is the Public Key with GSMA EPH CL Key Pair LB 01 according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID.

4.1.16.10.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.10.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW6985h (Conditions of use not satisfied).	RQ.2.23.8

4.1.16.11 Test Case 11

4.1.16.11.1 Test Case Description

Failure when trying to open a session after having reached the maximum number of sessions allowed. The target key to be used is the Public Key with Label GSMA ISSUER PKLB 03according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by ID.

4.1.16.11.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.11.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
4	T → IoT SAFE	Repeat Step 2 and Step 3 5 times, where n is the maximum number of sessions allowed. The number of sessions is increased by 1 in each Verify Signature command.	
5	T → IoT SAFE T ← IoT SAFE	Send a Verify Signature – Init command opening a session with number 6 and the following information: • Public Key ID of the target key. • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
6	T ← IoT SAFE	Expected response: SW6989h (Maximum number of sessions reached).	RQ.2.23.8

4.1.16.12 Test Case 12

4.1.16.12.1 Test Case Description

Failure when requesting a signature verification with the Verify Signature – Update command without having opened the correspondent session previously. The data used for signature is DATA_1 (5.1). Also try to cancel a session that has not been previously opened.

4.1.16.12.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.12.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, requesting a signature verification for DATA_1 and SIGNATURE_1.	RQ.2.24.1 RQ.2.24.2 RQ.2.24.4
3	T ← IoT SAFE	Expected response: SW6A86h (Incorrect P1, P2).	RQ.2.24.5
4	T → IoT SAFE	Send a Verify Signature – Init command opening a session number 1	RQ.2.23.1 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
5	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
6	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, requesting a signature verification for DATA_1 and SIGNATURE_1.	RQ.2.24.1 RQ.2.24.2 RQ.2.24.3 RQ.2.24.4
7	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.24.5
8	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, requesting a signature verification for DATA_1 and SIGNATURE_1.	RQ.2.24.1 RQ.2.24.2 RQ.2.24.4
9	T ← IoT SAFE	Expected response: SW6A86h (Incorrect P1, P2).	RQ.2.24.5
10	T → IoT SAFE	Send a Verify Signature – Init command opening a session number 1	RQ.2.23.1 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7

11	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, requesting a signature verification for DATA_1 and SIGNATURE_1.	RQ.2.24.1 RQ.2.24.2 RQ.2.24.3 RQ.2.24.4
12	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.24.5
13	T → IoT SAFE	Send a Verify Signature – Init command cancelling a session number 1	RQ.2.23.7
14	T ← IoT SAFE	Expected response: SW6A86h (Incorrect P1, P2).	RQ.2.23.8

4.1.16.13 Test Case 13

4.1.16.13.1 Test Case Description

Failure when opening a session with Pad and sign processing mode and requesting a data signature indicating the Data for which signature verification is requested field, which does not correspond with the operation mode indicated at the beginning. The target key to be used is the Public Key with Label GSMA KPLB 01 according the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or Brainpool section (2.8.1.1.2). The key is referenced by Label and the data used for signature is DATA_1 (5.1).

4.1.16.13.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.13.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key Label of the target key. • Mode of operation: Pad and sign • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.3 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
4	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, requesting a signature verification for plain DATA_1 and SIGNATURE_1.	RQ.2.24.1 RQ.2.24.3 RQ.2.24.4
5	T ← IoT SAFE	Expected response: SW6A80h (Incorrect data).	RQ.2.24.5

4.1.16.14 Test Case 14

4.1.16.14.1 Test Case Description

Failure when trying to open a session with missing mandatory data. For this case, no target key is indicated as it is the focus of the test case.

4.1.16.14.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.14.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Mode of operation: Full text • Hash algorithm: SHA-256 • Signature algorithm: ECDSA Note that no Public Key reference is indicated in the payload data.	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW6985h (Conditions of use not satisfied).	RQ.2.23.8
4	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key Label of the target key. • Hash algorithm: SHA-256 • Signature algorithm: ECDSA Note that no mode of operation is indicated in the payload data.	RQ.2.23.1 RQ.2.23.2 RQ.2.23.4 RQ.2.23.7
5	T ← IoT SAFE	Expected response: SW6985h (Conditions of use not satisfied).	RQ.2.24.5

4.1.16.15 Test Case 15

4.1.16.15.1 Test Case Description

Failure when trying to update a session without mandatory data. For this case, no target key is indicated as it is the focus of the test case.

4.1.16.15.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 1 applicability table under the section 2.2.1.

4.1.16.15.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Verify Signature – Init command opening a session with number 1 and the following information: • Public Key Label of the target key. • Mode of operation: Pad and sign • Hash algorithm: SHA-256 • Signature algorithm: ECDSA	RQ.2.23.1 RQ.2.23.3 RQ.2.23.4 RQ.2.23.5 RQ.2.23.7
3	T ← IoT SAFE	Expected response: SW9000h (Successful execution).	RQ.2.23.8
4	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, without Signature data in the payload	RQ.2.24.1 RQ.2.24.2 RQ.2.24.3
5	T ← IoT SAFE	Expected response: SW6A80h (Incorrect data).	RQ.2.24.5
6	T → IoT SAFE	Send a Verify Signature – Update command for the session number 1, without Signature data in the payload	RQ.2.24.1 RQ.2.24.2 RQ.2.24.3
7	T ← IoT SAFE	Expected response: SW6A80h (Incorrect data).	RQ.2.24.15

4.2 Test Cases for Applet Type 2

4.2.1 Compute HKDF

4.2.1.1 Test Case 1

4.2.1.1.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract general mode is set, and the Hash algorithm is SHA-256

4.2.1.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x00 (extract general mode) • Secret equal to SECRET_DATA_KEY_16 • Salt equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.3.6.1 RQ.3.6.2 RQ.3.6.4 RQ.3.6.5 RQ.3.6.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_1	RQ.3.6.8

4.2.1.2 Test Case 2

4.2.1.2.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract general mode is set, and the Hash algorithm is SHA-384

4.2.1.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x00 (extract general mode) • Secret equal to SECRET_DATA_KEY_16 • Salt equal to SALT_DATA_48 • Hash algorithm equal 0x0002 (sha384)	RQ.3.6.1 RQ.3.6.2 RQ.3.6.4 RQ.3.6.5 RQ.3.6.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_2	RQ.3.6.8

4.2.1.3 Test Case 3

4.2.1.3.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract general mode is set, and the Hash algorithm is SHA-512

4.2.1.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x00 (extract general mode) • Secret equal to SECRET_DATA_KEY_16 • Salt length equal to SALT_DATA_64 • Hash algorithm equal 0x0004 (sha512)	RQ.3.6.1 RQ.3.6.2 RQ.3.6.4 RQ.3.6.5 RQ.3.6.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_3	RQ.3.6.8

4.2.1.4 Test Case 4

4.2.1.4.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract PSK-based mode is set, the Hash algorithm is SHA-256 and the search by Secret key ID is used

4.2.1.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x01 (extract PSK-based mode) • Tag 86 as specified in section 2.9.1.1.1 for Key ID = GSMA SEC KEY ID 01 • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.3.6.1 RQ.3.6.3 RQ.3.6.4 RQ.3.6.5 RQ.3.6.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_4	RQ.3.6.8

4.2.1.5 Test Case 5

4.2.1.5.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract PSK-based mode is set, the Hash algorithm is SHA-384 and the search by Secret key ID is used

4.2.1.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x01 (extract PSK-based mode) • Tag 86 as specified in section 2.9.1.1.1 for Key ID = GSMA SEC KEY ID 01 • Salt length equal to SALT_DATA_48 • Hash algorithm equal 0x0002 (sha384)	RQ.3.6.1 RQ.3.6.3 RQ.3.6.4 RQ.3.6.5 RQ.3.6.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_5	RQ3.6.8

4.2.1.6 Test Case 6

4.2.1.6.1 Test Case Description

Successful execution of Compute HKDF command to generate key material, when the extract PSK-based mode is set, the Hash algorithm is SHA-512 and the search by Secret key Label is used

4.2.1.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x01 (extract PSK-based mode) • Tag 76 as specified in section 2.9.1.1.1 for Key ID = GSMA SEC KEY ID 02 • Salt length equal to SALT_DATA_64 • Hash algorithm equal 0x0004 (sha512)	RQ.3.6.1 RQ.3.6.3 RQ.3.6.4 RQ.3.6.5 RQ.3.6.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_6	RQ3.6.8

4.2.1.7 Test Case 7

4.2.1.7.1 Test Case Description

Successful execution of Compute HKDF command to generate key material with the key length as small as possible when extract general mode is used.

4.2.1.7.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.7.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x00 (extract general mode) • Secret equal to SECRET_DATA_KEY_1 • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.3.6.1 RQ.3.6.2 RQ.3.6.4 RQ.3.6.5 RQ.3.6.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_HKDF_7	RQ.3.6.8

4.2.1.8 Test Case 8

4.2.1.8.1 Test Case Description

Compute HKDF command with an incorrect Salt length fails with an error status word.

4.2.1.8.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.8.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x00 (extract general mode) • Secret equal to SECRET_DATA_KEY_16 • Salt length equal to 16 (0x10) • Hash algorithm equal 0x0001 (sha256)	RQ.1.3.1 RQ.1.3.3 RQ.1.3.4 RQ.1.3.5 RQ.1.3.7
3	T ← IoT SAFE	• Execution failure with SW6A80h	RQ1.3.8

4.2.1.9 Test Case 9

4.2.1.9.1 Test Case Description

Compute HKDF command with an incorrect P1 fails with the defined status word.

4.2.1.9.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.9.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0xFF • Tag 86 as specified in section 2.9.1.1.1 for Key ID = GSMA SEC KEY ID 01 • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.3.6.1 RQ.3.6.3 RQ.3.6.4 RQ.3.6.5 RQ.3.6.7
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.3.6.6

4.2.1.10 Test Case 10

4.2.1.10.1 Test Case Description

The Compute HKDF command fails when the key is not granted for HKDF key derivation algorithm.

4.2.1.10.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.10.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x01 (extract PSK-based mode) • Tag 86 as specified in section 2.9.1.1.1 for Key ID = GSMA SEC KEY ID 03 • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.3.6.1 RQ.3.6.4 RQ.3.6.5 RQ.3.6.7
3	T ← IoT SAFE	Execution failure with SW6985h	RQ3.6.6

•

4.2.1.11 Test Case 11

4.2.1.11.1 Test Case Description

Check that Compute HKDF command fails when the Label is not present.

4.2.1.11.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.1.11.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute HKDF command with the following parameters: • P1 set to 0x01 (extract PSK-based mode) • Tag 86 with value: 0xFFFFFFFF • Salt length equal to SALT_DATA_32 • Hash algorithm equal 0x0001 (sha256)	RQ.3.6.1 RQ.3.6.4 RQ.3.6.5 RQ.3.6.7
3	T → IoT SAFE	Execution failure with SW6985h	RQ.3.6.6

4.2.2 Compute PRF

4.2.2.1 Test Case 1

4.2.2.1.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the general mode is set, the secrete length is equal to 16 bytes and the length result value is set to 8.

4.2.2.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x00 (general mode) • Secret equal to SECRET_DATA_PRF_KEY16 • Label and Seed equal to LABEL_AND_SEED_16 • Pseudo-random length set to 8	RQ.3.7.1 RQ.3.7.2 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_1	RQ.3.7.8

4.2.2.2 Test Case 2

4.2.2.2.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the general mode is set, the secrete length is equal to 20 bytes and the length result value is set to 32.

4.2.2.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x00 (general mode) • Secret equal to SECRET_DATA_PRF_KEY20 • Label and Seed equal to LABEL_AND_SEED_11 • Pseudo-random length set to 32	RQ.3.7.1 RQ.3.7.2 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_2	RQ.3.7.8

4.2.2.3 Test Case 3

4.2.2.3.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the general mode is set, the secrete length is equal to 64 bytes and the length of concatenation of the label and seed parameters is 123.

4.2.2.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x00 (general mode) • Secret equal to SECRET_DATA_PRF_KEY64 • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.3.7.1 RQ.3.7.2 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_2	RQ.3.7.8

4.2.2.4 Test Case 4

4.2.2.4.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-Plain pre-master secret mode is set, the key is searched by ID and the length result value is set to 8.

4.2.2.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x01 (PSK-Plain mode) • Tag 86 as specified in section 2.9.1.4 for ID: 'GSMA SEC KEY ID 03' • Label and Seed equal to LABEL_AND_SEED_16 • Pseudo-random length set to 8	RQ.3.7.1 RQ.3.7.3 RQ.3.7.5 RQ.3.7.7

3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_4	RQ.3.7.8

4.2.2.5 Test Case 5

4.2.2.5.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-Plain pre-master secret mode is set, the key is searched by Label and the length result value is set to 8.

4.2.2.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x01 (PSK-Plain mode) • Tag 76 as specified in section 2.9.1.4 for key ID: ‘GSMA SEC KEY ID 03’ • Label and Seed equal to LABEL_AND_SEED_16 • Pseudo-random length set to 8	RQ.3.7.1 RQ.3.7.3 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_4	RQ.3.7.8

4.2.2.6 Test Case 6

4.2.2.6.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-Plain pre-master secret mode is set, and the length of concatenation of the label and seed parameters is 123.

4.2.2.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x01 (PSK-Plain mode) • Tag 76 as specified in section 2.9.1.4 for key ID: ‘GSMA SEC KEY ID 03’ • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.3.7.1 RQ.3.7.3 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_6	RQ.3.7.8

4.2.2.7 Test Case 7

4.2.2.7.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-ECDHE pre-master secret mode is set, the key is searched by ID and the length result value is set to 8.

4.2.2.7.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.7.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x02 (PSK-ECDHE mode) • Tag 86 as specified in section 2.9.1.4 for key with ID: ‘GSMA SEC KEY ID 03’ • ECDH result equal to ECDH_COMP_RESULT • Label and Seed equal to LABEL_AND_SEED_16 • Pseudo-random length set to 8	RQ.3.7.1 RQ.3.7.4 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_7	RQ.3.7.8

4.2.2.8 Test Case 8

4.2.2.8.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-ECDHE pre-master secret mode is set, the key is searched by Label and the length result value is set to 8.

4.2.2.8.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.8.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x02 (PSK-ECDHE mode) • Tag 76 as specified in section 2.9.1.4 for key ID: ‘GSMA SEC KEY ID 03’ • ECDH result equal to ECDH_COMP_RESULT • Label and Seed equal to LABEL_AND_SEED_16 • Pseudo-random length set to 8	RQ.3.7.1 RQ.3.7.4 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_7	RQ.3.7.8

4.2.2.9 Test Case 9

4.2.2.9.1 Test Case Description

Successful execution of Compute PRF command to generate the Pseudo-random value, when the PSK-ECDHE pre-master secret mode is set, and the length of concatenation of the label and seed parameters is 123.

4.2.2.9.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.9.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x02 (PSK-ECDHE mode) • Tag 76 as specified in section 2.9.1.4 for key ID: 'GSMA SEC KEY ID 03' • ECDH result equal to ECDH_COMP_RESULT • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.3.7.1 RQ.3.7.4 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	The applet returns the pseudo-random value TEST_PRF_9	RQ.3.7.8

4.2.2.10 Test Case 10

4.2.2.10.1 Test Case Description

Check that Compute PRF command fails when the ID is not present.

.

4.2.2.10.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.10.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x01 • Tag 76 as specified in section 2.9.1.4 for key ID: 'GSMA SEC KEY ID 03' • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.3.7.1 RQ.3.7.3 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	Execution failure with SW6985h	RQ.3.7.6

4.2.2.11 Test Case 11

4.2.2.11.1 Test Case Description

Check that Compute PRF command fails with incorrect P1.

.

4.2.2.11.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.11.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0xFF • Tag 76 with value 0xFF • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.3.7.1 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.3.7.6

4.2.2.12 Test Case 12

4.2.2.12.1 Test Case Description

Check that Compute PRF command fails since the secret key is not granted with PRF SHA-256 key derivation algorithm.

4.2.2.12.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.2.12.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Compute PRF command with the following parameters: • P1 set to 0x01 • Tag 76 as specified in section 2.9.1.4 for key 'GSMA SEC KEY ID 01' • Label and Seed equal to LABEL_AND_SEED_123 • Pseudo-random length set to 32	RQ.3.7.1 RQ.3.7.3 RQ.3.7.5 RQ.3.7.7
3	T ← IoT SAFE	Execution failure with SW6985h	RQ.3.7.6

4.2.3 Get data – Application

4.2.3.1 Test Case 1

4.2.3.1.1 Test Case Description

Successful execution of Get Data – Application command to check the information about the applet and its capacity.

4.2.3.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.3.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Application command	RQ3.8.1 RQ3.8.3
3	T ← IoT SAFE	Expected response: A list of all the data field contained by the Applet and codified according section 2.9.1	RQ3.8.2 RQ3.8.4

4.2.3.2 Test Case 2

4.2.3.2.1 Test Case Description

Check that Get Data – Application command with an incorrect P1 fails with the defined status word.

4.2.3.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.3.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data - Application command with a P1 value as FFh.	RQ3.8.3
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ3.8.2

4.2.3.3 Test Case 3

4.2.3.3.1 Test Case Description

Check that Get Data – Application command with an incorrect P2 fails with the defined status word.

4.2.3.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.3.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Application command with a P2 as FFh	RQ3.8.3
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ3.8.2

4.2.3.4 Test Case 4

4.2.3.4.1 Test Case Description

Check that Get Data – Application command with wrong Le fails with the defined status word.

4.2.3.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.3.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Application with Le as FFh.	RQ3.8.3
3	T ← IoT SAFE	Execution failure with SW6700h	RQ3.8.2

4.2.4 Get data – File

4.2.4.1 Test Case 1

4.2.4.1.1 Test Case Description

Successful execution of Get Data – File command to check the File information of a specific file based on File ID/Label from the Applet Store. The target file to be used is the File 1 according to the Applet Store Default Configuration NIST keys section (2.8.1.1.1) or File 2 from Brainpool section (2.8.1.1.2) or File 1 from Combined (NIST and Brainpool) section (2.8.1.1.3) or File 1 from Secret Key section (2.9.1.1.1).

4.2.4.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.4.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – File command sequence using File ID values from all files present within configurations defined in the initial conditions section.	RQ.3.9.1 RQ.3.9.3
3	T ← IoT SAFE	Expected response: A list of File information structure of the targeted File ID contained by the Applet Store and codified according [1] section 2.14.4.	RQ 3.9.2 RQ.3.9.4
4	T → IoT SAFE	Send a Get Data – File command sequence using File Label values from all files present within configurations defined in the initial conditions section.	RQ.3.9.1 RQ.3.9.3

5	T ← IoT SAFE	Expected response: A list of File information structure of the targeted File Label contained by the Applet Store and codified according [1] section 2.14.4.	RQ 3.9.2 RQ.3.9.4

4.2.4.2 Test Case 2

4.2.4.2.1 Test Case Description

Check that Get Data – File command with an incorrect P1 fails with the defined status word.

4.2.4.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.4.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – File command with incorrect P1 value = FF	RQ.3.9.1
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ 3.9.2

4.2.4.3 Test Case 3

4.2.4.3.1 Test Case Description

Check that Get Data – File command with an incorrect P2 fails with the defined status word.

4.2.4.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.4.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – File command with an incorrect P2 value = FF	RQ 3.9.1
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.3.9.2

4.2.4.4 Test Case 4

4.2.4.4.1 Test Case Description

Check that Get Data – File command with a non-existent File ID/Label with the defined status word.

4.2.4.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.4.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – File command sequence using File ID "GSMA FID 03"	RQ.3.9.1
3	T ← IoT SAFE	Execution failure with SW6A82h	RQ.3.9.2
4	T → IoT SAFE	Send a Get Data – File command sequence using File ID "GSMA FLB 03"	RQ.3.9.1
5	T ← IoT SAFE	Execution failure with SW6A82h	RQ 3.9.2

4.2.4.5 Test Case 5

4.2.4.5.1 Test Case Description

Check that Get Data – File command with Incorrect data with the defined status word.

4.2.4.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.4.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – File command with an incorrect tag value in the data field. DataIn= FFh.	RQ.3.9.1
3	T ← IoT SAFE	Execution failure with SW6A80h	RQ.3.9.2

4.2.5 Get data – Object List

4.2.5.1 Test Case 1

4.2.5.1.1 Test Case Description

Successful execution of Get Data – Object List command to check the object list into the Applet Store.

4.2.5.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.5.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Object List command sequence	RQ.3.10.1 RQ.3.10.2 RQ.3.10.3 RQ.3.10.4
3	T ← IoT SAFE	Expected response: A list of all the objects (and their attributes) contained by the Applet Store and codified according [1] section 2.14.4.	RQ.3.10.5 RQ.3.10.6

4.2.5.2 Test Case 2

4.2.5.2.1 Test Case Description

Check that Get Data – Object List command with an incorrect P1 fails with the defined status word.

4.2.5.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.5.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data command with a incorrect P1 value = FF	RQ.3.10.1 RQ.3.10.4
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.3.10.6

4.2.5.3 Test Case 3

4.2.5.3.1 Test Case Description

Check that Get Data – Object List command with an incorrect P2 fails with the defined status word.

4.2.5.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.5.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Object List command with an incorrect P2 value = FF	RQ.3.10.1 RQ.3.10.4
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.3.10.6

4.2.5.4 Test Case 4

4.2.5.4.1 Test Case Description

Check that Get Data – Object List command first command with P2 indicating More Outgoing Data fails with the defined status word.

4.2.5.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.5.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – Object List command with More Outgoing Data mode (P2 = 01h) for the first APDU.	RQ.3.10.1 RQ.3.10.4
3	T ← IoT SAFE	Execution failure with SW6A86h	RQ.3.10.6

4.2.6 Get data – Secret key information

4.2.6.1 Test Case 1

4.2.6.1.1 Test Case Description

Successful execution of Get Data – secret key information command to check the secret key information into the Applet Store, when the search by Secret key ID is used

4.2.6.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.6.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – secret key information command with value of Tag 86 as specified in section 2.9.1.1.1 for key ID = GSMA SEC KEY ID 01	RQ3.11.1
3	T ← IoT SAFE	Expected response: Command response as reported in 2.17.4.1 of [1] ,with the values section 2.9.1.1.1 for key ID = GSMA SEC KEY ID 01	RQ3.11.1

4.2.6.2 Test Case 2

4.2.6.2.1 Test Case Description

Successful execution of Get Data – secret key information command to check the secret key information into the Applet Store, when the search by Secret key Label is used

4.2.6.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.6.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – secret key information command sequence with value of Tag 76 as specified in section 2.9.1.1.1 for key ID = GSMA SEC KEY ID 01	RQ3.11.1

3	T ← IoT SAFE	Expected response: Command response as reported in 2.17.4.1 of [1] ,with the values section 2.9.1.1.1 for key ID = GSMA SEC KEY ID 01	RQ3.11.1

4.2.6.3 Test Case 3

4.2.6.3.1 Test Case Description

Successful execution of Get Data – secret key information command to check the secret key information into the Applet Store when the search by Secret key Label is used and the secret key does not have a label.

4.2.6.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.6.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – secret key information command with value of Tag 86 as specified in section 2.9.1.1.1 for key ID = GSMA SEC KEY ID 02	RQ3.11.1
3	T ← IoT SAFE	Expected response: Command response as reported in 2.17.4.1 ,with the values section 2.9.1.1.1 for key ID = GSMA SEC KEY ID 02	RQ3.11.1

4.2.6.4 Test Case 4

4.2.6.4.1 Test Case Description

Check that Get Data – Get Data – secret key information command fails when the Label is not present.

4.2.6.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.6.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – secret key information command with value of Tag 86 = 3030303030303030	RQ3.11.1
3	T ← IoT SAFE	Execution failure with SW6985h	RQ3.11.1

4.2.6.5 Test Case 5

4.2.6.5.1 Test Case Description

Check that Get Data – Get Data – secret key information command fails when the ID is not present.

4.2.6.5.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.6.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Data – secret key information command with value of Tag 76 = 3030303030303030	RQ3.11.1
3	T ← IoT SAFE	Execution failure with SW6985h	RQ3.11.1

4.2.7 Get Random

4.2.7.1 Test Case 1

4.2.7.1.1 Test Case Description

Successful execution of Get Random command with expected length of 256 bytes and get a random number back.

4.2.7.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.7.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Random command with expected length of 256 bytes	RQ3.12.1 RQ3.12.3
3	T ← IoT SAFE	Expected response: Random number of 256 bytes	RQ3.12.2 RQ3.12.4

4.2.7.2 Test Case 2

4.2.7.2.1 Test Case Description

Successful execution of Get Random command with expected length of 20 byets and get a random number back.

4.2.7.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.7.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Get Random command with expected length of 20 bytes	RQ3.12.1 RQ3.12.3
3	T ← IoT SAFE	Expected response: Random number of 20 bytes	RQ3.12.2 RQ3.12.4

4.2.7.3 Test Case 3

4.2.7.3.1 Test Case Description

Execution of Get Random command failed with SW6A86h.

4.2.7.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.7.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair P1 as FFh	RQ3.12.3
3	T ← IoT SAFE	SW6A86h	RQ3.12.2

4.2.7.4 Test Case 4

4.2.7.4.1 Test Case Description

Execution of Get Random command failed with SW6A86h.

4.2.7.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.7.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Generate key pair P2 as FFh	RQ3.12.3
3	T ← IoT SAFE	SW6A86h	RQ3.12.2

4.2.8 Read file

4.2.8.1 Test Case 1

4.2.8.1.1 Test Case Description

Successful execution of Read file command with expected data.

4.2.8.1.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.8.1.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	

2	T → IoT SAFE	Execute get data file with value of Tag 83 as specified in section 2.8.1.1.1 for file1 to retrieve the size of file. Send a Read File command with value of Tag 83 as specified in section 2.8.1.1.1 for file1	RQ3.13.1 RQ3.13.3
3	T ← IoT SAFE	Response SHOULD contain data as mentioned in File1	RQ3.13.2 RQ3.13.4

4.2.8.2 Test Case 2

4.2.8.2.1 Test Case Description

Execution of Read file command failed with expected SW.

4.2.8.2.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.8.2.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Read File command with value of Tag 83 as specified in section 2.8.1.1.1 for file1 and offset outside file data	RQ3.13.3
3	T ← IoT SAFE	SW6981h	RQ3.13.2

4.2.8.3 Test Case 3

4.2.8.3.1 Test Case Description

Execution of Read File command failed with expected SW in case ‘Read’ is not granted as object access condition.

4.2.8.3.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.8.3.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Read File command with value of Tag 83 as specified in section 2.8.1.1.1 for File 1_a.	RQ3.13.3
3	T ← IoT SAFE	SW6985h	RQ3.13.2

4.2.8.4 Test Case 4

4.2.8.4.1 Test Case Description

Execution of Read file command failed due to a ‘Deactivated’ file state, with expected SW.

4.2.8.4.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.8.4.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Read File command with value of Tag 83 as specified in section 2.8.1.1.1 for File 1_b.	RQ3.13.3
3	T ← IoT SAFE	SW6985h	RQ3.13.2

4.2.8.5 Test Case 5

4.2.8.5.1 Test Case Description

Execution of Read file command failed due to incorrect tag value with expected SW.

4.2.8.5.2 Initial Conditions

Default initial conditions as defined in section 5.1.1 corresponding to the relevant configuration.

4.2.8.5.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send Read File command with Incorrect tag value as FFh	RQ3.13.3
3	T ← IoT SAFE	SW6A80h	RQ3.13.2

4.2.8.6 Test Case 6

4.2.8.6.1 Test Case Description

Execution of Read file command failed due to file is non-existent in the applet store with expected SW.

4.2.8.6.2 Initial Conditions

Applet installed according with configuration(s) referenced inside the Applet Type 2 applicability table under the section 2.2.2.

4.2.8.6.3 Test Procedure

Step	Direction	Description	RQ
1	T → IoT SAFE	Select Instance	
2	T → IoT SAFE	Send a Read File command with value of Tag 83 as FFFF.FF.	RQ3.13.3

3	T ← IoT SAFE	SW6A82h	RQ3.13.2

5 Data values used in Test Cases

This section shows the hexa hardcoded values to be used in the test cases that requires them.

DATA_1

2923BE84E16CD6AE529049F1F1BBE9EBB3A6DB3C870C3E99245E0D1C06B747DEB312
4DC843BB8BA61F035A7D0938251F5DD4CBFC96F5453B130D890A1CDBAE32209A50EE
407836FD124932F69E7D49DCAD4F14F2444066D06BC430B7323BA122F622919DE18B1F
DAB0CA9902B9729D492C807EC599D5E980B2EAC9CC53BF67D6

HASH

EC0A89849484E8B9A988E10AD448D65DCA39D48CC823404C362E08896D7B3364

DATA_2

2923BE84E16CD6AE529049F1F1BBE9EBB3A6DB3C870C3E99245E0D1C06B747DEB312
4DC843BB8BA61F035A7D0938251F5DD4CBFC96F5453B130D890A1CDBAE32209A50EE
407836FD124932F69E7D49DCAD4F14F2444066D06BC430B7323BA122F622919DE18B1F
DAB0CA9902B9729D492C807EC599D5E980B2EAC9CC53BF67D6BF14D67E2DDC8E668
3EF574961FF698F61CDD11E9D9C167272E61DF0844F4A7702D7E8392C53CBC9121E33
749E0CF4D5D49FD4A4597E35CF3222F4CCCFD3902D48D38F75E6D91D2AE5C0F72B78
8187440E5F5000D4618DBE7B0515073B33821F187092DA6454CEB1853E6915F8466A049
6730ED9162F6768D4F74A4AD0576876FA16BB11ADAE248879FE52DB2543E53CF445D3
D828CE0BF5C560593D97278A59762DD0C2C9CD68D4496A792508614014B13B6AA5112
8C18CD6A90B87978C2FF1151D9A95C19BE1C07EE9A89AA786C2B554BF9AE7D923D15
5903828D1D96CA1665E4EE1309CFED9719FE2A5E20C9BB44765382A4689A982797A767
8C263B126DF

DATA_3

89AFC39D41D3B327814B80940B042590F96556EC91E6AE7939BCE31F3A18BF2B

SIGNATURE_1

Generated using KeyPair (GSMA KPLB 01) and DATA_1:

NIST-256r1:
6B17D1F2E12C4247F8BCE6E563A440F277037D812DEB33A0F4A13945D898C2963D3FD
758AA446847EB1100ABCA4ED4D428C897280763201397B1D0683F28B223

Brainpool-256r1:
3044022008923B8A5504B262AE7B9DE4B6F8E34F68C520CD45C2EC53BF39959437A79B
AE022005CF0A3BEB15ED9C78F5E0D4D463CD48FB43A0265495F4640EFFAE4371568C8
D

SIGNATURE_2

Generated using KeyPair (GSMA KPLB 01) and DATA_2:

NIST-256r1:
6B17D1F2E12C4247F8BCE6E563A440F277037D812DEB33A0F4A13945D898C29610957
69228E4E84122AF901D56E7F6492D9D4A9FEEAE8D10DF36B72D7FE56645

Brainpool-256r1:
3044022062B45E8FB99E633A08D7D2F8B7FFDF3546DD1751CD6511B472D5552F81187
566022009134184B27CA8C490AAC1154ADF65249597B4E7073E055F957BEEF2F211018
8

SECRET_DATA_KEY_1
41
SECRET_DATA_KEY_16
31323334353637383930313233343536
SECRET_DATA_KEY_32
3132333435363738393031323334353631323334353637383930313233343536
SECRET_DATA_KEY_64
31323334353637383930313233343536313233343536373839303132333435363132333435
363738393031323334353631323334353637383930313233343536

SALT_DATA_32
2E4D41E091D194F6358759CB2EA92C5CCF5B74116411324811C5E0B05EDB7814
SALT_DATA_48
AC2E4D41E091D194F6358759CB2EA92C5CCF5B74116411324811C5E0B05EDB782A971
DA578892667EC0CC6F1C01A0E5C
SALT_DATA_64
AC2E4D41E091D194F6358759CB2EA92C5CCF5B74116411324811C5E0B05EDB782A971
DA578892667EC0CC6F1C01A0E5C346D7AC11AB0F24531CD0B1B505A8F2D
TEST_HKDF _1
07AB09CD3ED863A15FF04DFB2A8D840C3506EADA76A52A9DFC8D347ED2B4D977
TEST_HKDF_2
432CC769F879F5DA88AF7CE3F4F2CE81A5C82082E5FEF2D00CA6B7D7B1D4CE75706C
3799892700758C4EE6C7E904C86F
TEST_HKDF _3
FB19326566DBA7BC0C815BB56FCA9F3A658D19FA6497253F58E776C6330AF02CA6671
BC995B3A9292E7D68C6371C680C9E708C6C7941DB8615A2241DEFEC9D83
TEST_HKDF _4
EA437DAF319C1DDF4684D8B44D11DD72D0DDA0D6FEF4A0F8464E512DC22D292E
TEST_HKDF _5
2B8F4C10F0B91BFB4F5BE29663F611E91D0D2B43FE1294D7F9B1F8E3AB415284CAD51
FFF94324D7C7A1E8D3D62A7A4B9
TEST_HKDF _6
33BD7BC0424DE7A0DB9D27650CA8247F6CAD159778AADD2D3AF70EAB6B823A5D459
2C47FE4EFCC8A39BA9F8C7C696C359059CEDA5200CC9B54C819DB1E88D29D
TEST_HKDF _7
BAA5D03730ED06F143E9A5367D1744E7E4AE79219504F5C8DE6B9F54B734CC41

```
SECRET_DATA_PRF_KEY16
  0102030405060708090A0B0C0D0E0F10
SECRET_DATA_PRF_KEY20
  0102030405060708090A0B0C0D0E0F1011121314
SECRET_DATA_PRF_KEY64
  0102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F2000AABB
  CCDDEEFF112233445566778899ABCDEF12345678903132333435363738
ECDH_COMP_RESULT
  0102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F20
LABEL_AND_SEED_11
  3132333435363738393041
LABEL_AND_SEED_16
  31323334353637383930414243444546
LABEL_AND_SEED_123
  0102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F2000AABB
  CCDDEEFF112233445566778899ABCDEF12345678903132333435363738010203040506O
  708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F2000AABBCCDDEEFF1122
  33445566778899ABCDEF1234567890313233
TEST_PRF_1
  65ED3375972F3685
TEST_PRF_2
  33B1113E6A4C5492B516FFF6A5828150FA3C3BBAFEAFE93DBE79D9AB8DD8B8FB
TEST_PRF_3
  730C54FA353C2BAE30D0BEF540854CBCFD271A9F3FD762FB1BA5FE1C372472CF
TEST_PRF_4
  4917B95553E035DA
TEST_PRF_6
  8C61A13ACD1D2AD3E2FFEF9E8918BB7FB8594AFA7793C90798B4E8E063514382
TEST_PRF_7
  8B8ECCD0681C79B8
TEST_PRF_8
  9462700FA83E8D5969A81AF085321620DB7DE3DCB0BA03C628381279045C3FA28
RSA_PUB_ID_01
  0001
ECC_PUB_ID_02
GSMA PKID 02
ECC_PUB_ID_09_NONEXISTENT
  0009
PUBLIC_KEY_VALUE_001_RSA
  Modulus                                                                =
  8A86DF0EA5F87639DE771BFB09DBAE62A9CC10343A340F1929F98CAD03DFEDED7BE
  9B0DDECF62F5A3ED26D33FD2C2AE6B5659799D17AE8743F21048F91973C035310131A
  73D399F25F8F77AD265A70E08B19FEBBCF0C958939F8C715D9E6DA29E4F9426FE8B24
  71B9034EED386851F0D8F24A281E750CB52EBB65C8A1F7BC1427ECEDE6817A3428782
  6B41A761E4D3E2CE61865C2CA0C4B581064CF1956307D55217450741443AA4771E7F30
  CB5856ABE6824585EE7D9A3D45291ABF4FDB23B32628EFD54AB194DEE5C3A93CA034
  8F302CCD23921814E7570247E05D82DF26FBABBCB8217B1C082BB6E79A23426FE218A
  246255E59349CE22A7F834858177ABB
  Exponent =  010001

PUBLIC_KEY_VALUE_002_ECC
```

```
  83C5FD58C7A626E2E13A8F71C7973B135CBE4FF330FC5654C238031F315C46421ED27
  49656C9E6F107A74224A0BBC6BD70D07FDFEBE493D97A9CF167CF991E16”
  KEY PAIR 01
  KP01_Pub_KEY_VALUE = DFFFFFDFF
  KP01_PRIV_KEY_VALUE_ = DFDFDF
PUBLIC_KEY_VALUE_001_RSA_INVALID_LEN
  Modulus                                                              =
  8A86DF0EA5F87639DE771BFB09DBAE62A9CC10343A340F1929F98CAD03DFEDED7BE
  9B0DDECF62F5A3ED26D33FD2C2AE6B5659799D17AE8743F21048F91973C035310131A
  73D399F25F8F77AD265A70E08B19FEBBCF0C958939F8C715D9E6DA29E4F9426FE8B24
  71B9034EED386851F0D8F24A281E750CB52EBB65C8A1F7BC1427ECEDE6817A3428782
  6B41A761E4D3E2CE61865C2CA0C4B581064CF1956307D55217450741443AA4771E7F30
  CB5856ABE6824585EE7D9A3D45291ABF4FDB23B32628EFD54AB194DEE5C3A93CA034
  8F302CCD23921814E7570247E05D82DF26FBABBCB8217B1C082BB6E79A23426FE218A
  246255E59349CE22A7F834858177ABBFFFFFFFFFFFFFFFFFFFE
  Exponent =  010001
PUBLIC_KEY_VALUE_002_ECC_INVALID_LEN
  83C5FD58C7A626E2E13A8F71C7973B135CBE4FF330FC5654C238031F315C46421ED27
  49656C9E6F107A74224A0BBC6BD70D07FDFEBE493D97A9CF167CF991E16FFFFFFFFF
  FFFFFFFFFFE
X509_CERT
  4D4949447A7A4343417265674177494241674955594157786F5746556548545135324C486
  B335A56515873505632637744515 94A4B6F5A496876634E4151454C0A42514177647A454
  C4D416B474131554542684D4352565578446A414D42674E564241674D42564E305958526
  C4D51307743775944565151484441524461585235 0A4D5177774367594456515 14B44414E
  555130457845544150426 74E564241734D43464E31596B6479623356774D5177774367594
  45651514444414E30593245780A476A415942676B71686B6947397730424351455743 3352
  6A595542305932457559323974 4D4234584454497A4D5441774E4445334D6A59794E6C6F
  58445449300A4D5441774D7A45334D6A59794E6C6F77647A454C4D416B4741315545426
  84D4352565578446A414D42674E564241674D42564E305958526C4D513077437759440A5
  65151484441524461585235 4D5177774367594456515 14B44414E555130457845544150 42
  674E564241734D43464E31596B6479623356774D51777743675944 0A565151444414E305
  9324578476A415942676B71686B69473977304243514557433 3526A595542305932457559
  3239744D494942496A414E42676B71686B69470A397730424151454641414F43415138414
  D49494243674B4341514541723175527030656B456F6B7543754A6E4D676A784F63784A5
  7706874694C62724F2B6C4F0A48635066326B44596E2B4B734A61584B327645416162567
  17A456A69394E7A594B65473553625531755 34B3755564E43437442785170722B516C773
  1427237350A684F437332736C51365245456F6D77717250536E665A6A747A2B4D4573584
  F3830415341677562415044654334786351486665514 26B44567754754C475674470A7143
  6363614B727367726E4E447A59616547586546393638 4F6A76535731353155397444 7A2F5
  14647734C2B3543335A307568664A526F74376E6B71586B4A730A4F2B6B7434715273697
  875636256736E70425862796E626952715152765743493 47563304A5051414F3344475658
  4D306D664C6F41616B656847717531436C360A2B7750415856676270536F654737685752
  44387A426E76663370464D4773647774726B486F7133784D3955524551726966774944415
  141426F314D77555441640A42674E56485134454667515539483965497646466F58444363
  775769316455786B7849786E573077487759445652306A42426777466F415539483965497
  646460A6F584443637757693164557 86B7849786E5730774477594456523054415148 2F42
  415577417745422F7A414E42676B71686B69473977304241517346414 14F430A415145415
  47379767A48564B4632685A525A76634744777A2F51525A78412B7838717545364D42477
  8634D63464C6A793363415A37517569723631487339 5A490A6F72386632717843526857 69
  2B586F6D3936494B4D346D5230635A332F614C6E6A2F527A615335766249616 94638477
  0393274585A2B6D6347522F4B4D5578440A55326973552B64674152327471354 16C686D7
```

```
35936374C67554D78344D795266345947726B5A4276756F7979377859732B74494B32417
7414541453851767036 0A67526B6D54497231456D69384A5341536A70433431792F307738
6B634F48344E684849323879702B2B5A455468416F6765627A6F5252583749746278 39395
1730A684B4E793856335A48654B615A2B4A5277376971564467642F4D307A53486F55497
1452B5A5071395358425635716854573247585847557 15A303036694F4C630A7975307A47
3465585874477A2F39396A336A6B796C79733373513D3D
```

Annex A Initial state (Normative)

The IOT SAFE applet SHALL be loaded under the ISD/ISD-R on the UICC/eUICC.

The IOT SAFE Applet vendor SHALL provide scripts for the installation and the personalization of the IOT SAFE Applet in .ldr format of the TCA Loader. The following Secure Channel Protocols MAY be used in these scripts: SCP03, SCP80, SCP81.

Annex B Document Management (Informative)

B.1 Document History

Version	Date	CR	Approval Authority	Editor / Company
V1.0	27/06/2024	First PRD version after transfer from TCA	IoT SAFE	Gloria Trujillo, GSMA (Editor)

Other Information

Type	Description
Document Owner	IoT SAFE Group
Editor / Company	Gloria Trujillo, 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

Your comments or suggestions & questions are always welcome.