Author: Nguyen Minh Khoa

Read Contactless IC Chip on Vietnamese Identification Card by BAC (Basic Access Control)

TABLE OF CONTENT

I. BAC procedure

II. Extract MRZ information

III. Derive the Document Basic Access Keys from K_{seed}

IV. Select Application

V. Authentication

VI. Secure Messaging

VII. 3DES Modes of Operation

VIII. Reference

IX. Appendix

I. BAC procedure

1. Derive the Document Basic Access Keys (K_{Enc} and K_{MAC}) from the MRZ.
2. Select eMRTD Application Authenticate using the Document Basic Access Keys and derive session keys KS_{Enc} and $KS_{MAC.}$
3. Start Secure Messaging to access EF.COM (Common Data), EF.DG1 (MRZ Data), EF.DG2 (Face Data), and EF.DG13 (Extra Personal Data).

II. Extract MRZ Information

- MRZ_information = 9 last digits of doc number || 1 check digit || 6 digits of birth || 1 check digit || 6 digits of expiring date || 1 check digit.

Figure 1. An example of extracting MRZ_information

- In ‘Image 1’, MRZ_information = “044009261144070219912315”.
- ICAO 9303 – Part 3 defines the check digit’ calculation.

III. Derive the Document Basic Access Keys from K_{seed}

1. Calculate K_{seed}
   - Calculate the SHA-1 hash of ‘MRZ_information’.
   - K_{seed} = first 16 bytes of the hash.
2. Concatenate K_{seed} and c:
   - $D = K_{seed} \| c$ (c = ‘00000001’ for K_{Enc} and c = ‘00000002’ for K_{MAC})
3. Calculate the SHA-1 hash of D.
4. 3DES key K_{Enc} and K_{MAC} = first 16 bytes of each hash (See section VII for 3DES Modes of Operations).

IV. Select Application

1. Command APDU: 0x00 || 0xA4 || 0x04 || 0x00 || 0x07 || 0xA0 || 0x00 || 0x00 || 0x02 || 0x47 || 0x10 || 0x01 (AID for BAC application)
2. Expected response APDU: 0x90 || 0x00

V. Authentication

1. GET CHALLENGE
   - Command APDU: 0x00 || 0x84 || 0x00|| 0x00 || 0x08
   - Expected response APDU: RND.IC (8 bytes) || 0x90 || 0x00
2. Generate an 8-byte random RND.IFD and a 16 byte random K_{IFD}.
3. S = RND.IFD || RND.IC || K_{IFD}
4. Encrypt S with 3DES key K_{Enc}
   - $E_{IFD} = E(K_{Enc}, S)$
5. Compute MAC over E_{IFD} with 3DES key K_{MAC}
   - $M_{IFD} = MAC(K_{MAC}, E_{IFD})$
6. Construct command data for EXTERNAL AUTHENTICATE
   - cmd_data = E_{IFD} || M_{IFD}
7. EXTERNAL AUTHENTICATE
   - Command APDU: 0x00 || 0x82 || 0x00 || 0x00 || 0x28 || cmd_data || 0x28
   - Expected response APDU: E_{IC} (32 bytes) || M_{IC} (8 bytes) || 0x90 || 0x00
8. Check the checksum M_{IC} of the cryptogram E_{IC} with $K_{MAC.}$
9. Decrypt the cryptogram E_{IC}
   - R = $D(K_{Enc}, E_{IC})$ = RND.IC (8 bytes) || RND.IFD (8 bytes) || K.IC (16 bytes)
10. Compare RND.IFD with the generated RND.IFD to check if IC returned the correct value.
11. Session key seed $KS_{seed} = K_{IFD}$ xor K_{IC}
12. Derive session keys KS_{Enc} and KS_{MAC} from KS_{seed} with the same mechanism shown in III.
13. Calculate ‘Send Sequence Counter’
    - SSC = 4 last bytes of RND.IC || 4 last bytes of RND.IFD

VI. Secure Messaging

1. Specifications
   - SSC needs to be increased by one every time before sending a command APDU and after receiving a response APDU.
   - Data encryption uses 3DES with KS_{Enc}.
   - MAC computation uses KS_{MAC}
2. Compute protected SM command

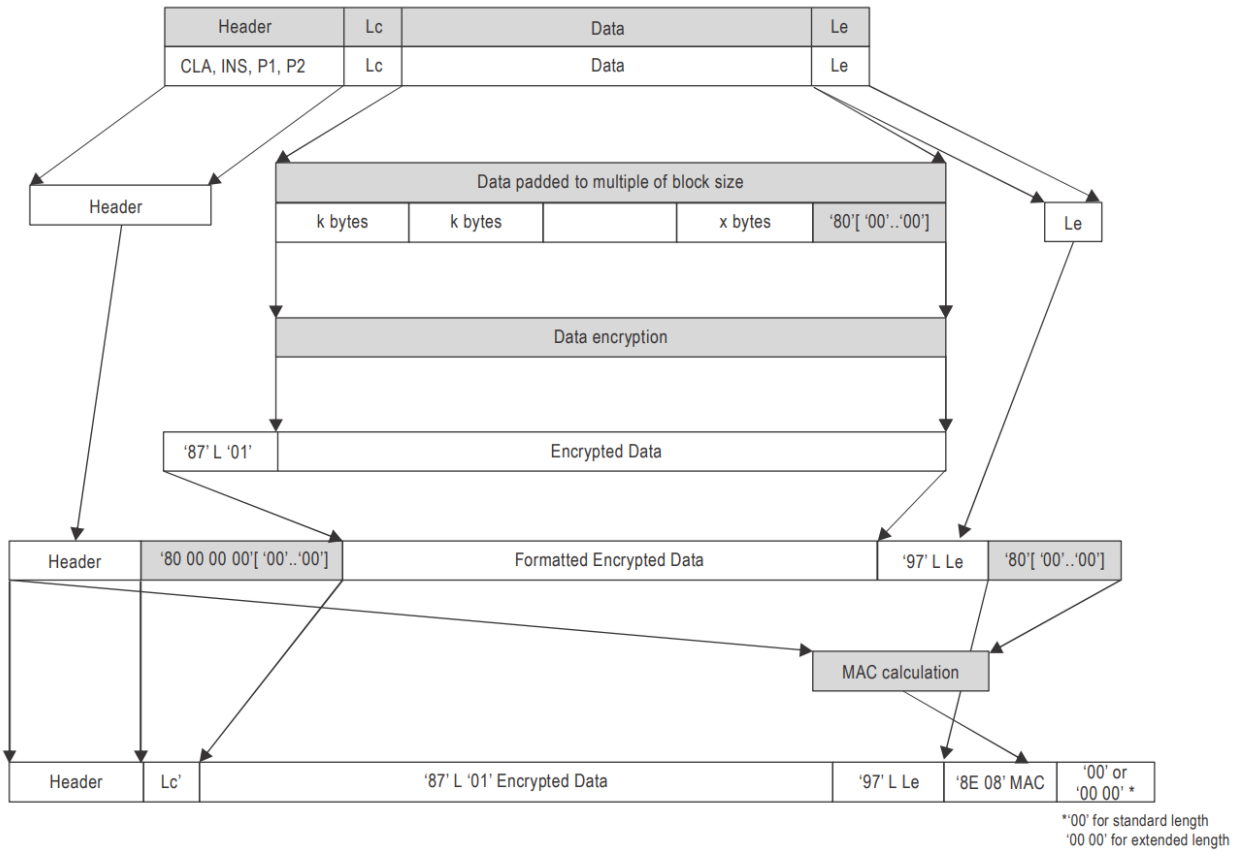

Figure 2. Computation of an SM command APDU for even INS Byte

- If INS is odd, use DO‘85’ instead of DO‘87’.
- The command header must be included in the MAC calculation. Therefore, the class byte CLA = 0x0C must be used.
- Figure 2 shows the transformation of an unprotected command APDU to a protected command APDU in the case Data and Le are available. If no Data is available, leave building DO‘87’ out. If Le is not available, leave building DO‘97’ out.
- Use L = 0x09 in DO‘87’ and L = 0x01 in DO‘97’.
- L_c’ = number of following bytes in the SM command excluding the last ‘00’ byte.
- MAC = MAC(SSC || Padded header || DO‘97’ || Padding)

3. Compute Protected SM response

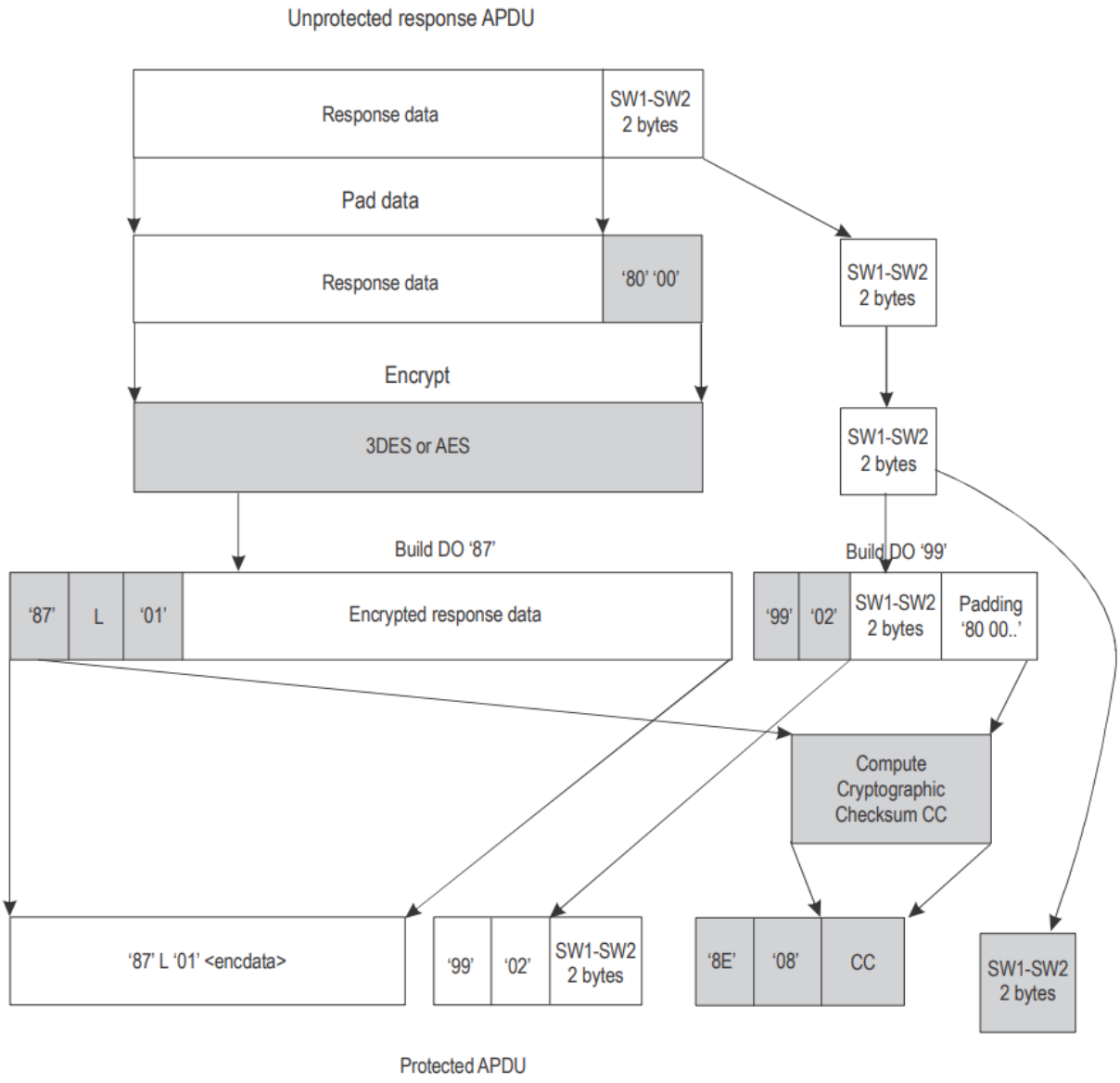

Figure 3. Computation of an SM response APDU for even INS Byte

- Figure 3 shows the transformation of an unprotected response APDU to a protected response APDU in case Data is available. If no Data is available, leave building DO‘87’ out.
- Compute MAC to compare with DO‘8E’ = MAC(SSC || DO‘87’ || DO‘99’)

4. SELECT
   - Unprotected command header: 0x0C || 0xA4 || 0x02 || 0x0C
   - Unprotected command $L_C = 2$
   - Unprotected command data field: File identifier (2 bytes)
   - No L_e
5. READ BINARY

- Unprotected command header: 0x0C || 0xB0 || P1 || P2
- P1 and P2 define the location of the byte where it will start reading (e.g., P1 = 0x00, P2 = 0x00 means it will start reading at the 1st byte of the file, P1 = 0x01, P2 = 0x03 means it will start reading at the 260th byte of the file,…)
- L_e defines the number of bytes it will read (0x00 = 256 bytes = max)
- No data field

6. EF.COM (‘01 1E’)

- Located in the LDS1 eMRTD application and contains LDS version information, Unicode version information, and a list of the Data Groups that are present for the application (For Vietnamese Identification Card, LDS version is 0107, Unicode version information is 040000, and Data Groups that are present for the application are 1, 2, 3, 13, 15, 14).

Tag	L	Value		
‘60’	Var	Application level information		
		Tag	**L**	**Value**
		‘5F01’	‘04’	LDS Version number with format aabb, where aa defines the version of the LDS and bb defines the update level.
		‘5F36’	‘06’	Unicode Version number with format aabbcc, where aa defines the major version, bb defines the minor version and cc defines the release level.
		‘5C’	Var	Tag list. List of all Data Groups present.

Figure 4. EF.COM Normative Tags

7. Data Group 1 (‘01 01’)

- The Data Elements of Data Group 1 (DG1) are intended to reflect the entire contents of the MRZ, whether it contains actual data or filler characters.

Tag	L	Value		
‘61’	Var			
		Tag	**L**	**Value**
		‘5F1F’	Var	The MRZ data object as a composite Data Element. (REQUIRED) (The Data Element contains all mandatory fields from Document Type through to Composite check digit.)

Figure 5. Data Group 1 Tags

Data Element	Optional or MANDATORY	Name of Data Element	Number of Bytes	Fixed or Variable	Type of Coding
01	M	Document code	2	F	A,S
02	M	Issuing State or organization	3	F	A,S
03	M	Document number (Nine most significant characters)	9	F	A,N,S
04	M	Check digit — Document number or filler character (<) indicating document number exceeds nine characters	1	F	N,S
05	M	Optional data and/or in the case of a Document Number exceeding nine characters, least significant characters of document number plus document number check digit plus filler character	15	F	A,N,S
06	M	Date of birth	6	F	N,S
07	M	Check digit — Date of birth	1	F	N
08	M	Sex	1	F	A,S
09	M	Date of Expiry	6	F	N
10	M	Check digit — Date of expiry	1	F	N
11	M	Nationality	3	F	A,S
12	M	Optional data	11	F	A,N,S
13	M	Composite check digit	1	F	N
14	M	Name of holder	30	F	A,N,S

Figure 6. Data Elements of DG1 for TD1 format

8. Data Group 2 (’01 02’)

- Data Group 2 (DG2) represents the image of the face of the holder as an input to a face recognition system.

Tag	L	Value
‘75’	Var	See Biometric encoding of EF.DG2

Figure 7. Data Group 2 Tags

Tag	L	Value				
'7F61'	Var	Biometric Information Template Group Template				
		Tag	**L**	**Value**		
		'02'	'01'	Integer — Number of instances of this type of biometric		
		'7F60'	Var	1st Biometric Information Template		
			Tag	**L**		
			'A1'	Var	Biometric Header Template (BHT)	
				Tag	**L**	**Value**
				'80'	'02'	ICAO header version 0101 (Optional) — Version of the CBEFF patron header format
				'81'	'01-03'	Biometric type (Optional)
				'82'	'01'	Biometric sub-type Optional for DG2
				'83'	'07'	Creation date and time (Optional)
				'85'	'08'	Validity period (from through) (Optional)
				'86'	'04'	Creator of the biometric reference data (PID) (Optional)
				'87'	'02'	Format owner (REQUIRED)
				'88'	'02'	Format type (REQUIRED)
			'5F2E' or '7F2E'	Var	Biometric data (encoded according to Format Owner) also called the biometric data block (BDB).	

Figure 8. Data Group 2 — Biometric Encoding Tags

Data Element	Optional or MANDATORY	Name of Data Element	Number of Bytes	Fixed or Variable	Type of Coding	Coding Requirements
01	M	Number of face biometric encodings recorded	1	F	N	1 to 9 identifying number of unique encodings of data on the face.
02	M	Header		Var	A,N	Data Element may recur as defined by Data element 01.
03	M	Face biometric data encoding(s)		Var	B	Data Element may recur as defined by Data element 01.

Figure 9. Data Elements of DG2

- For the Vietnamese Identification Card, the first 2 bytes of DG2 are 0x75 and 0x82. The 2 bytes after 0x75 || 0x82 will denote the number of bytes of DG2.
- The format of the image encoded is JPEG; therefore, an array of 4 bytes 0xFF || 0xD8 || 0xFF || 0xE0 should appear, and all bytes from that 4 bytes to the end of DG2 is the encoded data of the JPEG file that can directly write the binary to a JPEG file to get the image.

9. Data Group 13 (’01 0D’)

- Data Group 13 (DG13) stores extra information about the card’s holder and is encoded in UTF-8 to be able to store Vietnamese characters.
- The third and fourth bytes indicate the file length.
- The information is stored in the following order: Card ID, Full name, Date of birth, Gender, Nationality, Ethnicity, Religion, Place of origin, Place of residence, Personal identification, Issued date, Expiration date, Father’s name, Mother’s name, Old ID.

VII. 3DES Modes of Operation

1. Encryption
   - Two key 3DES in CBC mode with zero IV (i.e., 0x00 00 00 00 00 00 00 00) and padding method 2 is used.
2. Message Authentication
   - Cryptographic checksums are calculated using MAC Algorithm 3 with block cipher DES, zero IV (8 bytes), and padding method 2. The MAC length MUST be 8 bytes.
3. MAC Algorithm 3 specifications
   - Padding method 2
     + Add 0x80 to the end of the data, then if necessary, add 0x00 until the number of bytes is divisible by 8.
   - Splitting

+ The padded data D is split into q blocks D1, D2, ... Dq, each of length 8

   - Initial transformation 1
     + D_1 is encrypted with the key K:

       $H_1 = E(K, D_1)$
   - Iteration
     + Blocks H_2 ... H_q are calculated by encrypting, with the key K, the bitwise exclusive-or of the corresponding data block and the previous H block:

       for i = 2 to q

       $H_i = E(K, D_i \text{ xor } H_{i-1})$
   - Output transformation 3
     + Hq is decrypted with the key K′ and the result is encrypted with the key K:

       $G = E(K, D(K', H_q))$
   - Truncation
     + Get only the first 8 bytes

VIII. Reference:

1. ICAO 9303 – Part 3: Specifications Common to all MRTDs
2. ICAO 9303 – Part 10: Logical Data Structure (LDS) for Storage of Biometrics and Other Data in the Contactless Integrated Circuit (IC)
3. ICAO 9303 – Part 11: Security Mechanisms for MRTDs
4. ISO/IEC 7816-3:2006, Identification cards — Integrated circuit cards — Part 3: Cards with contacts — Electrical interface and transmission protocols
5. ISO/IEC 7816-4:2013, Identification cards — Integrated circuit cards — Part 4: Organization, security and commands for interchange
6. ISO/IEC 14443-4:2016, Identification cards — Contactless integrated circuit(s) cards — Proximity cards — Part 4: Transmission protocol
7. ISO/IEC 9797-1:2011, Information technology — Security techniques — Message Authentication Codes (MACs) — Part 1: Mechanisms using a block cipher