**Demo Applet**

APDU Specification

Contents

[1 Introduction 3](#_Toc402784680)

[2 Command List 4](#_Toc402784681)

[3 Status Words 5](#_Toc402784682)

[4 Select Applet 7](#_Toc402784683)

[5 Initialize Update 8](#_Toc402784684)

[6 External Authentication 9](#_Toc402784685)

[7 Put Data 10](#_Toc402784686)

[7.1 PUT DATA (UPDATE KEY) 10](#_Toc402784687)

[7.2 PUT DATA (DESTROY) 13](#_Toc402784688)

[8 Get Data 14](#_Toc402784689)

[9 Get Challenge 16](#_Toc402784690)

[10 Key Pair Generation 17](#_Toc402784691)

[11 Perform Security Operation (PSO) 18](#_Toc402784692)

[12 Manage Security Environment (MSE-RESTORE) 20](#_Toc402784693)

[13 Manage Security Environment (MSE-SET) 21](#_Toc402784694)

[14 Key Agreement 24](#_Toc402784695)

# Introduction

This specification will demonstrate the APDU structure used in all services that Demo Applet has. To know the working procedure and purpose of every service see Demo Applet Service Specification.

The Package ID of Demo Applet is: **6B6F6E61736C66697073**

# Command List

|  |  |  |
| --- | --- | --- |
| **Command** | **INS Byte** | **Standard** |
| GENERATE KEY PAIR | 46h | ISO/IEC 7816-8 |
| MANAGE SECURITY ENVIRONMENT | 22h | ISO/IEC 7816-4 |
| PERFORM SECURITY OPERATIONS | 2Ah | ISO/IEC 7816-8 |
| PUT DATA | DAh | ISO/IEC 7816-4 |
| GET CHALLENGE | 84h | ISO/IEC 7816-4 |
| GET DATA | CAh | ISO/IEC 7816-4 |
| GENERAL AUTHENTICATION | 86h | ISO/IEC 7816-4 |
| INITIALIZED UPDATE | 50h | GP Card Spec-v2.2 |
| EXTERNAL AUTHENTICATE | 82h | GP Card Spec-v2.2 |

# Status Words

| SW1-SW2 | Functionality |
| --- | --- |
| 0x6700 | SW\_WRONG\_LENGTH |
| 0x6901 | SW\_SIGNATURE\_VERIFICATION\_FAILED |
| 0x6902 | SW\_MUDULUS\_NOT\_INITIALIZED |
| 0x6903 | SW\_SE\_NOT\_RESTORED |
| 0x6904 | SW\_ALG\_NOT\_SUPPORTED |
| 0x6905 | SW\_KEYPAIR\_NOT\_SUPPORTED |
| 0x6906 | SW\_KEY\_INITIALIZATION\_FAILED |
| 0x6911 | INPUT\_PARAM\_OUT\_OF\_ALLOWED\_BOUNDS |
| 0x6912 | SW\_KEY\_NOT\_INITIALIZED |
| 0x6913 | SW\_ALG/KEY\_NOT\_SUPPORTED |
| 0x6914 | SW\_SIGNATURE/CIPHER\_NOT\_INITIALIZED |
| 0x6915 | SW\_INCOMING\_MSG\_NOT\_PADDED |
| 0x6982 | SW\_SECURITY\_STATUS\_NOT\_SATISFIED |
| 0x6984 | SW\_DATA\_INVALID |
| 0x6985 | SW\_CONDITIONS\_NOT\_SATISFIED |
| 0x6986 | SW\_COMMAND\_NOT\_ALLOWED |
| 0x69F1 | SW\_APPLET\_DEAD |
| 0x6A80 | SW\_WRONG\_DATA |
| 0x6A81 | SW\_FUNCTION\_NOT\_SUPPORTED |
| 0x6A86 | SW\_INCORRECT\_P1P2 |
| 0x6B00 | SW\_WRONG \_P1P2 |
| 0x6E00 | SW\_CLA\_NOT\_SUPPORTED |
| 0x6D00 | SW\_INS\_NOT\_SUPPORTED |
| 0x6F00 | SW\_INTERNAL\_ERROR |
| 0x9000 | SW\_NO\_ERROR |

# Select Applet

This command is for selecting **Demo-Applet**.

**SELECT: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h-03h |
| INS | A4h |
| P1 | 04h |
| P2 | 00h |
| Lc | Length of AID |
| Data | Instance AID |
| Le | Empty |

**SELECT: Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Empty |
| SW1 - SW2 | Status Bytes |

# Initialize Update

This is the first command among two for performing authentication according to GP SCP-03 protocol.

**INITIALIZE UPDATE: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 80h-83h |
| INS | 50h |
| P1 | 00h |
| P2 | 00h |
| Lc | 08h |
| Data | Host Challenge 8 bytes |
| Le | Empty |

**INITIALIZE UPDATE (SCP-03): Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Key diversification data (10 bytes)  || Key information (3 bytes)  || Card challenge (8 bytes)  || Card cryptogram (8 bytes)  || Sequence counter (3 bytes) |
| SW1 - SW2 | Status Bytes |

# External Authentication

External Authentication is the Second and last command for performing authentication according to GP SCP-03 protocol. Successful completion of this command ensure secure channel between off-card and on-card.

**EXTERNAL AUTHENTICATION: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 84h-87h |
| INS | 82h |
| P1 | 00h |
| P2 | 00h |
| Lc | 10h |
| Data | Host Cryptogram (8 bytes) | C-MAC (8 bytes) |
| Le | Empty |

**EXTERNEL AUTHENTICATION: Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Empty |
| SW1 - SW2 | Status Bytes |

# Put Data

Put Data will do two different things: Updating/loading different keys and destroying the CSPs.

## PUT DATA (UPDATE KEY)

The PUT DATA (UPDATE KEY) command stores externally generated keys to the key containers. All the keys except public key will be entered in encrypted form. For more information about updating/loading keys see “Update Demo key Service” in Demo Applet Service Specification document.

**PUT DATA (UPDATE KEY): Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h-03h |
| INS | DAh |
| P1 | 01h |
| P2 | 01h : DEM\_AUTH\_AES\_128  11h : DEM\_CON \_AES\_128  12h : DEM\_CON \_AES\_192  13h : DEM\_CON \_AES\_256  14h : DEM\_CON \_KEY\_TDES    21h :DEM\_MAC\_TDES  22h :DEM\_MAC\_AES\_128  23h :DEM\_MAC\_AES\_192  24h :DEM\_MAC\_AES\_256  25h :DEM\_MAC\_HMAC    31h :DEM\_PUB\_RSA\_1024\_MOD  32h :DEM\_PUB\_RSA\_2048\_MOD  33h :DEM\_PUB\_RSA\_1024\_EXP  34h :DEM\_PUB\_RSA\_2048\_EXP  35h : DEM\_PUB\_ECDSA  41h :DEM\_WRAP\_AES\_256  **Note:** For additional information see table below for P2 |
| Lc | 81h if P2 = 32h  Otherwise, Length of data field |
| Data | if P2: 32h   * 81h || first half of modulus (128) * 82h || second half of modulus (128)   Otherwise, Key value |
| Le | Empty |

**PUT DATA (UPDATE KEY): Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Empty |
| SW1 - SW2 | Status Bytes |

**Table: P2 for PUT-DATA**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Key/pin** | | **Key variation** | | | **All keys** | | | | **Properties** | | |
| **B7** | | **B6** | **B5** | **B4** | **B3** | **B2** | **B1** | **B0** | **Meaning** | **Hex** | |
| 0 | | 0 | - | - | - | - | - | - | Demo keys | | |
| **AUTH-key** | | | | | | | | | | | |
| 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 1 | AES-128 | 01 | |
| **CON-keys** | | | | | | | | | | | |
| 0 | 0 | | 0 | 1 | 0 | 0 | 0 | 1 | AES-128 | 11 | |
| 0 | 0 | | 0 | 1 | 0 | 0 | 1 | 0 | AES-192 | 12 | |
| 0 | 0 | | 0 | 1 | 0 | 0 | 1 | 1 | AES-256 | 13 | |
| 0 | 0 | | 0 | 1 | 0 | 1 | 0 | 0 | TDES | 14 | |
| 0 | 0 | | 0 | 1 | 1 | x | x | X | RFU | | |
| **MAC-keys** | | | | | | | | | | | |
| 0 | 0 | | 1 | 0 | 0 | 0 | 0 | 1 | TDES | 21 | |
| 0 | 0 | | 1 | 0 | 0 | 0 | 1 | 0 | AES-128 | 22 | |
| 0 | 0 | | 1 | 0 | 0 | 0 | 1 | 1 | AES-192 | 23 | |
| 0 | 0 | | 1 | 0 | 0 | 1 | 0 | 0 | AES-256 | 24 | |
| 0 | 0 | | 1 | 0 | 0 | 1 | 0 | 1 | HMAC | 25 | |
| 0 | 0 | | 1 | 0 | 1 | x | x | X | RFU | | |
| **PUB-keys** | | | | | | | | | | | |
| 0 | 0 | | 1 | 1 | 0 | 0 | 0 | 1 | RSA-1024 mod | 31 | |
| 0 | 0 | | 1 | 1 | 0 | 0 | 1 | 0 | RSA-2048 mod | 32 | |
| 0 | 0 | | 1 | 1 | 0 | 0 | 1 | 1 | RSA-1024 exp | 33 | |
| 0 | 0 | | 1 | 1 | 0 | 1 | 0 | 0 | RSA-2048 exp | 34 | |
| 0 | 0 | | 1 | 1 | 0 | 1 | 0 | 1 | ECDSA | 35 | |
| 0 | 0 | | 1 | 1 | 1 | x | X | X | RFU | | |
| **WRAP-key** | | | | | | | | | | | |
| 0 | 1 | | 0 | 0 | 0 | 0 | 0 | 1 | AES-256 | | 41 |
| 0 | 1 | | 0 | 0 | X | X | X | X | RFU | | |
| 0 | 1 | | X | X | X | X | X | X | RFU | | |
| x | - | | - | - | - | - | - | - | RFU | | |

## PUT DATA (DESTROY)

The PUT DATA (DESTROY) will initialize all the Key Containers with zeros except DEM-AUTH and DEM-KEY-WRAP key containers. These two key containers will initialize with the default authentication and wrap/unwrap key values.

**PUT DATA (DESTROY): Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h-03h |
| INS | DAh |
| P1 | 01h |
| P2 | FFh |
| Lc | 00h |
| Data | Empty |
| Le | Empty |

**PUT DATA (DESTROY): Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Empty |
| SW1 - SW2 | Status Bytes |

# Get Data

The GET DATA command retrieves the RSA Public Key components (Modulus and Exponent) and EC Public Key in plain form or Secret Key and Share Secret in encrypted form. It can also be used to read the Applet Info and current operation state of **Demo-Applet.**

**GET DATA: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h-03h |
| INS | CAh |
| P1 | 01h |
| P2 | OPERATION\_STATE :00h  SECRET\_KEY\_AUTH :01h  SECRET\_KEY\_CON :10h  RSA\_PUB\_KEY\_MOD\_CON :11h  RSA\_PUB\_KEY\_EXP\_CON :12h  SECRET\_KEY\_MAC :20h  RSA\_PUB\_KEY\_MOD\_DS :21h  RSA\_PUB\_KEY\_EXP\_DS :22h  EC\_PUB\_KEY\_DS :23h  SHARE\_SECRET :30h  SECRET\_KEY\_WRAP :40h  APPELT\_INFO :90h |
| LC | Empty |
| Data | Empty |

**GET DATA: Response APDU (**Public key components-p2:11, 12, 21, 22, and23**)**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Public key components |
| SW1 - SW2 | Status Bytes |

**GET DATA: Response APDU (**Operational State -p2:00**)**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | 00h: IDLE STATE  01h: AUTHENTICATE\_STATE  02h: AUTHENTICATE\_IDLE\_STATE  03h: PUT\_DATA\_STATE  04h: SIGNATURE\_VERIFY\_STATE |
| SW1 - SW2 | Status Bytes |

**GET DATA: Response APDU (**Applet Info -p2: 90**)**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **BYTE** | **VALUE** | | | | | | | | | |
| Data | **Tag** | **Length** | **Version** | | | **Tag** | **Length** | **Release Date** | | |
| 10h | 03h | Major | Minor | Revision | 20h | 04h | DD | MM | YY |
| SW1 - SW2 | Status Bytes | | | | | | | | | |

**Note:** Response of EC public key is the point of EC curve represented as an octet string in uncompressed forms as per ANSI X9.62.

# Get Challenge

The GET CHALLENGE command generates DRBG/NDRNG of requested lengths in the card and returns the generated random number.

**Generating DRBG:**

Step-1: Use APDU with tag value 10 in data field to initialize DRBG random generation. This initialization should be done if new seed value is to be used or after sending get challenge APDU with tag value 40h in the data field for DRBG/ NDRNG clear.

Step-2: Use APDU with tag value 30 in data field to generate DRBG random value as many times as required.

Step-3 (Optional): Clear DRBG random generation by sending APDU with tag value 40h.

**Generating NDRNG:**

Step-1: Use APDU with tag value 20 in data field to initialize NDRNG random generation. This initialization should be done after sending get challenge APDU with tag value 40h in the data field for DRBG/NDRNG clear.

Step-2: Use APDU with tag value 30 in data field to generate NDRNG random value as many times as required.

Step-3 (Optional): Clear NDRNG random generation by sending APDU with tag value 40h.

**GET CHALLENGE: APDU command**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLS | 00h-03h |
| INS | 84h |
| P1 | 00h |
| P2 | 00h |
| Lc | Data Length |
| Data | **Tag || Data :**  10h || Seed in encrypted form (DRBG: Init Random Object with Seed)  20h || Empty (NDRNG: Init Random Object without Seed)  30h || Expected Challenge Length (Challenge Length<256)  40h (Clear the random object) |
| Le | Empty |

**GET CHALLENGE: APDU Response**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Random data of specified length |
| SW1 - SW2 | Status Bytes |

# Key Pair Generation

The GENERATE KEYPAIR command generates new key pair and stores within card. Demo-Applet can generate two types of key pair: RSA and EC Key Pair.

**GENERATE KEY PAIR: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h-03h |
| INS | 46h |
| P1P2 | 0000h  Other values RFU |
| Lc | 01h |
| Data | 01h - RSA 2048-bit Key-Pair for Sign/Verify  02h - RSA-CRT 2048-bit Key-Pair for Sign/Verify  03h - RSA 2048-bit Key-Pair for Wrap/Unwrap  04h - RSA- CRT 2048-bit Key-Pair for Wrap/Unwrap  12h - EC P-224 Key-Pair  13h - EC P-256 Key-Pair  14h - EC P-384 Key-Pair  15h - EC P-521 Key-Pair |
| Le | Empty |

**GENERATE KEY PAIR: Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Empty |
| SW1 - SW2 | Status Bytes |

# Perform Security Operation (PSO)

PERFORM SECURITY OPERATION: ENCRYPT-DECRYPT, DIGITAL SIGNATURE, MAC and HASH. To use this command it is necessary to manage current security environment by using a MSE command.

**PSO: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h-03h |
| INS | 2Ah |
| P1 P2 | 8680h: ENCRYPT  8086h: DECRYPT  9E9Ah: DS/MAC  9080h: HASH  00A8h: VERIFY DS/MAC |
| Lc | Data length |
| Data | Data  For P1P2: 00A8,  20h|| full (signed data length < 256) or 1st part (128 bytes) of signed data (signed data length = 256)  40h|| last part(128 bytes) of signed data (signed data length = 256)  10h||Data to be verified (and start verify) |
| Le | Empty |

**PSO: Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Data |
| SW1 - SW2 | Status Bytes |

# Manage Security Environment (MSE-RESTORE)

The MANAGE SECURITY ENVIRONMENT: RESTORE command restores an empty or predefined Security Environment (SE).For further information see “MSE Service” in “Demo Applet User Guidance Manual” document.

**MSE- RESTORE: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h-03h |
| INS | 22h |
| P1 | F3h |
| P2 | 00h |
| Lc | Empty |
| Data | Empty |
| Le | Empty |

**MSE-RESTORE: Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Empty |
| SW1 - SW2 | Status Bytes |

In restore SE process, some predefined algorithm will be set for different security operation.

* Security component for *HASH(Message Digest)* operation will be restored with algorithm *ALG\_SHA\_256*
* Security component for *CONFIDENTIALITY (enc/dec)* operation will be restored with algorithm *ALG\_AES\_BLOCK\_128\_CBC\_NOPAD*
* Security component for *DS/MAC (Digital Signature and MAC)* operation will be restored with algorithm *ALG\_AES\_MAC\_128\_NOPAD*.

# Manage Security Environment (MSE-SET)

The MANAGE SECURITY ENVIRONMENT: SET command sets attributes in the current Security Environment (SE). It requires executing RESTORE command before the execution of first SET command.

**MSE-SET: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h-03h |
| INS | 22h |
| P1 | F1h |
| P2 | B6 : Digital Signature Template (DST)  AA : Hash Code Template (HT)  B8 : Confidentiality Template (CT) |
| Lc | 03 |
| Data | 8001||  IF P2 = B6:   * 01 ALG\_AES\_CMAC\_128 * 02 ALG\_AES\_MAC\_128\_NOPAD * 03 ALG\_DES\_MAC4\_ISO9797\_1\_M2\_ALG3 * 04 ALG\_DES\_MAC4\_ISO9797\_M1 * 05 ALG\_DES\_MAC4\_ISO9797\_M2 * 06 ALG\_DES\_MAC4\_NOPAD * 07 ALG\_DES\_MAC4\_PKCS5 * 08 ALG\_DES\_MAC8\_ISO9797\_1\_M2\_ALG3 * 09 ALG\_DES\_MAC8\_ISO9797\_M1 * 0A ALG\_DES\_MAC8\_ISO9797\_M2 * 0B ALG\_DES\_MAC8\_NOPAD * 0C ALG\_DES\_MAC8\_PKCS5 * 0D ALG\_HMAC\_SHA1 * 0E ALG\_HMAC\_SHA\_256 * 0F ALG\_HMAC\_SHA\_384 * 10 ALG\_HMAC\_SHA\_512 * 11 ALG\_RSA\_SHA\_ISO9796 * 12 ALG\_RSA\_SHA\_PKCS1 * 13 ALG\_RSA\_SHA\_PKCS1\_PSS * 14 ALG\_RSA\_SHA\_RFC2409 * 15 ALG\_RSA\_SHA\_224\_PKCS1 * 16 ALG\_RSA\_SHA\_224\_PKCS1\_PSS * 17 ALG\_RSA\_SHA\_256\_PKCS1 * 18 ALG\_RSA\_SHA\_256\_PKCS1\_PSS * 19 ALG\_RSA\_SHA\_384\_PKCS1 * 1A ALG\_RSA\_SHA\_384\_PKCS1\_PSS * 1B ALG\_RSA\_SHA\_512\_PKCS1 * 1C ALG\_RSA\_SHA\_512\_PKCS1\_PSS * 1D ALG\_ECDSA\_SHA * 1E ALG\_ECDSA\_SHA\_224 * 1F ALG\_ECDSA\_SHA\_256 * 20 ALG\_ECDSA\_SHA\_384 * 21 ALG\_ECDSA\_SHA\_512   IF P2 = AA   * 01ALG\_SHA * 02 ALG\_SHA\_224 * 03 ALG\_SHA\_256 * 04 ALG\_SHA\_384 * 05 ALG\_SHA\_512   IF P2 = B8  01 ALG\_AES\_BLOCK\_128\_CBC\_NOPAD,  02 ALG\_AES\_CBC\_ISO9797\_M1,  03 ALG\_AES\_CBC\_ISO9797\_M2,  04 ALG\_AES\_CBC\_PKCS5,  05 ALG\_AES\_BLOCK\_128\_ECB\_NOPAD,  06 ALG\_AES\_ECB\_ISO9797\_M1,  07 ALG\_AES\_ECB\_ISO9797\_M2,  08 ALG\_AES\_ECB\_PKCS5,  09 ALG\_DES\_CBC\_ISO9797\_M1,  0A ALG\_DES\_CBC\_ISO9797\_M2,  0B ALG\_DES\_CBC\_NOPAD,  0C ALG\_DES\_CBC\_PKCS5,  0D ALG\_DES\_ECB\_ISO9797\_M1,  0E ALG\_DES\_ECB\_ISO9797\_M2,  0F ALG\_DES\_ECB\_NOPAD,   * 10 ALG\_DES\_ECB\_PKCS5   11 ALG\_RSA\_NOPAD,  12 ALG\_RSA\_PKCS1 |
| Le | Empty |

**MSE-SET: Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Empty |
| SW1 - SW2 | Status Bytes |

# Key Agreement

The KEY AGREEMENT command generate share secret and store generated secret to the share secret container.

**Key Agreement: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h-03h |
| INS | 86h |
| P1 | P1 table for key agreement |
| P2 | 00h |
| Lc | Data length |
| Data | Data |
| Le | Empty |

**P1table for key agreement**

|  |  |
| --- | --- |
| **Type** | **VALUE** |
| EC\_FP\_224 | 12h |
| EC\_FP\_256 | 13h |
| EC\_FP\_384 | 14h |
| EC\_FP\_521 | 15h |

**Key Agreement: Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Public key for external entity |
| SW1 - SW2 | Status Bytes |