GLOBALPLATFORM®
THE STANDARD FOR SECURE DIGITAL SERVICES AND DEVICES

GlobalPlatform Technology

Confidential Card Content Management

Card Specification v2.3 – Amendment A

Version 1.1.1.12 (to be published as v1.2)

Public Review

May 2019

Document Reference: GPC_SPE_007

Contents

Figures

Tables

1 Introduction

This document defines a mechanism for an Application Provider to confidentially manage its application; i.e. to load, install, and personalize using a third party communication network. This third party shall not be able to access the clear text of any confidential data and code belonging to the Application Provider. The network may or may not provide its own additional security and confidentiality means. The entity representing this link could be a Link Platform Operator (LPO) such as an Over-The-Air platform operator compliant with ETSI specification TS 102 225 [102 225] and TS 102 226 [102 226], e.g. a Mobile Network Operator.

This document describes:

- Several scenarios that may be used to securely initialize new Secure Channel keys for Security Domains;
- A Controlling Authority Security Domain (CASD) that shall be installed and personalized to enable such scenarios;
- The STORE DATA commands and DGIs that shall be used in such scenarios;
- A mechanism to load an Executable Load File as a Ciphered Load File Data Block;
- A mechanism for Applications to request a CASD signature;
- The APIs supporting the listed mechanisms / scenarios.

1.1 Audience

This specification is intended for card manufacturers and application developers developing GlobalPlatform card implementations.

1.2 IPR Disclaimer

Attention is drawn to the possibility that some of the elements of this GlobalPlatform specification or other work product may be the subject of intellectual property rights (IPR) held by GlobalPlatform members or others. For additional information regarding any such IPR that have been brought to the attention of GlobalPlatform, please visit https://globalplatform.org/specifications/ip-disclaimers/. GlobalPlatform shall not be held responsible for identifying any or all such IPR, and takes no position concerning the possible existence or the evidence, validity, or scope of any such IPR.

1.3 References

Table 1-1: Normative References

Standard / Specification	Description	Ref
GlobalPlatform Card Specification v2.3	GlobalPlatform Technology Card Specification v2.3	[GPCS]
GlobalPlatform Java Card API	Java Card API and Export File for Card Specification (org.globalplatform) v1.7 (or higher)	[GPC API]
BSI TR-03111, Version 1.11	BSI Technical Guideline TR-03111: Elliptic Curve Cryptography	[TR-03111]

Standard / Specification	Description	Ref
EMV Book 2 Security and Key Management v4.2	EMV Integrated Circuit Card Specifications for Payment Systems – Book 2 Security and Key Management v4.2 June 2008	[EMV Book 2]
ETSI TS 102 225 (Release 6)	Smart cards; Secured packet structure for UICC based applications, European Telecommunications Standards Institute Project Smart Card Platform (EP SCP), 2004	[102 225]
ETSI TS 102 226 (Release 6)	Smart cards; Remote APDU structure for UICC based applications, European Telecommunications Standards Institute Project Smart Card Platform (EP SCP), 2004	[102 226]
FIPS PUB 186-4	Federal Information Processing Standards Publication 186-4: Digital Signature Standard (DSS)	[FIPS 186-4]
ISO/IEC 7816-4:2005	Identification cards – Integrated circuit(s) cards – Part 4: Organization, security and commands for interchange	[ISO 7816-4]
ISO/IEC 9796-2:2010	Information technology – Security techniques – Digital signature schemes giving message recovery – Part 2: Integer factorization based mechanisms	[ISO 9796-2]
ISO/IEC 9797-1:2011	Information technology – Security techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms using a block cipher	[ISO 9797-1]
ITU-T X.680	Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation	[ASN.1]
ITU-T X.690	Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)	[ASN.1 Encoding]
NIST SP 800-38B	Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, May 2005	[NIST 800-38B]
NIST SP 800-56A Revision 1	Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography, March 2007	[NIST 800-56A]
PKCS#1	PKCS #1 v2.1: RSA Cryptography Standard, *RSA Laboratories*, June 14, 2002	[PKCS#1]
RFC 4514 (X.501)	Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names – RFC 4514	[X.501]
RFC 5280 (X.509)	Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile – RFC 5280	[X.509]
RFC 5639	Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation	[RFC 5639]
RFC 5480	Elliptic Curve Cryptography Subject Public Key Information	[RFC 5480]

Table 1-2: Informative References

Standard / Specification	Description	Ref
Java Card API	Application Programming Interface, Java Card™ Platform, v3.0.1 Classic Edition, Sun Microsystems, Inc., May 2009	[JCAPI]
Java Card VM	Virtual Machine Specification, Java Card™ Platform, v3.0.1 Classic Edition, Sun Microsystems, Inc., May 2009	[JCVM]
Java Card JCRE	Runtime Environment Specification, Java Card™ Platform, v3.0.1 Classic Edition, Sun Microsystems, Inc., May 2009	[JCRE]

1.4 Terminology and Definitions

Terms used in this document are defined in [GPCS].

1.5 Abbreviations and Notations

Table 1-3: Abbreviations and Notations

Abbreviation / Notation	Meaning
AES	Advanced Encryption Standard
AP	Application Provider
APDU	Application Protocol Data Unit
API	Application Programming Interface
APSD	Application Provider Security Domain
AUT	AUThentication
BCD	Binary Coded Decimal
CA	Controlling Authority
CASD	Controlling Authority Security Domain
CDH	Cofactor Diffie-Hellman
CERT.AP.AUT	Application Provider Certificate holding a PK.AP.AUT
CERT.AP.CT	Application Provider Certificate holding a PK.AP.CT
CERT.AP.ECKA	Application Provider Certificate holding a PK.AP.ECKA
CERT.CASD.AUT	CASD Certificate holding a PK.CASD.AUT
CERT.CASD.CT	CASD Certificate holding a PK.CASD.CT
CERT.CASD.ECKA	CASD Certificate holding a PK.CASD.ECKA
CERT.CASD-SIGN.AUT	CASD Certificate holding a PK.CASD-SIGN.AUT
CRT	Control Reference Template
CT	Confidential Template

Abbreviation / Notation	Meaning
DAP	Directory Access Protocol
DEK	Data Encryption Key
DES	Data Encryption Standard
DGI	Data Grouping Identifier
DR	Derivation Random
ECC	Elliptic Curve Cryptography
ECKA	Elliptic Curve based Key Agreement
ENC	Encryption
ePK.AP.ECKA	Ephemeral AP Public Key used for ECKA
ePK.SD.ECKA	Ephemeral SD Public Key used for ECKA
eSK.AP.ECKA	Ephemeral AP Private Key used for ECKA
eSK.SD.ECKA	Ephemeral SD Private Key used for ECKA
KAT	Key Agreement Template
KIc	SCP80 Ciphering key
KID	Key Identifier
KS	Secret Key of a symmetric scheme
KS.AP.CT	Application Provider Symmetric Key used for Encryption/Decryption
KS.CASD.AUT	CASD Symmetric Key used for Signature/Verification
KS.CASD.CT	CASD Symmetric Key used for Encryption/Decryption
KVN	Key Version Number
LPO	Link Platform Operator
MAC	Message Authentication Code
MOC	Mandatory, Optional, Conditional
OID	Object Identifier
OTA	Over-The-Air
PK	Public Key of an asymmetric key pair.
PK.AP.AUT	Application Provider Public Key used for Signature Verification
PK.AP.CT	Application Provider Public Key used for Encryption
PK.AP.ECKA	Application Provider Public Key used for ECKA
PK.CA.AUT	CA Public Key used to verify certificates
PK.CA.ECDSA	CA Public Key used to verify ECDSA certificates
PK.CASD.AUT	CASD Public Key used for Signature Verification
PK.CASD.CT	CASD Public Key used for Encryption
PK.CASD.ECKA	CASD Public Key used for ECKA

Abbreviation / Notation	Meaning
PK.CASD-SIGN.AUT	CASD Public Key used for ECDSA Signature Verification
RGK	Randomly Generated Key
RSA	Rivest / Shamir / Adleman asymmetric algorithm
ShS	Shared Secret
ShSee	Shared Secret calculated from two ephemeral keys
ShSss	Shared Secret calculated from two static keys
SK	Private key of an asymmetric key pair
SK.AP.AUT	Application Provider Private Key used for Signature Creation
SK.AP.ECKA	Application Provider Private Key used for ECKA
SK.CA.AUT	CA Private Key used to sign certificates (off-card)
SK.CASD.AUT	CASD Private Key used for Signature
SK.CASD.CT	CASD Private Key used for Decryption
SK.CASD.ECKA	CASD Private Key used for ECKA
SK.CASD-SIGN.AUT	CASD Private Key used for ECDSA Signature

1.6 Revision History

GlobalPlatform technical documents numbered *n*.0 are major releases. Those numbered *n*.1, *n*.2, etc., are minor releases where changes typically introduce supplementary items that do not impact backward compatibility or interoperability of the specifications. Those numbered *n.n*.1, *n.n*.2, etc., are maintenance releases that incorporate errata and precisions; all non-trivial changes are indicated, often with revision marks.

Table 1-4: Revision History

Date	Version	Description
October 2007	v1.0	Initial release
January 2011	v1.0.1	Incorporates the contents of “Errata and Precisions for GlobalPlatform Card Specification Amendment A v1.0” as well as errata and precisions made during the development of the UICC Configuration v1.0 and v1.0.1. The contents of sections 4.8 and 4.9 are now integrated into GlobalPlatform Card Specification v2.2.1.

Date	Version	Description
November 2015	v1.1	• The entire structure of the document has been reviewed and modified. • Added new introductory section “Overview Of Personalization Models” with flow charts from UICC Configuration v1.0.1 and Amendment E v1.0.1 • Separated confidential Security Domain Key personalization into sections: o 4.3 Preparation of Controlling Authority Security Domain o 4.5 Establishment of Application Provider Secure Channel Keys • Deprecated DGIs '80AE' and '80DE' • Integrated Key Agreement Model from Amendment E v1.0.1 • Clarified in section 4.5.3 “Scenario #2: Push Model” that DGI '8010' may be sent in the same command APDU as DGI '00A6' or in the directly following command. • Added DGI '00A6' content description for Secure Channel Keys of different type and/or length • Renamed section 4.8 “Ciphered Load File” to section 4.6 “Ciphered Load File with Optional ICV” and replaced by content of section 4.3 “Encrypted Load File with optional ICV” of Amendment E • Moved the following sections to [GPCS]: o DGI for Personalizing Security Domain Keys o Introduction of a Token Using Symmetric Keys • Clarified that the Pull Model only retrieves the Randomly Generated Key (RGK). The algorithm for derivation of Key Set keys from the RGK has been integrated into the section describing the Pull Model. • Clarified that the Push Model transmits all the keys of a Key Set. • Extended scenarios 1, 2.A, and 2.B to RSA keys longer than 1024 bits using digital signature Scheme 2 defined in [ISO 9796-2] and RSAES-OAEP encryption scheme defined in [PKCS#1]. • Integrated the definition of the Forwarded CASD Data mechanism, previously described in Amendment C.

Date	Version	Description
September 2018	v1.1.1	• Changed wrong acronym KS.APSD.CT into KS.AP.CT. • In section 3.3.2, clarified that Push and Pull models are using Certificate with Message Recovery and Key Agreement model is using Certificate without Message Recovery. • In section 3.3.2, clarified that tags 'B0' and 'F0' are invalid ASN.1 tag but won't be changed for backward compatibility reasons. • In section 3.3.2, clarified the encoding of tag '42' in CASD certificates. Recommendation regarding usage of an Authority Key Identifier. • In section 3.3.2, within KVN '74', assigned a range of KID values reserved for proprietary usage. • In section 3.4.1, clarified that AP certificates (scenarios #1 and #2.A) shall be generated using the Certificate with Message Recovery format. Other formats are deprecated in this version. • In section 3.4.1, removed Table 3-11 that was just an example and is now obsolete. • In section 3.5, clarified that RSAES-OAEP shall be used for RSA ciphering with RSA keys longer than 1024 bits. • In section 3.5, clarified the encoding of some length fields when the encoded length can be 128 or greater. • In section 3.5.1, Table 3-18, reserved bit 4 of Scenario #3 parameters for GSMA as requested in 17ESIMWI4271_Doc_002r1_LS. • In section 3.5.2, clarified encryption algorithm to be used with KS.CASD.CT and KS.AP.CT. • In section 3.5.2, clarified signature algorithm to be used with KS.CASD.AUT. • Fixed a few typos.
December 2018	v1.1.1.4	Committee Review
February 2019	v1.1.1.6	Member Review
May 2019	v1.1.1.12	Public Review
TBD	v1.2	Public Release • New section 5 defines a new Controlling Authority Signature service to applications. • New sections 3.2.4 and 3.5.5 define a new Confidential Key Loading Model, Scenario #4, using Elliptic Curve cryptography but not requiring the pre-establishment of a secure channel. New section 6 to define the new INITIALIZE SECURITY command required for Scenario #4. • Clarifications of the configuration of the CASD capabilities for Scenario #1 and Scenario #2 in Tables 3-2 and 3-3.

2 Use Cases and Requirements

This document proposes a specification addendum to support the following requirements:

- An Application Provider shall be able to load confidential applications using an untrusted transport link to protect its assets.
- A Link Platform Operator (LPO) shall be able to create an on-card component to load and manage confidential applications and transfer the ownership to an Application Provider.
- An Application Provider shall be able to instantiate its own applications and to personalize them.

Security Domains are responsible for the on-card application management and therefore are the candidates to realize those requirements. Thus the Application Provider uses a Security Domain (APSD) to manage confidential loading:

- The keys used by the APSD shall not be known by the LPO. To achieve this, an on-card controlling entity (extended role of a Controlling Authority) shall be able to secure the key creation of the APSD.
- This controlling entity shall be able to secure the APSD personalization.
- The APSD shall be personalized using the LPO network.

Additionally it is required that:

- It shall be possible to support confidential loading on non-PK cards.
- The specification modifications will not require modifications to the current loading mechanism (format of commands remain unchanged) to avoid modifying existing OTA infrastructure.
- It shall be possible to support AP Application personalization. The content of the data being personalized is beyond the scope of this specification.

3 Confidential Personalization of Secure Channel Keys

This section defines the APDU commands and the Data Grouping Identifiers to be submitted to the card in order to personalize the Controlling Authority and the Application Provider Security Domains.

3.1 Controlling Authority Security Domain

To enable the confidential loading of an initial Secure Channel Key Set for newly created Security Domains, a Controlling Authority Security Domain (CASD) is introduced which provides the related basic cryptographic function. In addition this CASD may or may not provide control over Card Content management through the Mandated DAP Verification as defined in GlobalPlatform Card Specification [GPCS].

Only a single CASD instance providing services for the confidential establishment of (initial) Secure Channel Key Sets may be installed on the card.

Personalization of the CASD certificates and related cryptographic keys shall be performed following procedures preserving the confidentiality of such credentials with respect to the Card Issuer and any actor represented on the card.

The AIDs for the CASD shall be as follows:

- Executable Load file (package) AID: 'A0 00 00 01 51 53 50 43 41 53 44'
- Executable Module (applet) AID: 'A0 00 00 01 51 53 50 43 41 53 44 00'
- Application AID: 'A0 00 00 01 51 53 50 43 41 53 44 00'

The CASD Executable Load File is associated with the Issuer Security Domain.

The installation of a CASD instance shall fail if attempted when the card is in the SECURED life cycle state.

3.2 Overview of Personalization Models

Three main schemes are defined to personalize the Application Provider Security Domain:

1. Pull Model (Scenario #1): Security Domain keys are generated on-card and then returned to the Application Provider. An asymmetric and a symmetric cryptographic scheme are defined.
2. Push Model (Scenario #2): Security Domain keys are sent to the Application Provider Security Domain protected by asymmetric cryptography. Two variants are defined: with Application Provider Certificate (#2A) and without Application Provider Certificate (#2B).
3. Key Agreement Model (Scenario #3): Security Domain keys are generated on-card and off-card using the Elliptic Curve Key Agreement scheme described in NIST SP 800-56A [NIST 800-56A] as "(Cofactor) One-Pass Diffie-Hellman, C (1e, 1s, ECC CDH)".
4. Key Agreement Model (Scenario #4): Security Domain keys are generated on-card and off-card using the Elliptic Curve Key Agreement scheme described in [NIST 800-56A] as "(Cofactor) Full Unified Model, C(2e, 2s, ECC CDH)", using a mechanism that does not require the use of a secure channel to ensure the identity of the parties.

The above scenarios (#1, #2A, #2B, #3 and #4) may all be enabled concurrently on the same card, except for the symmetric and asymmetric variants of scenario #1 which are mutually exclusive.

3.2.1 Pull Model

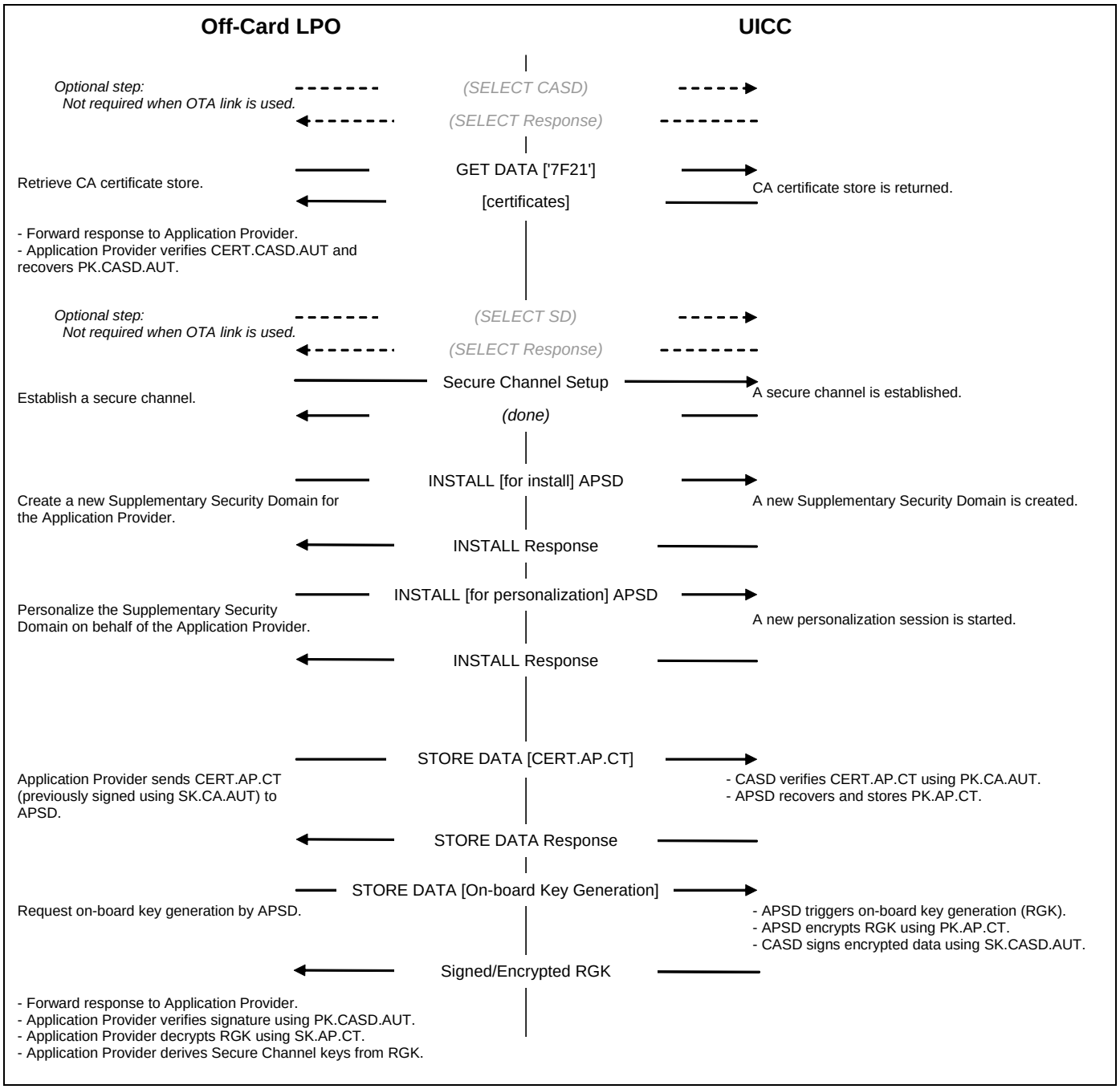
In the Pull Model the APSD keys are generated on-card and retrieved by the AP. The model supports the use of asymmetric and symmetric keys.

3.2.1.1 Asymmetric Key Mode

The CASD is personalized using the Certificate CERT.CASD.AUT (containing a Public Key PK.CASD.AUT) and a Private Key SK.CASD.AUT. An Application Provider Certificate (CERT.AP.CT), signed with the CA Private Key SK.CA.AUT, is sent to and verified by the CASD using the CA Public Key PK.CA.AUT. In response to an on-card key generation request, the card provides the Application Provider with a Randomly Generated Key (RGK) encrypted with the AP Public Key (PK.AP.CT) and signed by the CASD Private Key (SK.CASD.AUT). The RGK is used on-card and off-card to derive the three APSD Secure Channel keys.

This scenario can only be used to set up either 16-byte DES keys or 128-bit AES keys. To set up 128-bit AES keys (e.g. for Secure Channel Protocol '03') using this scenario, it is recommended to use a PK/SK.CASD.AUT with a length of 3072 bits or more.

Figure 3-1: Scenario #1 (Pull Model), Using PK Scheme

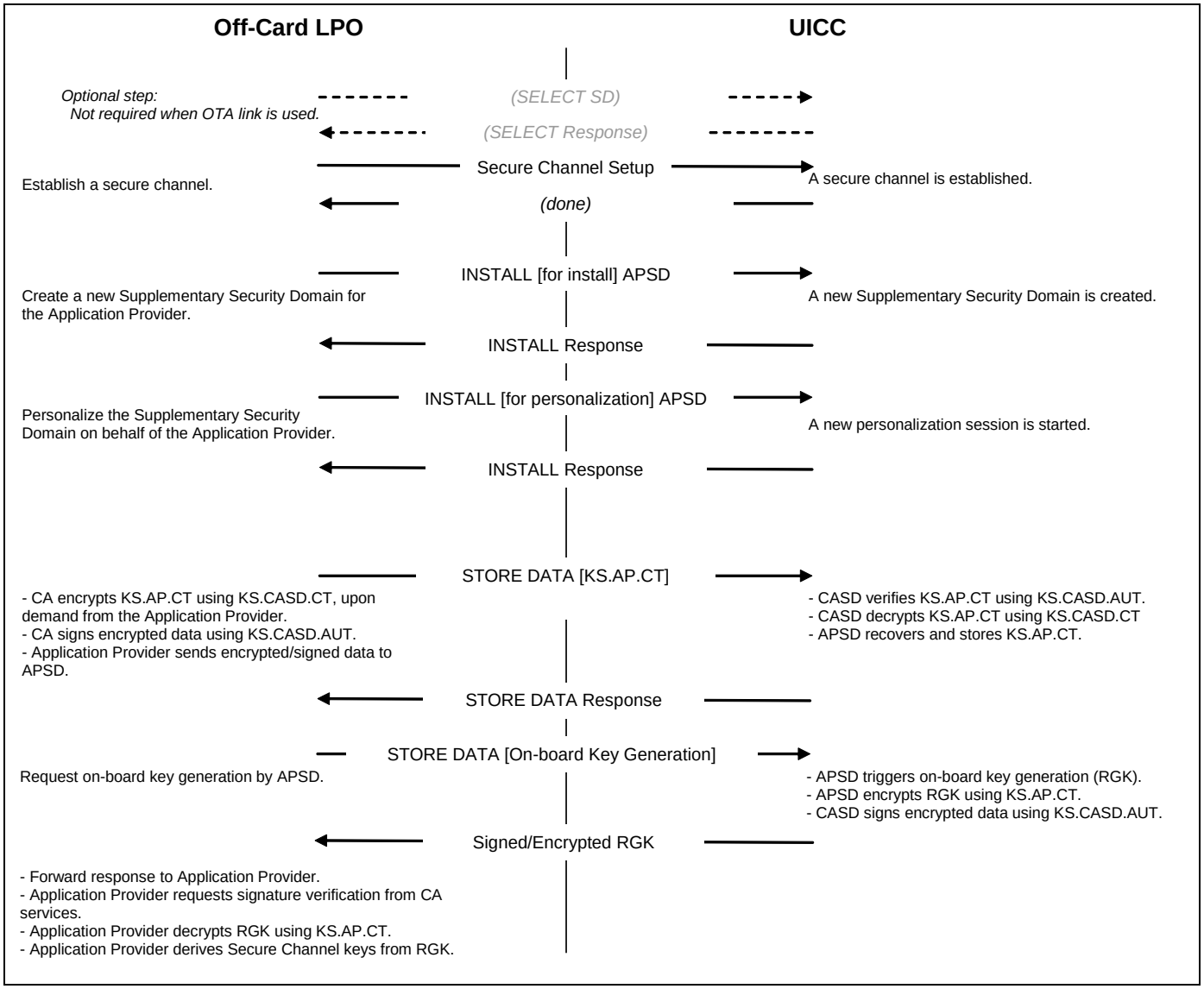


NOTE: In the figure above, the two STORE DATA commands sent to APSD shall be sent in the same personalization session, i.e. the first STORE DATA command shall indicate P1.b8=0 and P2='00', and the second one shall indicate P1.b8=1 (last block) and P2='01'.

3.2.1.2 Symmetric Key Mode

The CASD is personalized with two symmetric keys, an encryption key (KS.CASD.CT) and a signature key (KS.CASD.AUT). Before initiating the on-card key generation, an Application Provider secret encryption key KS.AP.CT is sent to the card encrypted with the CASD symmetric encryption key KS.CASD.CT and signed by the CASD symmetric signature key KS.CASD.AUT. Then, in response to the on-card key generation request, the Randomly Generated Key (RGK) master key is returned to the Application Provider encrypted by KS.AP.CT and signed with KS.CASD.AUT. The RGK is used on-card and off-card to derive the three APSD Secure Channel keys.

Figure 3-2: Scenario #1 (Pull Model), Using Non-PK Scheme



NOTE: In the figure above, the two STORE DATA commands sent to APSD shall be sent in the same personalization session, i.e. the first STORE DATA command shall indicate P1.b8=0 and P2='00', and the second one shall indicate P1.b8=1 (last block) and P2='01'.

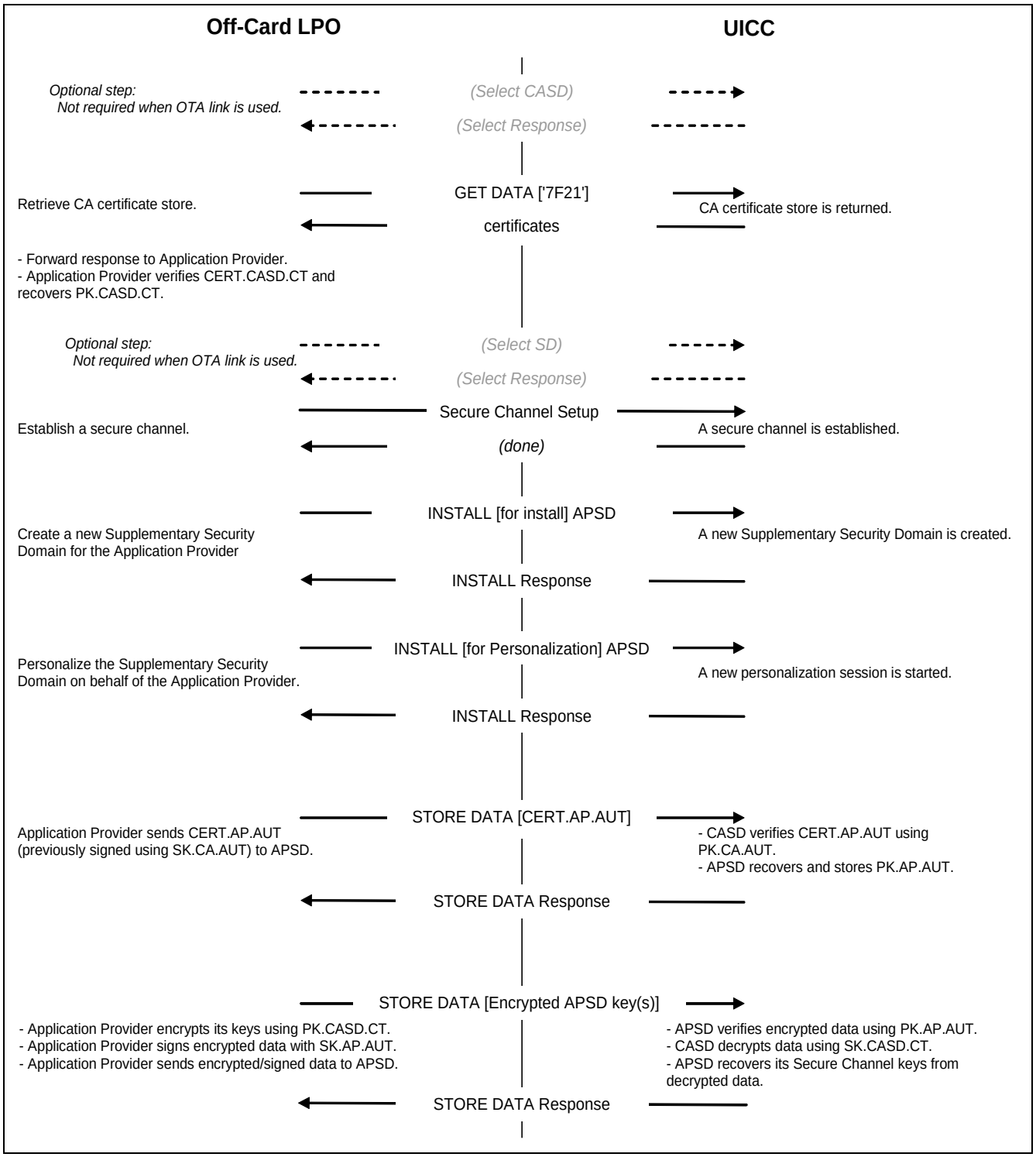
3.2.2 Push Model

In the Push Model the APSD keys are generated off-card and 'pushed' to the card. Two different personalization scenarios are supported, Push Model with and without Application Provider Certificate.

3.2.2.1 Push Model with Application Provider Certificate (Scenario #2.A)

The CASD is personalized with a certificate CERT.CASD.CT (containing the Public Key PK.CASD.CT), the related Private Key SK.CASD.CT, and the Public Key PK.CA.AUT. An Application Provider Certificate (CERT.AP.AUT) signed with the CA Private Key (SK.CA.AUT) is sent to card and verified by the CASD using its Public Key PK.CA.AUT. Subsequently the APSD keys are sent to the card encrypted with the CASD Public Key PK.CASD.CT and signed by the APSD Private Key SK.AP.AUT. The APSD verifies the signature using PK.AP.AUT extracted from the certificate CERT.AP.AUT. Upon successful verification and following the decryption of the keys using the CASD key recovery service, the APSD accepts the new Secure Channel keys.

Figure 3-3: Scenario #2.A (Push Model with Application Provider Certificate)

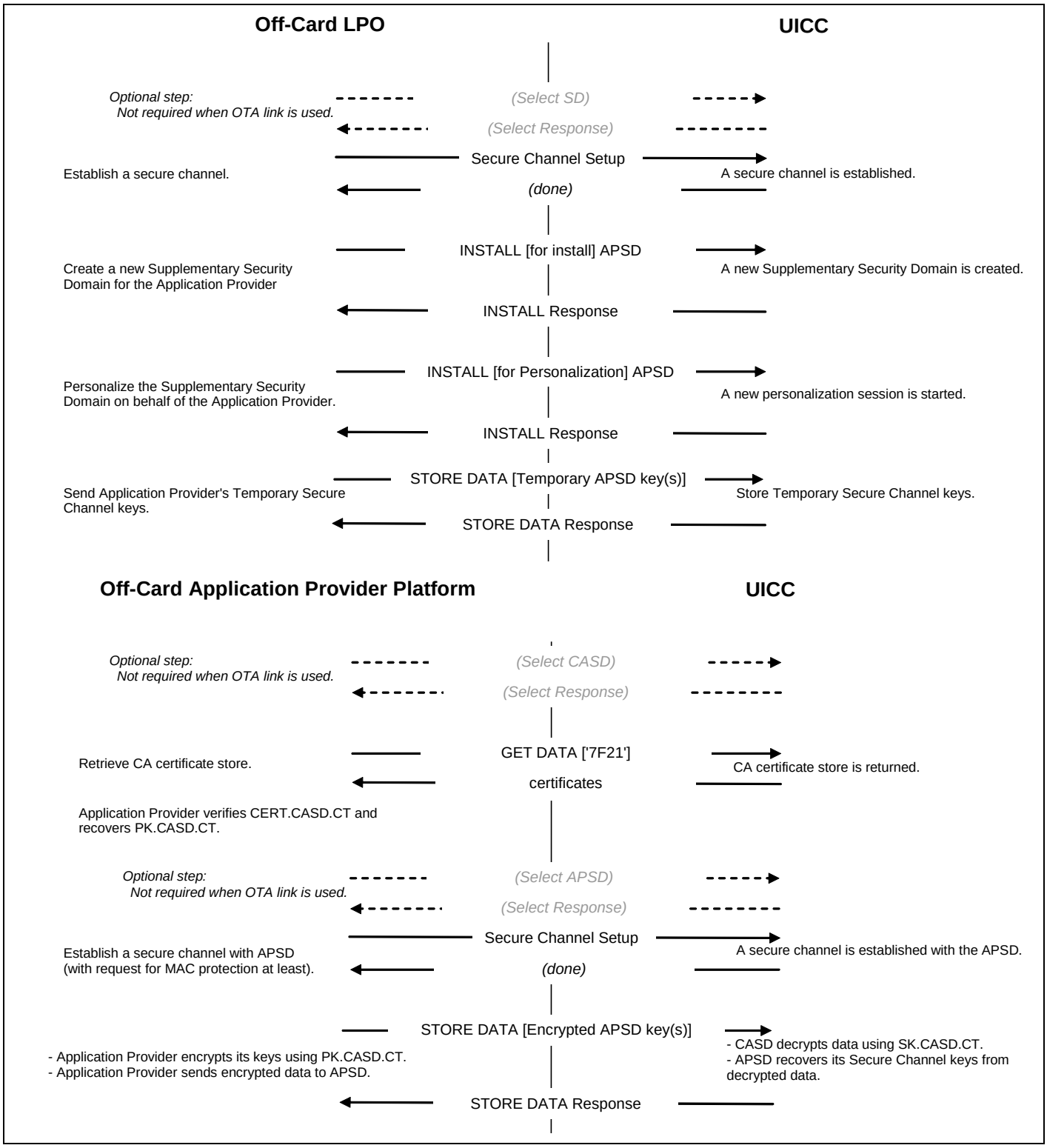


NOTE: In the figure above, the two STORE DATA commands sent to APSD shall be sent in the same personalization session, i.e. the first STORE DATA command shall indicate P1.b8=0 and P2='00', and the second one shall indicate P1.b8=1 (last block) and P2='01'.

3.2.2.2 Push Model without Application Provider Certificate (Scenario #2.B)

The CASD is personalized with a certificate CERT.CASD.CT (containing Public Key PK.CASD.CT) and the related Private Key SK.CASD.CT. The APSD is loaded with temporary Secure Channel keys on behalf of the AP, e.g. by an LPO. The new APSD keys are sent to the card encrypted by the CASD Public Key (PK.CASD.CT) and additionally secured (encrypted and/or signed) using the temporary APSD Secure Channel keys. When the security checks succeed, the temporary Secure Channel keys are replaced by the received keys.

Figure 3-4: Scenario #2.B (Push Model without Application Provider Certificate)

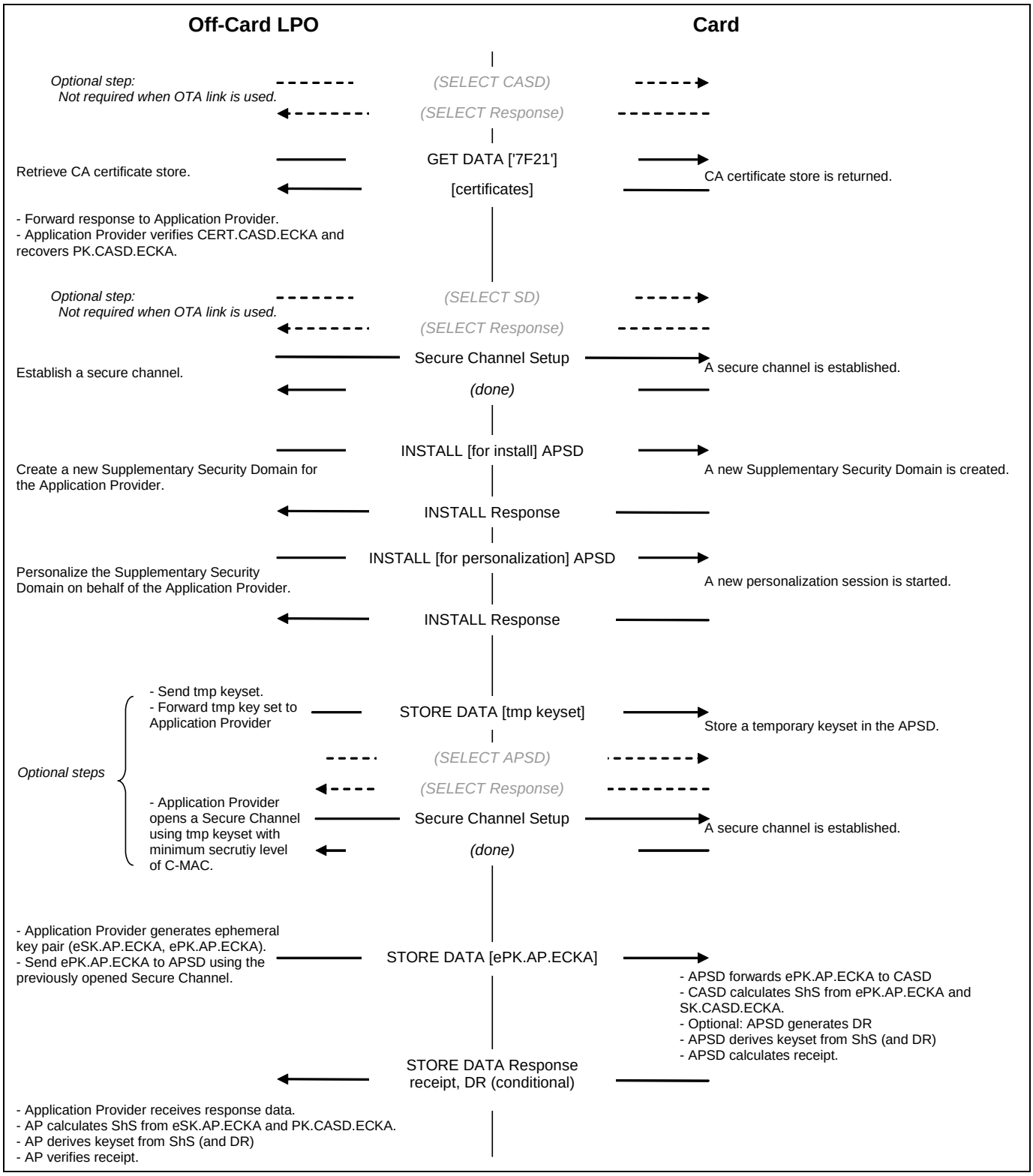


3.2.3 Key Agreement Model

The CASD is personalized with a secret key SK.CASD.ECKA to be used with the (Cofactor) One-Pass Diffie-Hellman, C(1e, 1s, ECC CDH) scheme as specified in [NIST 800-56A] and with the corresponding certificate CERT.CASD.ECKA. Using a static key pair in the CASD allows the response from the card to be authenticated. The AP uses an ephemeral key pair (eSK.AP.ECKA / ePK.AP.ECKA).

To set up the key set, the STORE DATA command can either be sent directly to the APSD or can be sent to the associated SD when preceded by an INSTALL [for personalization]. A secure channel with at least MAC protection is required to allow for an authentication of each command's origin. A derivation random (DR) may be used by the CASD to enforce the randomness of the new Secure Channel keys.

Figure 3-5: Scenario #3 (Using Key Agreement)



3.2.4 Key Agreement with no Secure Channel

The CASD is personalized with a public key PK.CA.ECDSA, which is used to verify the public key certificate provided by the AP, CERT.AP.ECKA, extracting the AP's static public key agreement key PK.AP.ECKA.

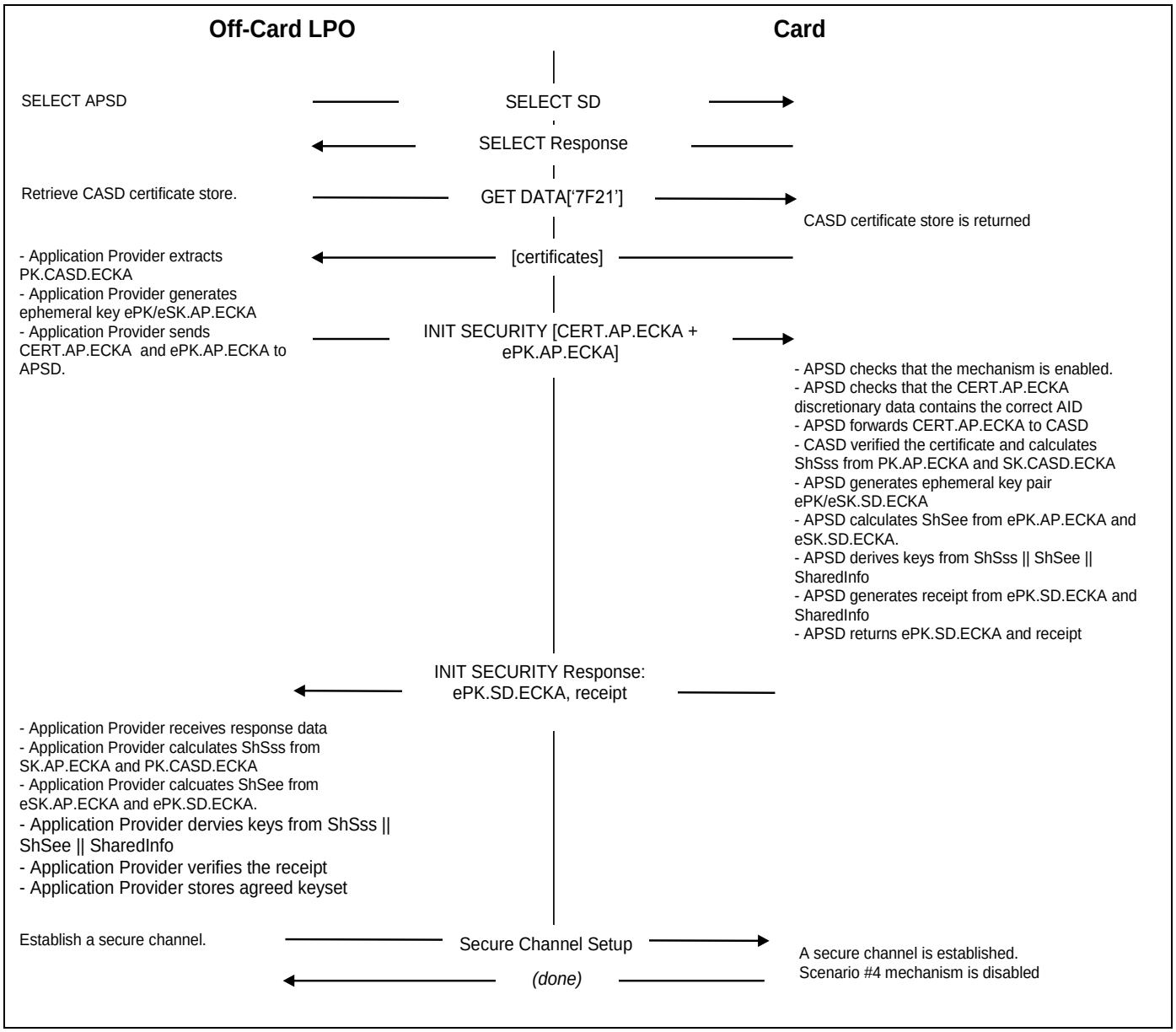
The CASD is also personalized with a private key SK.CASD.ECKA (to be used with the (Cofactor) Full Unified Model C(2e, 2s, ECC CDH) scheme as specified in [NIST 800-56A]), and with the corresponding certificate CERT.CASD.ECKA. The private key SK.CASD.ECKA is used in conjunction with PK.AP.ECKA to derive a static shared secret ShSss.

The AP generates its ephemeral key, and sends the public key, ePK.AP.ECKA, to the APSD with the static key certificate.

The APSD also generates its own ephemeral key pair (eSK.SD.ECKA / ePK.SD.ECKA) and uses eSK.SD.ECKA in conjunction with ePK.AP.ECKA to derive another shared secret ShSee.

This scenario is to be used when the APSD to be personalized does not have the capability to establish a secure channel, therefore the APSD processes an INITIALIZE SECURITY command sent with no secure messaging. The mechanism is only to be made available while no secure session keys have been personalized. The APSD is assured that the key agreement process was concluded successfully the first time a Secure Channel session is successfully established between the AP and the APSD using the derived keys. At this point the mechanism is disabled; subsequent personalization of keys may be performed using the conventional PUT KEY and STORE DATA commands as defined in [GPCS]. On the other hand, as long as no Secure Channel session has been successfully established between the AP and the APSD, this scenario may be restarted and previously generated keys (if any) shall be discarded.

Figure 3-6: Scenario #4 (Key Agreement with No Secure Channel)



3.3 Preparation of Controlling Authority Security Domain

This section defines the commands and data to be submitted to prepare the Controlling Authority Security Domain (CASD) for the support of the confidential key establishment of new Application Provider Security Domains.

3.3.1 CASD Data

The STORE DATA command shall be used in conjunction with DGI '0070' to personalize the Security Domain Data (tag '66') as described in [GPCS] section 11.11.2.3. The data elements to be supported and their content are defined in the relevant Configuration Specifications.

For the definition of the SD Recognition Data elements, see [GPCS] sections H.2 and H.3.

The GET DATA command shall be used with tag '66' (as described in [GPCS] sections H.2 and H.3) to retrieve CASD Data that shall contain CASD Capability Information as follows:

Table 3-1: CASD Data

Tag	Length	Data / Description	Presence
'66'	1	Security Domain Data	Mandatory
'73'	1	Security Domain Management Data	Mandatory
'06'	7	'2A 8648 86FC6B 01'	Mandatory
'60'	11	SD Management Type & Version	Mandatory
'06'	9	'2A 8648 86FC6B 02 02 02'	Mandatory
'63'	9	SD Identification Scheme	Mandatory
'06'	7	'2A 8648 86FC6B 03'	Mandatory
'64'	11	SD Secure Channel Protocol & Options	Mandatory
…	…	…	…
'65'	11	SD Configuration Details	Mandatory
'06'	9	'2A 8648 86FC6B 05' **+ CASD Capability Information**	Mandatory

The CASD shall be personalized so that CASD Capability Information and CASD credentials remain consistent and reflect the scenarios actually supported by the CASD. The CASD is not responsible for checking the consistency of its Capability Information.

CASD Capability Information is encoded on two bytes as shown in Table 3-2 and Table 3-3.

Table 3-2: CASD Capability Information – Byte 1

b8	b7	b6	b5	b4	b3	b2	b1	Description
-	-	-	-	-	-	-	1	Scenario #1 using Triple DES or RSA1024 (as specified by B2.b1)
-	-	-	-	-	-	1	-	Scenario #2.A using RSA1024
-	-	-	-	-	1	-	-	Scenario #2.B using RSA1024
-	-	-	-	1	-	-	-	Scenario #3 using ECC-256
-	-	-	1	-	-	-	-	Scenario #3 using ECC-384
-	-	1	-	-	-	-	-	Scenario #3 using ECC-512
-	1	-	-	-	-	-	-	Scenario #3 using ECC-521
1	-	-	-	-	-	-	-	`AuthoritySignature` interface

Table 3-3: CASD Capability Information – Byte 2

b8	b7	b6	b5	b4	b3	b2	b1	Description
-	-	-	-	-	-	-	x	This bit is only relevant if B1.b1 is set to 1; else it shall be set to 0
-	-	-	-	-	-	-	0	Scenario #1 using Triple DES with 16-byte key length
-	-	-	-	-	-	-	1	Scenario #1 using RSA1024
-	-	-	-	-	-	1	-	Scenario #1 using RSA keys longer than 1024 bits
-	-	-	-	-	1	-	-	Scenario #2.A using RSA keys longer than 1024 bits
-	-	-	-	1	-	-	-	Scenario #2.B using RSA keys longer than 1024 bits
x	x	x	x	-	-	-	-	RFU (0)
-	-	-	1	-	-	-	-	Scenario #4 using ECC-256
-	-	1	-	-	-	-	-	Scenario #4 using ECC-384
-	1	-	-	-	-	-	-	Scenario #4 using ECC-512
1								Scenario #4 using ECC-521

Additional bytes may be appended in the future.

3.3.2 CASD Certificates and Asymmetric Keys

The following types of certificates shall be stored in the CASD depending on the supported scenarios:

- **Pull Model, asymmetric (Scenario #1):**

  Certificate CERT.CASD.AUT containing the public key PK.CASD.AUT used by the Application Provider to verify the CASD signature over the on-board generated key material

- **Push Model (Scenario #2.A):**

  Certificate CERT.CASD.CT containing the public key PK.CASD.CT used by the Application Provider to encrypt the key material pushed to the APSD

- **Push Model (Scenario #2.B):**

  Certificate CERT.CASD.CT containing the public key PK.CASD.CT used by the Application Provider to encrypt the key material pushed to the APSD

- **Key Agreement Models ECKA-EG (Scenario #3 and Scenario #4):**

  Certificate CERT.CASD.ECKA containing the public key PK.CASD.ECKA used by the Application Provider to calculate the Shared Secret (ShS or ShSss)

The Data Group Identifier '7F21' shall be used with the STORE DATA command for storing a set of certificates. The content of DGI '7F21' is one (or more) certificate(s) coded in TLV format with a tag value of '7F21', e.g. if several scenarios are supported or if chains of certificates are supported. This set of certificates is known as the CASD certificate store. It is only possible to update all CASD certificates at once, i.e. it is not possible to update or delete a single certificate within the CASD certificate store. It shall be possible to retrieve the entire content of the CASD certificate store using the GET DATA command (P1P2 = '7F21').

Table 3-4 describes the certificate format based on digital signature with message recovery that shall be used for use with the Push and Pull Model.

Table 3-4: Certificate Structure for CERT.CASD.CT/AUT (Certificate with Message Recovery)

Tag	Length	Description	Presence
'7F21'	Variable	Certificate	Mandatory
'93'	1-16	Certificate Serial Number	Mandatory
'42'	Variable	CA Identifier	Mandatory
'5F20'	1-16	Subject Identifier	Mandatory
'95'	1	Key Usage ('82' digital signature verification, or '88' encipherment for confidentiality; see [GPCS] Table 11-17)	Mandatory
'5F25'	4	Effective Date (YYYYMMDD, BCD format)	Optional
'5F24'	4	Expiration Date (YYYYMMDD, BCD format)	Mandatory
'45'	Variable	CASD Image Number	Mandatory
'53'	1-127	Discretionary Data (unspecified format)	Optional
'73'	1-127	Discretionary Data (BER-TLV encoded)	Optional
'5F37'	Variable	Signature	Mandatory
'5F38'	Variable	Public Key Modulus Remainder	Mandatory

The CA Identifier (tag '42') identifies the CA and its value shall match the Subject field of the parent certificate. If the parent certificate is formatted as described in RFC 5280 ([X.509]), then the Subject field should include the CN (common name) attribute (see RFC 4514 – [X.501]).

The CASD Image Number (tag '45') uniquely identifies the CASD according to the CA numbering scheme and should match the value of tag '45' stored in the Data Store of the CASD (that can be retrieved with a GET DATA command).

The Subject Identifier (tag '5F20') uniquely identifies the owner of the public key that is certified. The owner might be the Secure Element itself (i.e. for a CASD certificate, the owner is the CASD hosted by the Secure Element).

Optional tags '53' and '73' are mutually exclusive. Tag '73' is intended for BER-TLV encoded discretionary data.

If the CA manages more than one key pair to sign certificates, it shall include an Authority Key Identifier field (tag 'C9') within Discretionary Data (tag '73') to further distinguish between these different key pairs. Such an Authority Key Identifier may be a digest (e.g. SHA-256) of the CA Public Key that shall be retrieved and used to verify the CASD certificate.

Table 3-5 describes the (TLV-encoded) data to be signed using the CA Private Key (SK.CA.AUT, private counterpart of PK.CA.AUT) to generate a Certificate with message recovery:

Table 3-5: Data Signed to Generate the Signature of CERT.CASD.CT/AUT

Tag	Length	Description	Presence
'7F49'	Variable	Public Key	Mandatory
'82'	1 or 3	Public Key Exponent	Mandatory
'81'	Variable	Public Key Modulus	Mandatory
'93'	1-16	Certificate Serial Number	Mandatory
'42'	Variable	CA Identifier	Mandatory
'5F20'	1-16	Subject Identifier	Mandatory
'5F25'	4	Effective Date (YYYYMMDD, BCD format)	Conditional
'5F24'	4	Expiration Date (YYYYMMDD, BCD format)	Mandatory
'95'	1	Key Usage ('82' digital signature verification, or '88' encipherment for confidentiality see [GPCS] Table 11-17)	Mandatory
'45'	Variable	CA Security Domain Image Number	Mandatory
'53'	1-127	Discretionary Data (unspecified format)	Conditional
'73'	1-127	Discretionary Data (BER-TLV encoded)	Conditional

The data listed in Table 3-5 shall be signed by the CA using the signature algorithm described in ISO/IEC 9796 ([ISO 9796-2]). Scheme 1[1] (with trailer option 1 and SHA-1 digest) shall be used if SK.CA.AUT is a 1024-bit RSA key. Scheme 2 (with trailer option 1 and SHA-256 digest) shall be used if SK.CA.AUT is an RSA key longer than 1024 bits, using the alternative signature production function defined in [ISO 9796-2] Annex B.6.

The data recovered by decrypting the signature (tag '5F37'), as described in [ISO 9796-2], includes part of tag '7F49', including tag '82' (Public Key Exponent) and part of tag '81' (Public Key Modulus leftmost bytes). Public Key Modulus rightmost bytes (Remainder) are retrieved from tag '5F38'. To verify the signature, the implementation shall rebuild the data described in Table 3-5 using the data recovered from the signature, the value part of tag '5F38', and the clear-text data provided in the certificate.

[1] A simplified description of this scheme can be found in [EMV Book 2].

When the Key Agreement Model (with or without Secure Channel) is used, the CASD certificate shall use the following format:

Table 3-6: Certificate Structure for CERT.CASD.ECKA (Certificate without Message Recovery)

Tag	Length	Description	Presence
'7F21'	Variable	Certificate	Mandatory
'93'	1-16	Certificate Serial Number	Mandatory
'42'	Variable	CA Identifier	Mandatory
'5F20'	1-16	Subject Identifier	Mandatory
'95'	2	Key Usage ('00 80' key agreement) see [GPCS] section 11.1.9	Mandatory
'5F25'	4	Effective Date (YYYYMMDD, BCD format)	Optional
'5F24'	4	Expiration Date (YYYYMMDD, BCD format)	Mandatory
'45'	Variable	CASD Image Number	Mandatory
'53'	1-127	Discretionary Data (unspecified format)	Optional
'73'	1-127	Discretionary Data (BER-TLV encoded)	Optional
'7F49'	Variable	Public Key data object (see Table 3-8)	Mandatory
'5F37'	Variable	Signature	Mandatory

The following data shall be signed using the ECDSA algorithm (see [GPCS] section B.4.3) to generate the signature of the CASD certificate (tag '5F37'):

Table 3-7: Data Signed to Generate the Signature of CERT.CASD.ECKA

Tag	Length	Description	Presence
'93'	1-16	Certificate Serial Number	Mandatory
'42'	Variable	CA Identifier	Mandatory
'5F20'	1-16	Subject Identifier	Mandatory
'95'	2	Key Usage	Mandatory
'5F25'	4	Effective Date (YYYYMMDD, BCD format)	Conditional
'5F24'	4	Expiration Date (YYYYMMDD, BCD format)	Mandatory
'45'	Variable	CA Security Domain Image Number	Mandatory
'53'	1-127	Discretionary Data (unspecified format)	Conditional
'73'	1-127	Discretionary Data (BER-TLV encoded)	Conditional
'7F49'	Variable	Public Key data object (see Table 3-8)	Mandatory

The format of the ECC Public Key data object (tag '7F49') is described in the following table:

Table 3-8: Public Key Data Object Data Field

Tag	Length	Description	Presence
'7F49'	Variable	Public Key data object	Mandatory
'B0'	Variable	Public Key Point Q (see note below)	Mandatory
'F0'	1 or 2	Key Parameter Reference (see note below)	Mandatory

NOTE: GlobalPlatform acknowledges as an error that values 'B0' and 'F0' are not valid ASN.1 tag values. This mistake was introduced in previous versions of this specification. Although this may be a problem for standard tools used to edit or display certificates and ASN.1 structures in general, it's been decided, for the moment at least, to keep such values to preserve backward compatibility with existing implementations already coping with such errors.

CASD Public and Private Keys associated with CASD certificates CERT.CASD.AUT and CERT.CASD.CT shall be stored in the CASD as needed (i.e. depending on the supported scenarios) using the STORE DATA command. The Data Group Identifiers to be used with the STORE DATA command are defined in [GPCS] section 11.11.4.2. The following KVN/KID identities are defined and shall be used when providing the keys to the CASD:

- CA Public Key PK.CA.AUT: KVN='74', KID='01', Key Usage='82'
- CA Public Key PK.CA.ECDSA: KVN='74', KID ='05', Key Usage ='82'
- CASD Private Key for Signature SK.CASD.AUT: KVN='74', KID='02', Key Usage='42'
- CASD Private Key for Decryption SK.CASD.CT: KVN='74', KID='03', Key Usage='48'
- CASD Private Key for Elliptic Curve SK.CASD.ECKA: KVN='74', KID='04', Key Usage='00 80'
- Within KVN='74', KID values ranging from '70' to '7F' are reserved for proprietary usage.

The Key Access Conditions, if provided, shall be set to '01' (SD only).

3.3.3 CASD Symmetric Keys

When the Pull Model (scenario #1) is used with symmetric keys, the CASD has to be provided with symmetric encryption and signature keys KS.CASD.CT and KS.CASD.AUT. These keys secure the transmission of the Application Provider encryption key (KS.AP.CT) to the APSD (used to encrypt the on-board generated key material).

The following key identities shall be used when provisioning the CASD symmetric keys with the PUT KEY or STORE DATA command:

- CASD Signature Key: KS.CASD.AUT (KVN='74', KID='02'), Key Usage='42'
- CASD Encryption Key: KS.CASD.CT (KVN='74', KID='03'), Key Usage='48'

The Key Access Conditions, if provided, shall be set to '01' (SD only).

3.3.4 Forwarded CASD Data

A mechanism is defined to enable retrieval of data forwarded from the CASD. This feature allows an Application Provider to select (or target through an OTA RAM script) its own Security Domain to retrieve data actually owned by the CASD.

The Application Provider shall send to its own Security Domain a GET DATA command complying with the following requirements:

- The P1-P2 parameters shall be set to 'BF30'
- The command shall have a command data field encoding a request for one (and only one) of the following data:
  - CA Security Domain Recognition Data: '5C 01 66'
  - CA Certificate Store (containing a sequence of one or more CASD Public Key Certificates): '5C 02 7F21'

If the command data field is coded differently, a status word of '6A80' shall be returned.

If the CASD is not present or not yet able to provide such data, a status word of '6A88' shall be returned. Otherwise, the response shall contain the requested data object retrieved from the CASD, encapsulated in a data object with tag 'BF30'.

For example,

- The following GlobalPlatform GET DATA command:

  '80 CA BF 30 04 5C 02 7F 21'

  would receive an answer of the following form:

  'BF 30' (length) '7F 21' (length) (value)

- The following ISO GET DATA command:

  '00 CA BF 30 04 5C 02 7F 21'

  would receive an answer of the following form:

  '7F 21' (length) (value)

3.4 Preparation of Application Provider Security Domain

As a prerequisite for establishing the Secure Channel Keys of an APSD, the APSD has to be provisioned with the following content:

- All Models except for Scenario #4:
  Recognition and configuration Data. The same rules as for the CASD apply (see section 3.3.1).
- Pull Model with asymmetric keys:
  Certificate CERT.AP.CT containing the AP public key PK.AP.CT
- Pull Model with symmetric keys:
  Application Provider Symmetric Encryption Key KS.AP.CT
- Push Model with Application Provider Certificate (scenario #2.A):
  Certificate CERT.AP.AUT containing the AP public key PK.AP.AUT
- Push Model without Application Provider Certificate (scenario #2.B):
  Temporary Secure Channel Keys (loaded as defined in [GPCS] and not described here)
- Key Agreement Model (scenario #3):
  (Optional) Temporary Secure Channel Keys (loaded as defined in [GPCS] and not described here)

3.4.1 Application Provider Certificate

The Controlling Authority generates the Application Provider Public Key Certificates. When the APSD receives the certificate, it requests the CASD to verify the Application Provider Public Key Certificate.

The Data Group Identifiers for Application Provider Certificate is defined in Table 3-9.

Table 3-9: Data Grouping Identifiers for Application Provider Certificate

DGI	DGI Length	Data Content	Function	Encrypt
'00DE'	Variable	Certificate with Message Recovery	Application Provider Public Key Certificate	No

NOTE: The following DGIs described in earlier versions of this specification are deprecated:

- DGI '00AE': Certificate without Message Recovery
- DGI '80AE': Encrypted Certificate with Message Recovery
- DGI '80DE': Encrypted Certificate without Message Recovery

The value of DGI '00DE' shall be formatted as described in Table 3-4 with the exception of the “CA Security Domain Image Number” TLV ('45'), which is not present. A CERT.AP.AUT is differentiated from a CERT.AP.CT by the content of the “Key Usage” field.

Table 3-10 describes the (TLV-encoded) data to be signed using the CA Private Key (SK.CA.AUT, private counterpart of PK.CA.AUT) to generate the signature of the AP certificate (i.e. content of tag '5F37'), as described in [ISO 9796-2].

Table 3-10: Data Signed to Generate the AP Public Key Certificate with Message Recovery

Tag	Length	Description	Presence
'7F49'	Variable	Public Key	Mandatory
'82'	1 or 3	Public Key Exponent	Mandatory
'81'	Variable	Public Key Modulus	Mandatory
'93'	1-16	Certificate Serial Number	Mandatory
'42'	Variable	CA Identifier	Mandatory
'5F20'	1-16	Subject Identifier	Mandatory
'95'	1	Key Usage ('82' digital signature verification, or '88' encipherment for confidentiality see [GPCS] Table 11-17)	Mandatory
'5F25'	4	Effective Date (YYYYMMDD, BCD format)	Conditional
'5F24'	4	Expiration Date (YYYYMMDD, BCD format)	Mandatory
'53'	1-127	Discretionary Data	Conditional

The CASD is responsible for verifying the certificate using PK.CA.AUT. Upon successful verification, the CASD extracts PK.AP.CT from the certificate and presents it to the APSD. The Public Key Modulus leftmost bytes are recovered from the verification of the signature ('5F37'). The Public Key Modulus rightmost bytes are recovered from the Public Key Modulus Remainder ('5F38').

Table 3-11: [table deleted in v1.1.1]

3.4.2 Loading Application Provider Symmetric Key

When the Pull Model with symmetric keys is used, the Application Provider symmetric encryption key KS.AP.CT is used to encrypt the on-card generated Security Domain key when retrieved by the Application Provider. The key is loaded to the APSD using the STORE DATA as a case 3 command with the following DGIs in the following sequence:

- DGI '00B8': Key KS.AP.CT meta-data
- DGI '80B8': Encrypted value of key KS.AP.CT
- DGI '00AE': Integrity signature over key KS.AP.CT

The Data Group Identifier of the Application Provider Key Control Reference Template (CRT) is defined in Table 3-12 – the STORE DATA command shall be coded as a case 3 command.

Table 3-12: Data Grouping Identifier for Application Provider Key CRT

DGI	DGI Length	Data Content	Function	Encrypt
'00B8'	Variable	Control Reference Template for confidentiality (CT)	Application Provider Key information data	No

The data content for the Application Provider Key CRT DGI '00B8' is defined in Table 3-13.

Table 3-13: Data Content for DGI '00B8'

Tag	Length	Data Element	Presence
'B8'	Variable	CRT tag (CT)	Mandatory
'95'	1	Key Usage Qualifier = '40' (encipherment of sensitive data in responses)	Mandatory
'80'	1	Key Type according to [GPCS] Table 11-16	Mandatory
'81'	1 or 2	Key Length	Mandatory
'82'	1	Key Identifier = '00' - '7F'	Optional
'83'	1	Key Version Number = '01' - '6F'	Optional
'B8'	Variable	CRT tag (CT)	Conditional
…	…	…	…

The Data Grouping Identifier '80B8' is used for transferring KS.AP.CT encrypted using KS.CASD.CT and the encryption method described in [GPCS] section B.1.1.1. If the plain text data is not a multiple of the encryption block length, then the last block shall be filled with arbitrary padding bytes.

Table 3-14: Data Grouping Identifier for Encrypted Application Provider Key

DGI	DGI Length	Data Content	Function	Encrypt
'80B8'	Variable	Encrypted key	Encrypted Application Provider Key	Yes

DGI '00AE' contains the signature of the CRT and KS.AP.CT computed using KS.CASD.AUT and the signature method described in [GPCS] section B.1.2.1

Table 3-15: Content of DGI '00AE' for AP Symmetric Key Integrity Signature

Tag	Length	Description
'8E'	8	Signature of (DGI '00B8' || DGI '80B8')

To recover KS.AP.CT, the APSD provides the CASD with the concatenation of DGIs '00B8', '80B8', and '00AE'. After a successful verification of the signature provided within tag '8E' (using KS.CASD.AUT) the CASD decrypts DGI '80B8' (using KS.CASD.CT).

3.5 Establishment of Application Provider Secure Channel Keys

3.5.1 Common Data and Data Structures

In all personalization scenarios except Scenario #4, the DGI '00A6' defined in Table 3-16 shall be sent within a STORE DATA command in all personalization models:

- to trigger the on-card Security Domain key generation in the Pull Model
- to initiate the transmission of the new Secure Channel Keys to the APSD in the Push Model
- to trigger the key agreement in the Key Agreement Model

Table 3-16: Data Grouping Identifier for Key Generation

DGI	DGI Length	Data Content	Function	Encrypt
'00A6'	Variable	Control Reference Template (KAT)	Key agreement/generation CRT	No

The value of DGI '00A6' shall be encoded as described in Table 3-17 if all generated keys shall have the same type and length, or Table 3-18 if generated keys shall have different types and/or different lengths. The use of Table 3-18 is limited to Scenario #3.

In scenario #4, tag 'A6' described in Table 3-17 shall be provided directly in the discretionary data (tag '73') of the AP's certificate CERT.AP.ECKA.

Table 3-17: Content of DGI '00A6' to Generate Secure Channel Keys of Same Types and Lengths

Tag	Length	Data Element	Presence
'A6'	Variable	CRT tag (KAT)	Mandatory
'90'	1 or 2	Byte 1: Scenario identifier: '01': Pull Model '02': Push Model with AP Certificate '03': Key Agreement Model (ECKA-EG) '04': Push Model without AP Certificate '05': Key Agreement with no Secure Channel '06' – '2F': Reserved for GlobalPlatform use '30' – '9F': RFU 'A0' – 'BF': Reserved for Proprietary Use 'C0' – 'FF': RFU Byte 2: Scenario parameters (see Table 3-19), only present if byte 1 = '03' or '05'	Mandatory
'95'	1	Key Usage Qualifier '5C' (1 secure channel base key) or '10' (3 secure channel keys) (see [GPCS] Table 11-17)	Mandatory
'96'	1	Key Access according to [GPCS] Table 11-18	Optional
'80'	1	Key Type according to [GPCS] Table 11-16	Mandatory
'81'	1	Key Length (in bytes)	Mandatory
'82'	1	Key Identifier = '00' - '7F'	Optional
'83'	1	Key Version Number = '01' - '6F'	Optional
'91'	Variable	Initial value of sequence counter	Optional
'45'	1-n	Security Domain Image Number (SDIN)	Optional
'84'	1-n	HostID (shall only be present if scenario '03' parameter b3 is set)	Conditional

Table 3-18: Content of DGI '00A6' to Generate Secure Channel Keys of Different Types or Lengths

Tag	Length	Data Element	Presence
'A6'	Variable	CRT tag (KAT)	Mandatory
'90'	2	Byte 1: Scenario identifier – value '03' only Byte 2: Scenario parameters (see Table 3-19)	Mandatory
'82'	1	Key Identifier = '00' - '7F'	Optional
'83'	1	Key Version Number = '01' - '7F'	Mandatory
'B9'	Variable	Description of 1st key	Conditional
'95'	1	Key Usage Qualifier '3C' (Pre Shared Key for Secure Messaging) '34' (C-MAC + R-MAC) '38' (C-ENC + R-ENC) 'C8' (C-DEK + R-DEK) (see [GPCS] Table 11-17)	Mandatory
'96'	1	Key Access according to [GPCS] Table 11-18	Optional
'80'	1	Key Type according to [GPCS] Table 11-16	Mandatory
'81'	1	Key Length (in bytes)	Mandatory
'B9'	Variable	Description of 2nd key	Conditional
…	…	…	…
'91'	Variable	Initial value of sequence counter	Optional
'45'	1-n	Security Domain Image Number (SDIN)	Optional
'84'	1-n	HostID (shall only be present if scenario parameter b3 is set)	Conditional

The value of tag '90' (Scenario Identifier) allows the Security Domain to decide a course of action, when receiving tag 'A6' (e.g. triggering on-card key generation or waiting for other DGI containing keys). The specific actions to be performed by the different personalization models are described in dedicated subsections below.

The card shall verify the values provided for scheme identifier, scheme parameters, key identifier, and key type and apply the following rules:

- Tag '82' indicates the Key Identifier assigned to the first key. The assigned Key Identifier shall be incremented by 1 for each subsequent key. If tag '82' (Key Identifier) is not present, a Key Identifier of '01' shall be assigned to the first key.
- Implementations may use proprietary rules to handle the case where tag '83' is not present; however such rules remain out of scope of this document.
- The Secure Channel Protocol for which the Secure Channel keys are generated is implicitly known from the specified Key Version Number (tag '83') and the list of Secure Channel Protocols supported by the Security Domain processing the STORE DATA command. Such rules remain out of scope of this document and shall be defined in configuration documents.

- The length of tag '91' depends on the Secure Channel Protocol that is expected to be used with the newly generated key set. If not present, the sequence counter (if any) shall be initialized to a value of 0.
- When the Key Access field (tag '96') is not present, the default key access value is '00'.
- If the specified key information (e.g. key type, key length, key identifier, etc.) or the number of keys is not consistent with the intended Secure Channel Protocol, an error should be returned.

The Key Agreement Model scenario parameters (byte 2 of tag '90') are defined as follows:

Table 3-19: Parameters for Scenario #3 and Scenario #4

b8	b7	b6	b5	b4	b3	b2	b1	Description
-	-	-	-	-	-	-	1	Do not delete existing keys
-	-	-	-	-	-	1	-	Include DR in key derivation process
-	-	-	-	-	1	-	-	Include Host and Card ID in key derivation process
-	-	-	-	1	-	-	-	Reserved for use by GSMA
x	x	x	x	-	-	-	-	RFU (0)

If the indicated key set already exists, it shall be replaced; else a new key set shall be created.

If the bit for “Do not delete existing keys” is set, all other keys in the SD shall remain unchanged; else all keys in the SD except the newly generated key set shall be deleted. Using bit 1 set to 1, it is for example possible to run scenario #3 more than once to generate additional key sets for the same Security Domain. Note that this bit does not apply to Scenario #4 and shall be ignored. If that scenario is replayed before a Secure Channel session has been successfully established, then the previous key set shall be discarded.

If bit 3 is set (“Include Host and Card ID in key derivation process”) and tag '84' (Host ID) is not present within tag 'A6' (see Table 3-17 and Table 3-18), then an error shall be returned. Similarly, if bit 3 is not set and tag '84' (Host ID) is present within tag 'A6', then an error shall be returned.

3.5.2 Scenario #1: Pull Model

The STORE DATA command is used as a case 4 command with DGI '00A6' to trigger the on-card generation as described in section 3.5.1. When the STORE DATA command is received the following steps shall occur:

- A 16-byte key shall be randomly generated (RGK).
- If the Key Usage Qualifier provided by the CRT is '10' (see Table 3-17), then a secure channel key set with three keys known as K_{ENC}, K_{MAC}, and K_{DEK} of the same type and length and other key attributes as provided in the CRT shall be derived from the RGK.

  If three static DES keys need to be derived from the RGK, the following algorithm shall be used:

  The K_{ENC} is a 16-byte (112 bits plus parity) DES key derived in the following way:

  DES3(RGK)['FF FF FF FF FF FF' || 'F0' || '01'] || DES3(RGK)['FF FF FF FF FF FF' || '0F' || '01']

  The K_{MAC} is a 16-byte (112 bits plus parity) DES key derived in the following way:

  DES3(RGK)['FF FF FF FF FF FF' || 'F0' || '02'] || DES3(RGK)['FF FF FF FF FF FF' || '0F' || '02']

  The K_{DEK} is a 16-byte (112 bits plus parity) DES key derived in the following way:

  DES3(RGK)['FF FF FF FF FF FF' || 'F0' || '03'] || DES3(RGK)['FF FF FF FF FF FF' || '0F' || '03']

  If three static 128-bit AES keys need to be derived from the RGK, the following algorithm shall be used:

  The K_{ENC} is a 128-bit AES key derived in the following way:

  AES(RGK)['FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF' || '0F' || '01']

  The K_{MAC} is a 128-bit AES key derived in the following way:

  AES(RGK)['FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF' || '0F' || '02']

  The K_{DEK} is a 128-bit AES key derived in the following way:

  AES(RGK)['FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF' || '0F' || '03']

- The RGK shall be encrypted.
  - If the asymmetric scheme is used, the RGK is ciphered using the PK.AP.CT public key. If PK.AP.CT is a 1024-bit RSA public key, then the RSAES-PKCS-V1.5 encryption algorithm as specified in [GPCS] shall be used. If PK.AP.CT is an RSA public key longer than 1024 bits, then the RSAES-OAEP encryption algorithm as defined in [GPCS] section B.3.2.2 shall be used. The length of the RGK is provided by the value of the CRT tag '81' within the DGI '00A6'.
  - If the symmetric scheme is used, the RGK is ciphered using KS.AP.CT and the encryption method described in [GPCS] section B.1.1.1. If the plain text data is not a multiple of the encryption block length, then the last block shall be filled with arbitrary padding bytes.
- The APSD shall then provide the CASD with the following data:

Table 3-20: Pull Model – Data to be Signed On-Card by the CASD

Length	Description
1	Length of AP Security Domain AID
5-16	AP Security Domain AID
1	Length of Control Reference Information
Variable	Control Reference Information (Content of 'A6')
Variable	Length of Encrypted RGK. The length shall be BER-TLV encoded. For backward compatibility, a length of 128 may be encoded as '80' or '81 80'.
Variable	Encrypted RGK

- Depending on the Pull Mode the CASD signs the data with its asymmetric or symmetric key (SK.CASD.AUT / KS.CASD.AUT).
  - If the asymmetric scheme is used, the above data shall be signed by the CASD using the signature algorithm described in [ISO 9796-2]. Scheme 1[2] (with trailer option 1 and SHA-1 digest) shall be used if SK.CASD.AUT is a 1024-bit RSA key. Scheme 2 (with trailer option 1 and SHA-256 digest) shall be used if SK.CASD.AUT is an RSA key longer than 1024 bits, using the alternative signature production function defined in [ISO 9796-2] Annex B.6.
  - If the symmetric scheme is used, the above data shall be signed by the CASD using KS.CASD.AUT and the signature method described in [GPCS] section B.1.2.1.

A response is returned to the Application Provider depending on the cryptographic scheme used:

- If the symmetric scheme was used, then the signed data (as described in Table 3-20) shall be returned, followed by the generated signature wrapped in LV format.
- If the asymmetric scheme was used, then the following data shall be returned:

Table 3-21: Pull Model – Response Data in PK Mode

Length	Description
Variable	Length of CASD Signature. The length shall be BER-TLV encoded. For backward compatibility, a length of 128 may be encoded as '80' or '81 80'.
Variable	CASD Signature
Variable	Length of Remainder Data. The length shall be BER-TLV encoded. For backward compatibility, a length of 128 may be encoded as '80' or '81 80'.
Variables	Remainder Data

[2] A simplified description of this scheme can be found in [EMV Book 2].

NOTE: In previous versions of this specification, the Length of Encrypted RGK in Table 3-20, as well as Length of CASD Signature and Length of Remainder Data in Table 3-21, were described as being encoded on 1 byte only. Therefore, some implementations may have assumed that a length of 128 could be encoded as '80' and more generally, that this 1-byte length field could encode lengths up to 255 ('FF'). This version of the specification clarifies that these length fields, like all length fields in GlobalPlatform specifications (see [GPCS] section 11.1.5), shall be coded according to ASN.1 BER-TLV rules. Nevertheless, for compatibility with such legacy implementations, off-card entities should tolerate and be able to interpret 1-byte length fields encoding values up to 255 ('FF'). E.g. a length field starting with 'C0' should be interpreted as a 1-byte length field encoding a length of 192. Of course, a BER-TLV length field (e.g. starting with '81' or '82') shall always be correctly interpreted.

3.5.3 Scenario #2: Push Model

The STORE DATA command is used with DGI '00A6' to initiate the personalization of the Secure Channel Keys as described in section 3.5. When the STORE DATA command is received, the following steps shall occur:

- The key attributes provided with CRT 'A6' are kept and assigned to the new Secure Channel Keys if the procedure succeeds.
- The APSD requests the CASD to extract and decrypt the encrypted Secure Channel Keys provided with DGI '8010' (see Table 3-22) in the same or the next STORE DATA command.
- If the Push Model with Application Provider certificate was indicated (scenario #2.A), the APSD extracts and verifies the signature over the DGIs '00A6' and '8010' provided with DGI '00AE'.
- If the Push Model without Application Provider certificate was indicated (scenario #2.B) there is no DGI '00AE' and the command origin is verified by ensuring the required Secure Channel session.
- If the command authentication succeeds, the new APSD Secure Channel Keys are established with the received attributes.

DGI '8010' contains the encrypted symmetric keys:

Table 3-22: Data Grouping Identifier for Encrypted Secure Channel Keys

DGI	DGI Length	Data Content	Function	Encrypt
'8010'	Variable	Encrypted Secure channel symmetric keys	Application provider symmetric secure channel keys	Yes

The underlying plain text structure is provided in Table 3-23 and Table 3-24.

If the asymmetric scheme is used, the data content is ciphered using the PK.CASD.CT public key. If PK.CASD.CT is a 1024-bit RSA public key, then the RSAES-PKCS-V1.5 encryption algorithm as specified in [GPCS] shall be used. If PK.CASD.CT is an RSA public key longer than 1024 bits, then the RSAES-OAEP encryption algorithm as defined in [GPCS] section B.3.2.2 shall be used. The length of each key is provided by the value of the CRT tag '81' within the DGI '00A6'.

If the symmetric scheme is used, the data content is ciphered with KS.CASD.CT using the encryption method described in [GPCS] section B.1.1.1. If the plain text data is not a multiple of the encryption block length, then the last block shall be filled with arbitrary padding bytes.

If the DGI '00A6' indicates three Keys, the Data content is as defined in Table 3-23.

Table 3-23: Data Content for DGI '8010' – Encrypted Secure Channel Keys Values

Data Element	Presence
ENC key	Mandatory
MAC key	Mandatory
DEK key	Mandatory

If the DGI '00A6' indicates one Master Key, then the data content is as defined in Table 3-24.

Table 3-24: Data Content for DGI '8010' – Secure Channel Base Key Value

Data Element	Presence
Secure Channel Base Key	Mandatory

DGI '00AE' provides integrity data for the Secure Channel Keys. The content of DGI '00AE' is as follows:

Table 3-25: Push Model – Verification Template Content

Tag	Length	Description
'9E'	Variable	Signature of (DGI '00A6' || DGI '8010')

The content of tag '9E' is a signature generated by the Application Provider over the concatenation of DGI '00A6' and DGI '8010', using the SK.AP.AUT private key. If SK.AP.AUT is a 1024-bit RSA private key, then the RSASSA-PKCS-v1_5 signature algorithm as defined in [GPCS] section B.3.1.1 shall be used. If SK.AP.AUT is an RSA private key longer than 1024 bits, then the RSASSA-PSS signature algorithm as defined in [GPCS] section B.3.2.1 shall be used. This signature shall be verified by the APSD, using the Application Provider Public Key (PK.AP.AUT).

3.5.4 Scenario #3: Key Agreement Model

The STORE DATA command is used with DGI '00A6' to initiate the personalization of the Secure Channel Keys as described in section 3.5.1. When the STORE DATA command is received, the following steps shall occur:

- The APSD ensures that the STORE DATA command(s) involved in the procedure are processed within a Secure Channel with at least MAC protection.
- The key attributes and the Scenario Parameters provided with CRT 'A6' are kept and applied to the new Secure Channel Keys if the procedure succeeds.
- The APSD extracts the Application Provider ephemeral Public Key ePK.AP.ECKA provided with DGI '7F49' (see Table 3-26) following DGI '00A6' and requests the CASD to generate the shared secret ShS using this key and the static CASD Private Key SK.CASD.ECKA according to BSI TR-03111 ([TR-03111]).
- If the parameter “Include DR in key derivation process” in the Parameters for Scenario #3 is set (see Table 3-19), the APSD generates a derivation random (DR) of a length depending on the ECC key length as defined in Table 3-27.
- The APSD derives key material KeyData as defined in [TR-03111] for the “X9.63 Key Derivation Function” from the SharedInfo parameters defined in Table 3-28 using SHA-256 for one or more keys as described in Table 3-29 and Table 3-30.
- Depending on the Parameters for Scenario #3 (see Table 3-19), the APSD creates a new key set or updates an existing one and deletes all other existing keys.
- A response is returned by the APSD including a receipt and, optionally, the requested DR as defined in Table 3-27.

DGI '7F49' is used to send the ephemeral public key of the AP to the card. It shall follow DGI '00A6'.

Table 3-26: Data Grouping Identifier for ECC AP Ephemeral Public Key

DGI	Length	Description
'7F49'	Variable	ePK.AP.ECKA

The ephemeral public key shall be coded using uncompressed encoding as specified in [TR-03111] section 3.2.1.

If a DR is requested, its length shall relate to length the ECC key as follows:

Table 3-27: Length of the Derivation Random

ECC key length in bits	Length of DR in bytes
256-383	16
384-511	24
512+	32

The concatenation of the values in Table 3-28 shall be used as SharedInfo in the key derivation process to generate the required KeyData.

Table 3-28: SharedInfo

Parameter	Length	Presence
Key usage qualifier	1	Mandatory
Key type	1	Mandatory
Key length	1	Mandatory
Derivation random (DR)	16, 24, or 32	Conditional (see Note 1)
HostID-LV	Variable	Conditional (see Note 2)
IIN-LV	Variable	Conditional (see Note 2)
CIN-LV	Variable	Conditional (see Note 2)

Note 1: The DR shall be present when indicated in the Parameters for Scenario #3 (see Table 3-19).

Note 2: This parameter shall be included if the parameter "Include Host and Card ID in key derivation process" in the Parameters for Scenario #3 is set (see Table 3-19). (The presence of unique host, i.e. off card entity, and card identifiers is required in NIST SP 800-56A [NIST 800-56A].)

HostID-LV is the length and the value field of the HostID given in the command data.

IIN-LV is the length and the value field of the Issuer Identification Number (see [GPCS]).

CIN-LV is the length and the value field of the Card Image Number (see [GPCS]).

In addition to the key(s) required for secure channel usage, a receipt key is used to calculate the receipt to be included in the response to the STORE DATA command. The type and length of the receipt key, as well as the signature algorithm used to generate the receipt, depend on the type and length of the generated secure channel key(s).

If the generated Secure Channel keys all share the same type and length (see Table 3-17), then:

- The receipt key shall be of the same type and length (L) as the generated Secure Channel keys.
- The receipt generation algorithm shall be the signature algorithm described in:
  - ISO/IEC 9797-1 [ISO 9797-1] with details given in [GPCS] section B.1.2.2, if the receipt key is a DES key.
  - NIST SP 800-38B [NIST 800-38B] if the receipt key is an AES key.

KeyData shall be assigned to the keys as follows (with L being the key length).

Table 3-29: KeyData Assignment in Case of One Base Key

KeyData	Key
1 to L	receipt key
L+1 to 2L	secure channel base key

Table 3-30: KeyData Assignment in Case of a Full Key Set

KeyData	Key
1 to L	receipt key
L+1 to 2L	S-ENC / KIc
2L+1 to 3L	S-MAC / KID
3L+1 to 4L	DEK

If the generated Secure Channel keys are of different types and/or lengths (see Table 3-18), then:

- The receipt key shall be a 16-byte AES key
- The receipt generation algorithm shall be the signature algorithm described in [NIST 800-38B].
- *KeyData* shall be assigned sequentially to the receipt key and to each of the secure channel key(s) as described in Table 3-31.

Table 3-31: Assignment of KeyData for Keys of Different Types or Lengths

KeyData	Key
1 to 16	receipt key
17 to 17+L1	1st secure channel key (length L1)
18+L1 to 18+L1+L2	2nd secure channel key (length L2)
…	…

The APSD shall generate a receipt by calculating a signature across the data described in Table 3-32, using the receipt key and the algorithm defined above. The receipt key shall be deleted after calculating the receipt.

Table 3-32: Input Data for Receipt Calculation

Tag	Length	Data Element	Presence
'A6'	Variable	CRT TLV with all sub TLVs as provided in the STORE DATA command	Mandatory
'85'	Variable	DR	Conditional

The STORE DATA response shall contain the following data:

Table 3-33: Response Data for Scenario #3

Tag	Length	Data Element	Presence
'85'	Variable	DR	Conditional
'86'	Variable	Receipt	Mandatory

3.5.5 Scenario #4: Key Agreement Model with no Secure Channel

Scenario #4 is used to set up secure channel keys in an APSD that has no access to a secure channel (for example a self-extradited SD that hasn't been personalized with a secure channel key set). If this scenario is supported then the mechanism is enabled when the APSD is installed and remains enabled until a secure channel is successfully opened between the Application Provider and the APSD using the key set generated by this process. Scenario #4 uses the (Cofactor) Full Unified Model C(2e, 2s, ECC CDH) Scheme as specified in [NIST 800-56A].

The Application Provider uses the GET DATA command to extract the CASD's certificate CERT.CASD.ECKA. This is verified by the Application Provider and the static public key PK.CASD.ECKA is retrieved.

The INITIALIZE SECURITY command is used to initiate the personalization of the Secure Channel Keys. This command is used to provide the Application Provider certificate CERT.AP.ECKA and the Application Provider ephemeral public key ePK.AP.ECKA; see Table 3-34.

Table 3-34: Contents of the INITIALIZE SECURITY Command for Scenario #4

Tag	Length	Description
'7F21'	Variable	The certificate CERT.AP.ECKA
'7F49'	Variable	ePK.AP.ECKA

The format of the certificate is as defined in Table 3-6; the data to be signed to generate the signature is as defined in Table 3-7.

The following Discretionary Data (tag '73') shall be present:

Table 3-35: Discretionary Data Mandatory for Scenario #4

Tag	Length	Description
'4F'	'05' to '10'	Target APSD AID
'A6'	Variable	KAT TLV (see Table 3-17)

When the INITIALIZE SECURITY command is received with no secure session in place, the following steps shall occur:

- The APSD checks that the Scenario #4 mechanism is enabled and that the INITIALIZE SECURITY command provides the certificate and ephemeral key necessary to trigger the scenario.
- The APSD checks the discretionary data in the certificate:
  - That the AID provided in the discretionary data matches its own AID;
  - That the 'A6' data are correctly formed and correspond to the Scenario #4 mechanism.
- The key attributes and the Scenario Parameters provided with the CRT 'A6' data extracted from the discretionary data are kept and applied to the new Secure Channel Keys if the procedure succeeds.
- The recipient APSD forwards the certificate to the CASD, which verifies the certificate and checks that the AID in the discretionary data matches the AID of the calling SD.
- The CASD extracts the Application Provider static public key PK.AP.ECKA from the certificate and generates the shared secret ShSss using this key and its static private key SK.CASD.ECKA, according to [NIST 800-56A]. The CASD returns the static shared secret ShSss to the APSD.

- The APSD generates its own ephemeral key pair eSK/ePK.SD.ECKA and uses its Private ephemeral key together with the Application Provider's Public ephemeral key to generate the shared secret ShSee, according to [NIST 800-56A].
- The APSD derives key material KeyData as defined in [TR-03111] for the "X9.63 Key Derivation Function" from the concatenation of the shared secrets ShSss and ShSee, and the SharedInfo parameters defined in Table 3-28 using SHA-256 for one or more keys as described in Table 3-29 and Table 3-30.
- The APSD creates a new key set and populates it with the derived secure channel keys.
- A response is returned by the APSD including the APSD's ephemeral public key ePK.SD.ECKA, and a receipt as defined in Table 3-27.

In addition to the key(s) required for secure channel usage, a receipt key is used to calculate the receipt to be included in the response to the INITIALIZE SECURITY command. The receipt key is generated and used according to the rules set out for Scenario #3 in section 3.5.4. The APSD shall generate a receipt across the data described in Table 3-36.

The receipt key shall be deleted after calculating the receipt and cryptogram.

Table 3-36: Input Data for Receipt Calculation

Tag	Length	Data Element	Presence
'4F'	'05'-'10'	AID of the APSD	Mandatory
'A6'	Variable	CRT TLV with all sub TLVs as provided in CERT.AP.ECKA	Mandatory
'7F49'	Variable	The APSD's ephemeral public key ePK.SD.ECKA	Mandatory

The INITIALIZE SECURITY response shall contain the following data:

Table 3-37: Response Data for Scenario #4

Tag	Length	Data Element	Presence
'7F49'	Variable	The APSD's ephemeral public key ePK.SD.ECKA	Mandatory
'86'	Variable	Receipt	Mandatory

On receipt of the response data, the Application Provider can reproduce the generation of the secure channel keys:

- The static Public key extracted from the CASD's certificate, PK.CASD.ECKA, is used in conjunction with the Application Provider's Private static key to generate the static shared secret ShSss;
- The ephemeral Public key provided in the response data, ePK.SD.ECKA, is used in conjunction with the Application Provider's Private ephemeral key to generate the ephemeral shared secret ShSee;
- The Application Provider uses the shared secrets, the SharedInfo, and the X9.63 Key Derivation Function to generate the same keys as the APSD generated.

The Application Provider then uses the derived receipt key to verify the receipt attached to the APSD's response.

Successful verification of the receipt allows the Application Provider to be assured of the veracity of the secure channel keys. However, the APSD has had no confirmation that the Application Provider generated the keys correctly, or even received the APSD's response. Confirmation is only guaranteed when a Secure Channel session has been established successfully using the key set. At this point the APSD disables the Scenario #4 mechanism: key management can now be performed using Scenarios #1, #2, or #3 defined in this document or the PUT KEY/STORE DATA commands described in [GPCS].

3.6 Ciphered Load File Data Block Privilege

The description of this feature has been moved to [GPCS].

See [GPCS] Table 6-1, section 9.1.3.7, section 9.3.5, Table 11-9, and section C.6.

3.7 DGI for Personalizing Security Domain Keys

The description of this feature has been moved to [GPCS]. See [GPCS] section 11.11.4.

3.8 Introduction of a Token Using Symmetric Keys

The description of this feature has been moved to [GPCS]. See [GPCS] section C.4.

4 API for Confidential Personalization

Two interfaces are defined to support the confidential application personalization. These interfaces are the `Personalization` and `Authority` interfaces and are added into the package `org.globalplatform`. See GlobalPlatform Java Card API v1.7 ([GPC API]).

4.1 `Personalization` Interface

Application personalization is defined in [GPCS] section 7.3.2, Security Domain Support for Application Personalization. In certain situations the Applet needs to send response data during personalization. For this purpose, the new interface `org.globalplatform.Personalization` is introduced. The Associated Security Domain shall call the method `processData()` of the `Personalization` interface if it is implemented by the Applet which is being personalized, otherwise it shall call the method `processData()` of the `Application` interface. The Associated Security Domain is responsible for sending out this response data.

The APDU command forwarded by the Security Domain shall be unwrapped according to the Security Level of the current Secure Channel Session. The buffer parameter is the APDU buffer, the offset parameter is set to zero, and the length parameter is set to the length of the unwrapped APDU command message.

The STORE DATA command forwarded to the Applet may be a case 3 or 4 (ISO/IEC 7816-4 [ISO 7816-4]) command, as described in [GPCS] section 11.11. The bit 1 (rightmost bit) of the reference control parameter P1 is used to indicate to the card that it is an [ISO 7816-4], case 4 command, and therefore, response data is expected. For a case 4 command, the Application may or may not return data. For a case 3 command, response data are not expected and the error code '6A86' is returned if the Applet indicates that response data are available. The Applet may check the case of the command by looking at the P1 parameter in the APDU buffer.

4.2 `Authority` Interface

This interface provides services to verify a key and to sign data. The CASD may register an instance of this interface to the OPEN as a Global Service with the service family identifier = '83' (see [GPCS] section 8.1.3), to make this service available to other Security Domains. However, the usage of this interface is optional (i.e. internal interactions between Security Domains and the CASD may be based on proprietary internal interfaces). The registration shall fail if the card has already reached the SECURED life cycle state. Only Security Domains shall be able to use this Global Service.

The use of the Authority Interface is not defined for the Key Agreement Model.

5 Controlling Authority Services to Applications

5.1 Overview

The Controlling Authority Security Domain (CASD) as defined in section 3.1 may additionally provide an on-demand signature services to applications, the CASD Signature Service.

This service is used through the `AuthoritySignature` interface defined in section 5.3. To make it available, the CASD shall uniquely register an instance of the `AuthoritySignature` interface as a Global Service under service identifier '8700'.

This section defines which additional data shall be personalized in the CASD to enable such services.

5.2 Preparation of Controlling Authority Security Domain

5.2.1 CASD Data

Support of the CASD Signature Service shall be indicated in the CASD Capability Information as defined in Table 3-4.

5.2.2 CASD Certificates and Asymmetric Keys

The following certificate shall be stored in the CASD certificate store as defined in section 3.3.2:

- Certificate CERT.CASD-SIGN.AUT containing the public key PK.CASD-SIGN.AUT used by the Application Provider to verify signatures generated for on-card Applications by the CASD Signature Service.

The certificate format shall be the following:

Table 5-1: Certificate Structure for CERT.CASD-SIGN.AUT (Certificate without Message Recovery)

Tag	Length	Description	Presence
'7F21'	Variable	Certificate	Mandatory
'93'	1-16	Certificate Serial Number	Mandatory
'42'	Variable	CA Identifier	Mandatory
'5F20'	1-16	Subject Identifier	Mandatory
'95'	1	Key Usage ('82' digital signature verification) see [GPCS] section 11.1.9	Mandatory
'5F25'	4	Effective Date (YYYYMMDD, BCD format)	Optional
'5F24'	4	Expiration Date (YYYYMMDD, BCD format)	Mandatory
'45'	Variable	CASD Image Number	Mandatory
'73'	1-127	Discretionary Data (see Table 5-3)	Mandatory
'7F49'	Variable	Public Key data object (see Table 5-4)	Mandatory
'5F37'	Variable	Signature	Mandatory

Table 5-2 describes the data to be signed using the CA Private Key (SK.CA.AUT, private counterpart of PK.CA.AUT) and the ECDSA algorithm (see [GPCS] section B.4.3) to generate the signature of the CASD certificate (tag '5F37'):

Table 5-2: Data Signed to Generate the Signature of CERT.CASD-SIGN.AUT

Tag	Length	Description	Presence
'93'	1-16	Certificate Serial Number	Mandatory
'42'	Variable	CA Identifier	Mandatory
'5F20'	1-16	Subject Identifier	Mandatory
'95'	1	Key Usage	Mandatory
'5F25'	4	Effective Date (YYYYMMDD, BCD format)	Conditional
'5F24'	4	Expiration Date (YYYYMMDD, BCD format)	Mandatory
'45'	Variable	CA Security Domain Image Number	Mandatory
'73'	1-127	Discretionary Data (see Table 5-3)	Conditional
'7F49'	Variable	Public Key data object (see Table 5-4)	Mandatory

The format of the Discretionary Data (tag '73') is described in the following table:

Table 5-3: Discretionary Data field

Tag	Length	Description	Presence
'73'	Variable	Discretionary Data	Mandatory
'C9'	20	Value part of the `KeyIdentifier` of the `AuthorityKeyIdentifier` as defined in sections 4.2.1.1 and 4.2.1.2 of [X.509]. It corresponds to the SHA-1 hash of the value of the `subjectPublicKey` (excluding the tag, length, and number of unused bits) of the CA Public Key PK.CA.AUT that shall be used by the off-card entity to verify the CASD certificate CERT.CASD-SIGN.AUT.	Mandatory
'E1'	Variable	DER encoded value part of the `SignatureIdentifier` as defined in section 4.1.1.2 of [X.509]	Mandatory

The format of the ECC Public Key data object (tag '7F49') is described in the following table:

Table 5-4: Public Key Data Object Data Field

Tag	Length	Description	Presence
'7F49'	Variable	Public Key data object	Mandatory
'80'	Variable	Public Key Point Q	Mandatory
'81'	1 or 2	Key Parameter Reference	Mandatory

The following KVN/KID identity is defined and shall be used when providing the keys to the CASD:

- CASD Private Key for ECDSA Signature SK.CASD-SIGN.AUT: KVN='74', KID='10', Key Usage='42'

The Key Access Conditions, if provided, shall be set to '01' (CASD only).

5.3 `AuthoritySignature` Interface

This interface is defined in package `org.globalplatform` and allows requesting the CASD to generate a signature over the input data provided by a client Application.

To make this service available, the CASD shall uniquely register an instance of the `AuthoritySignature` interface as a Global Service under service identifier '8700'.

The client Application shall initialize the service and select a signature mode with the `init()` method. The to-be-signed data and output data described hereafter apply when the selected signature mode is MODE_INTEGRITY (='01').

The CASD just signs the data provided by the application; the CASD doesn't perform any checks on the payload provided by the application. However, before generating the signature, the CASD appends some metadata to the payload in order to provide additional information to the off-card entity: information on the application requesting the signature, identification of the algorithm, and public key to be used to verify the signature.

Notice that this signature generated by the CASD does not provide any protection with respect to replay scenarios. If any such protection is required, the application using this service should implement suitable countermeasures.

In ASN.1 notation [ASN.1], the format of `SignOutputData`, the output data generated by the `sign()` method, is the following:

```
SignOutputData ::= [APPLICATION 26] SEQUENCE{
  securityEnvironmentTemplate  SecurityEnvironmentTemplate,
  signature [APPLICATION 55] OCTET STRING
}
```

With:

- `securityEnvironmentTemplate` – A Security Environment Template as defined in section 5.3.1.
- `signature` – The signature – using ECDSA algorithm (see [GPCS] section B.4.3) and SK.CASD-SIGN.AUT – of the data defined in Table 5-6.

The DER TLV encoding ([ASN.1 Encoding]) of `SignOutputData` is described in the following table:

Table 5-5: DER-TLV Encoding of `SignOutputData`

Tag	Length	Description	Presence
'7A'	Variable	`SignOutputData`	Mandatory
'7B'	Variable	`securityEnvironmentTemplate`	Mandatory
'5F37'	Variable	`signature`	Mandatory

The data signed to generate `signature` are the following:

Table 5-6: Data Signed by the CASD

Tag	Length	Description	Presence
-	-	Application Payload	Mandatory
'7B'	Variable	`securityEnvironmentTemplate`	Mandatory

5.3.1 Security Environment Template

In ASN.1 notation ([ASN.1]), the format of `SecurityEnvironmentTemplate` is the following:

```
SecurityEnvironmentTemplate ::= [APPLICATION 27] SEQUENCE {
  clientApplicationInformation [0]SEQUENCE {
    requestedSignatureMode [0]OCTET STRING (SIZE(1)),
    instanceAID [1]OCTET STRING (SIZE(5..16)),
    executableLoadFileAID [3]OCTET STRING (SIZE(5..16)),
    executableLoadFileVersionNumber [4]OCTET STRING
    },
  algorithmIdentifier [1]AlgorithmIdentifier,
  keyIdentifier [2]KeyIdentifier
 }

-- The Algorithm Information as defined in section 4.1.1.2 of RFC5280
AlgorithmIdentifier ::= SEQUENCE {
  algorithm OBJECT IDENTIFIER,
  parameters ANY DEFINED BY algorithm OPTIONAL }

-- The Key Information as defined in section 4.2.1.1 and 4.2.1.2 of
RFC5280
KeyIdentifier ::= OCTET STRING
```

With:

- `clientApplicationInformation` – Information on the application that requested the CASD signature, the client Application. It includes:
  - `requestedSignatureMode` – The signature mode selected by the client Application with the `init()` method.
  - `instanceAID` – The Instance AID of the client Application.
  - `executableLoadFileAID` – The Executable Load File AID of the client Application.
  - `executableLoadFileVersionNumber` – The version of the Executable Load File of the client Application.

    NOTE: The format and the content of the Executable Load File Version Number depend on the format of the Load File. On a Java Card based card, this is a 2-byte version number reflecting the major and minor version attributes (in this order) of the Java Card CAP file.

- `algorithmIdentifier` – The algorithm identifier (as defined in section 4.1.1.2 of [X.509]) of the algorithm used by the CASD to sign the client Application payload. It includes:
  - `algorithm` – The OID for ECDSA is "iso(1) member-body(2) us(840) ansi-X9-62(10045) keyType(2) ecPublicKey(1)" as defined in [RFC 5480]. The corresponding DER encoding is '06072A8648CE3D0201'.
  - `parameters` – The value depends on the algorithm used. For the ECDSA algorithm, Table 5-7 describes the `parameters` value in OID format for the different ECC curves as well as the corresponding DER encoding.
- `KeyIdentifier` – The `keyIdentifier` of the `AuthorityKeyIdentifier` (as defined in sections 4.2.1.1 and 4.2.1.2 of [X.509]); i.e. the SHA-1 hash of the value of the `subjectPublicKey` (excluding the tag, length, and number of unused bits) of PK.CASD-SIGN.AUT, the key needed to verify the signature.

The value of the `AlgorithmIdentifier.parameters` for the different ECC Curves is described in the following table:

Table 5-7: Value of the `AlgorithmIdentifier.parameters` for the different ECC Curves

Curve	Key Parameter Reference Value (see [GPCS] Table B-2)	Value of `AlgorithmIdentifier.parameters` OID format and DER TLV encoding
ECC curves defined in [FIPS 186-4] and OID defined in [RFC 5480]		
P-256	'00'	OID format: "iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3) prime(1) secp256r1 (7)" DER TLV encoding: '06082A8648CE3D0301'
P-384	'01'	OID format: "iso(1) identified-organization(3) certicom(132) curve(0) secp384r1 (34)" DER TLV encoding: '06052B81040022'
P-521	'02'	OID format: "iso(1) identified-organization(3) certicom(132) curve(0) secp521r1 (35)" DER TLV encoding: '06052B81040023'

Curve	Key Parameter Reference Value (see [GPCS] Table B-2)	Value of `AlgorithmIdentifier.parameters` OID format and DER TLV encoding
ECC curves and OID defined in [RFC 5639]		
brainpoolP256r1	'03'	OID format: "iso(1) identified-organization(3) teletrust(36) algorithm(3) signatureAlgorithm(3) ecSign(2) ecStdCurvesAndGeneration(8) ellipticCurve(1) versionOne(1) brainpoolP256r1(7)" DER TLV encoding: '06092B2403030208010107'
brainpoolP256t1	'04'	OID format: "iso(1) identified-organization(3) teletrust(36) algorithm(3) signatureAlgorithm(3) ecSign(2) ecStdCurvesAndGeneration(8) ellipticCurve(1) versionOne(1) brainpoolP256t1(8)" DER TLV encoding: '06092B2403030208010108'
brainpoolP384r1	'05'	OID format: "iso(1) identified-organization(3) teletrust(36) algorithm(3) signatureAlgorithm(3) ecSign(2) ecStdCurvesAndGeneration(8) ellipticCurve(1) versionOne(1) brainpoolP384r1(11)" DER TLV encoding: '06092B240303020801010B'
brainpoolP384t1	'06'	OID format: "iso(1) identified-organization(3) teletrust(36) algorithm(3) signatureAlgorithm(3) ecSign(2) ecStdCurvesAndGeneration(8) ellipticCurve(1) versionOne(1) brainpoolP384t1(12)" DER TLV encoding: '06092B240303020801010C'
brainpoolP512r1	'07'	OID format: "iso(1) identified-organization(3) teletrust(36) algorithm(3) signatureAlgorithm(3) ecSign(2) ecStdCurvesAndGeneration(8) ellipticCurve(1) versionOne(1) brainpoolP512r1(13)" DER TLV encoding: '06092B240303020801010D'
brainpoolP512t1	'08'	OID format: "iso(1) identified-organization(3) teletrust(36) algorithm(3) signatureAlgorithm(3) ecSign(2) ecStdCurvesAndGeneration(8) ellipticCurve(1) versionOne(1) brainpoolP512t1(14)" DER TLV encoding: '06092B240303020801010E'

6 Commands

6.1 INITIALIZE SECURITY Command

6.1.1 Definition and Scope

The INITIALIZE SECURITY command is used to submit CERT.AP.ECKA in support of Scenario #4, described in section 3.5.5. The certificate is verified by the CASD and contains the Application Provider's public key PK.AP.ECKA, used in the key agreement process. The command data also contains the Application Provider's ephemeral public key, ePK.AP.ECKA.

This command is to be used when no secure channel session has been established.

6.1.2 Command Message

The INITIALIZE SECURITY command message shall be coded according to the following table.

Table 6-1: INITIALIZE SECURITY Command Message

Code	Value	Meaning
CLA	'80' - '83', 'C0' - 'CF'	See [GPCS] section 11.1.4.
INS	'2E'	INITIALIZE SECURITY
P1	'xx'	Command chaining information
P2	'00'	Reference control parameter P2
Lc	'xx'	Length of data field
Data	'xx xx...'	Certificate and ephemeral public key
Le	'00'	

6.1.2.1 Reference Control Parameter P1

Bit 8 is used for the command chaining information, indicating whether the data field of the command is an intermediate or the last block of the command data.

Table 6-2: Values of Reference Control Parameter P2

b8	b7	b6	b5	b4	b3	b2	b1	Description
X								Command chaining information: 1: More command(s) 0: Final (only) command

If the command data do not exceed 255 bytes, the command chaining information (b8) shall be set to 0 (Final command). Otherwise, bit 8 shall be set to 1 (More command(s)) until the last block of the command data is sent.

6.1.2.2 Reference Control Parameter P2

The reference control parameter P2 shall always be set to 0.

6.1.2.3 Data Field Sent in the Command Message

The data field of the command message shall contain part or all of the data described in the following table. Such data may be split across several consecutive INITIALIZE SECURITY commands as described by the P1 parameter.

Table 6-3: INITIALIZE SECURITY Command Data

Tag	Length	Value Description	MOC
'7F21'	Var	Certificate (CERT.AP.ECKA)	M
'7F49'	Var	ePK.AP.ECKA	M

The format of the certificate is provided in Table 3-6, with mandatory Discretionary Data (tag '73' containing the data items defined in Table 3-35). The data signed to generate the certificate are defined in Table 3-7.

The ephemeral public key shall be coded using uncompressed encoding as specified in [TR-03111] section 3.2.1.

6.1.3 Response Message

6.1.3.1 Data Field Returned in the Response Message

The data field of the response is defined in Table 3-37.

6.1.3.2 Processing State Returned in the Response Message

A successful execution of the command shall be indicated by status bytes '90 00'.

This command may either return a general error condition as listed in [GPCS] section 11.1.3 or one of the following error conditions.

Table 6-4: INITIALIZE SECURITY Error Conditions

SW1	SW2	Meaning
'66'	'00'	Verification of the certificate failed
'6A'	'80'	Incorrect values in command data

Annex A ASN.1 Syntax of Output of the AuthoritySignature Interface (Normative)

```
GPCASDSigaturekModule-v1000 DEFINITIONS IMPLICIT TAGS ::=
BEGIN
SignOutputData ::= [APPLICATION 26] SEQUENCE{
  securityEnvironmentTemplate  SecurityEnvironmentTemplate,
  signature [APPLICATION 55] OCTET STRING
}
SecurityEnvironmentTemplate ::= [APPLICATION 27] SEQUENCE {
  clientApplicationInformation [0]SEQUENCE {
    requestedSignatureMode [0]OCTET STRING (SIZE(1)),
    instanceAID [1]OCTET STRING (SIZE(5..16)),
        executableLoadFileAID [3]OCTET STRING (SIZE(5..16)),
    executableLoadFileVersionNumber [4]OCTET STRING
    },
  algorithmIdentifier [1]AlgorithmIdentifier,
  keyIdentifier [2]KeyIdentifier
 }

-- The Algorithm Information as defined in section 4.1.1.2 of RFC5280
AlgorithmIdentifier ::= SEQUENCE {
  algorithm OBJECT IDENTIFIER,
  parameters ANY DEFINED BY algorithm OPTIONAL }

-- The Key Information as defined in section 4.2.1.1 and 4.2.1.2 of
RFC5280
KeyIdentifier ::= OCTET STRING

END
```