**Demo Applet**

APDU Specification

# Introduction:

This specification will demonstrate the APDU structure of all services the Demo Applet has. To know the working procedure and purpose of every service see Demo Applet Specification. The AID of Demo Applet is: **6B6F6E61736C66697073**

# Status Words:

|  |  |
| --- | --- |
| **SW1-SW2** | **Functionality** |
| 0x9000 | SW\_NO\_ERROR |
| 0x6E00 | SW\_CLA\_NOT\_SUPPORTED |
| 0x6D00 | SW\_INS\_NOT\_SUPPORTED |
| 0x6B00 | SW\_WRONG \_P1P2 |
| 0x6700 | SW\_WRONG\_LENGTH |
| 0x6A80 | SW\_WRONG\_DATA |
| 0x6982 | SW\_SECURITY\_STATUS\_NOT\_SATISFIED |
| 0x6983 | SW\_AUTH\_METHOD\_BLOCKED |
| 0x6985 | SW\_CONDITIONS\_NOT\_SATISFIED |
| 0x6A81 | SW\_FUNCTION\_NOT\_SUPPORTED |
| 0x6F00 | SW\_INTERNAL\_ERROR |
| 0x6984 | SW\_DATA\_INVALID |
| 0x9003 | SW\_SIGNATURE\_VERIFICATION\_FAILED |
| 0x6901 | SW\_CSP\_NOT\_INITIALIZED |
| 0x6902 | SW\_MUDULUS\_NOT\_INITIALIZED |
| 0x6903 | SW\_SE\_NOT\_RESTORED |
| 0x6904 | SW\_ALGO\_NOT\_SUPPORTED |
| 0x6905 | SW\_EC\_KEY\_NOT\_SUPPORTED |
| 0x6906 | SW\_KEY\_NOT\_INITIALIZED |
| 0x63CX | Verify Fail, X tries left |

# Authentication:

**INITIALIZE UPDATE: Command APDU**

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

**INITIALIZE UPDATE: Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Data |
| SW1 - SW2 | 9000 |

**Authentication:**

**EXTERNEL AUTHENTICATION: Command APDU**

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

**EXTERNEL AUTHENTICATION: Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Empty |
| SW1 - SW2 | 9000 |

# Update Demo Keys:

The PUT DATA: LOAD KEY command stores externally generated keys to the key containers. Demo-Applet has AUTH key, WRAP key, MAC key and 3P PUBLIC key containers.

**PUT DATA: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h |
| INS | DAh |
| P1 | 01h |
| P2 | 01 : DEM\_AUTH\_2KEY\_TDES  11 : DEM\_WRAP\_AES\_128  12 : DEM\_WRAP\_AES\_192  13 : DEM\_WRAP\_AES\_256  14 : DEM\_WRAP\_2KEY\_TDES    21 : DEM\_MAC\_TDES  22 : DEM\_MAC\_AES\_128  23 : DEM\_MAC\_HMAC    31 : DEM\_PUB\_RSA\_1024\_MOD  32 : DEM\_PUB\_RSA\_2048\_MOD  33 : DEM\_PUB\_RSA\_1024\_EXP  34 : DEM\_PUB\_RSA\_2048\_EXP  35 : DEM\_PUB\_ECDSA  36 : DEM\_PUB\_ECDH |
| Lc | Length of data field  If P2: 32  LC = 0x81 |
| Data | Key data  if P2: 32  1st byte 0x81, first half of modulus (128)  1st byte 0x82, second half of modulus (128) |
| Le | Empty |

**PUT DATA: 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 | 2key-TDES | 01 |
| WRAP-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 | HMAC | 23 |
| 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 | 0 | 1 | 1 | 0 | ECDH | 36 |
| 0 | 0 | 1 | 1 | 1 | x | x | x | RFU | |
| x | x | - | - | - | - | - | - | RFU | |

# GET DATA:

The GET DATA command retrieves the RSA Public Key components (Modulus and Exponent), or the EC Public Key.

**GET DATA: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h |
| INS | CAh |
| P1 | 01 |
| P2 | RSA\_PUB\_KEY\_MOD :40  RSA\_PUB\_KEY\_EXP :50  EC\_PUB\_KEY :60 |
| LC | Empty |
| Data | Empty |

**GET DATA: Response APDU**

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

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

# GET CHALLANGE:

The GET CHALLENGE command generates a random number of requested lengths in the card and sends the generated random number.

**GET CHALLENGE: APDU command**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLS | 00h |
| INS | 84h |
| P1 | 00 |
| P2 | 00 |
| Lc | Data Length |
| Data | **Challenge length (one byte) | Tag | Data :**  Challenge Length | 10 | Empty  Challenge Length | 20 | SEED |
| Le | Empty |
| CLS | 00h |

**GET CHALLENGE: APDU Response**

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

# Key Pair Generation:

The GENERATE KEYPAIR command generates and stores new key pair into KeyPair containers. Demo-Applet has two key pair containers for RSA and EC Key Pair.

**GENERATE KEY PAIR: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h |
| INS | 46h |
| P1P2 | 0000h  Other values RFU |
| Lc | 01 |
| Data | 01 - RSA 2048-bit Key-Pair Generation  12 - EC P-224 Key-Pair Generation  13 - EC P-256 Key-Pair Generation  14 - EC P-384 Key-Pair Generation  15 - EC P-521 Key-Pair Generation |
| Le | 00 |

**GENERATE KEY PAIR: Response APDU**

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

# Perform Security Operation (PSO):

**PSO: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h |
| INS | 2Ah |
| P1 P2 | 8680h: WRAP  8086h: UNWRAP  9E9Ah: DS  9080h: HASH  00A8h: VERIFY DS |
| Lc | Data length |
| Data | Data  If P1P2: 00A8,  20 + 1st part of signed data  40 + last part of signed data  10 + Data to be verified (and start verify) |
| Le | Empty |

**PSO: Response APDU**

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

# Manage Security Environment (MSE):

The MANAGE SECURITY ENVIRONMENT: RESTORE command restores an empty or predefined Security Environment (SE).

**MSE- RESTORE: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h |
| 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 cryptographic operation.

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

# Manage Security Environment (MSE):

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 |
| 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\_MAC\_128\_NOPAD //MAC Generation * 02 ALG\_DES\_MAC4\_ISO9797\_1\_M2\_ALG3 * 03 ALG\_DES\_MAC4\_ISO9797\_M1 * 04 ALG\_DES\_MAC4\_ISO9797\_M2 * 05 ALG\_DES\_MAC4\_NOPAD * 06 ALG\_DES\_MAC4\_PKCS5 * 07 ALG\_DES\_MAC8\_ISO9797\_1\_M2\_ALG3 * 08 ALG\_DES\_MAC8\_ISO9797\_M1 * 09 ALG\_DES\_MAC8\_ISO9797\_M2 * 0A ALG\_DES\_MAC8\_NOPAD * 0B ALG\_DES\_MAC8\_PKCS5 * 0C ALG\_HMAC\_SHA1 * 0D ALG\_HMAC\_SHA\_256 * 0E ALG\_HMAC\_SHA\_384 * 0F ALG\_HMAC\_SHA\_512 * 10 ALG\_RSA\_SHA\_ISO9796 //Signature * 11 ALG\_RSA\_SHA\_PKCS1 //Signature * 12 ALG\_RSA\_SHA\_PKCS1\_PSS * 13 ALG\_RSA\_SHA\_RFC2409 * 14 ALG\_RSA\_SHA\_224\_PKCS1 * 15 ALG\_RSA\_SHA\_224\_PKCS1\_PSS * 16 ALG\_RSA\_SHA\_256\_PKCS1 * 17 ALG\_RSA\_SHA\_256\_PKCS1\_PSS * 18 ALG\_RSA\_SHA\_384\_PKCS1 * 19 ALG\_RSA\_SHA\_384\_PKCS1\_PSS * 1A ALG\_RSA\_SHA\_512\_PKCS1 * 1B ALG\_RSA\_SHA\_512\_PKCS1\_PSS * 1C ALG\_ECDSA\_SHA * 1D ALG\_ECDSA\_SHA\_224 * 1E ALG\_ECDSA\_SHA\_256 * 1F ALG\_ECDSA\_SHA\_384 * 20 ALG\_ECDSA\_SHA\_512   IF P2 = AA   * 01 ALG\_SHA * 02 ALG\_SHA\_256 * 03 ALG\_SHA\_384 * 04 ALG\_SHA\_512 * 05 ALG\_SHA\_224   IF P2 = B8   * 01 ALG\_AES\_BLOCK\_128\_CBC\_NOPAD * 02 ALG\_AES\_BLOCK\_128\_ECB\_NOPAD * 03 ALG\_AES\_CBC\_ISO9797\_M2 * 04 ALG\_AES\_CBC\_PKCS5 * 05 ALG\_AES\_ECB\_ISO9797\_M2 * 06 ALG\_AES\_ECB\_PKCS5 * 07 ALG\_DES\_CBC\_ISO9797\_M1 * 08 ALG\_DES\_CBC\_ISO9797\_M2 * 09 ALG\_DES\_CBC\_NOPAD * 0A ALG\_DES\_CBC\_PKCS5 * 0B ALG\_DES\_ECB\_ISO9797\_M1 * 0C ALG\_DES\_ECB\_ISO9797\_M2 * 0D ALG\_DES\_ECB\_NOPAD * 0E ALG\_DES\_ECB\_PKCS5 |
| Le | Empty |

**MSE-SET: Response APDU**

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

# Key Agreement:

**Key Agreement: Command APDU**

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

**P1 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 + ’00 00 00’ + generated Secret |
| SW1 - SW2 | Status Bytes |

# RESET RETRY COUNTER:

**RESET RETRY COUNTER: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h |
| INS | 2Ch |
| P1 | 00h: reset retry counter and set reference data  01h: reset retry counter  02h: set reference data  03h: get PIN verification status (9000h or 6982h) |
| P2 | *PIN reference number*  00h: SO\_PIN  01h: USER\_PIN |
| Lc | 00h or Length of data field |
| Data | * Empty (P1 set to 03h) * Unblocking reference data padded (00h) to 8 bytes, followed by the new reference data padded (00h) to 8 bytes (P1 set to 00h) * Unblocking reference data padded (00h) to 8 bytes (P1 set to 01h) * New reference data padded (00h) to 8 bytes (P1 set to 02h) |
| Le | Empty |

**RESET RETRY COUNTER Response APDU**

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

# VERIFY:

**VERIFY: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h |
| INS | 20h |
| P1 | 00h |
| P2 | *PIN reference number*  00h: SO\_PIN  01h: USER\_PIN |
| Lc | Length of data field |
| Data | PIN reference data (verification data) padded (00h) to 8 bytes |
| Le | Empty |

**VERIFY: Response APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| Data | Empty |
| SW1 - SW2 | 63CX : X is the number of tries remains(Max 15) |

**DESTROY:**

**DESTROY: Command APDU**

|  |  |
| --- | --- |
| **BYTE** | **VALUE** |
| CLA | 00h |
| INS | 30h |
| P1P2 | 0000h |
| Lc | Empty |
| Data | Empty |
| Le | Empty |

**DESTROY: Response APDU**

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