Composition de Crypto - 2012

La composition est composée de plusieurs exercices

Le support de cours est permis. La majeure partie des questions font appel au sens pratique.

1. Attaque Meet in the Middle

Le but de cet exercice est d'étudier le nombre de clés nécessaires pour décrypter une clés double-DES.

Rappel on appelle un double DES, l'implémentation suivante :

$M' = 2DES_{K1,K2}(M) = DES_{K1}(DES_{K2}(M))$

a) Rappeler la valeur de la taille d'une clé DES et en déduire la valeur de la taille d'une clé double-DES.
b) En déduire le nombre maximum de clés nécessaires pour une attaque de type Brute Force Attack sur un double-DES.
c) On va supposer que nous avons deux couples de blocs (B1, B'1) et (B2, B'2) respectivement bloc en clair et bloc chiffré connus. (B est en clair et B' chiffré). Pour améliorer l'attaque précédente, on élabore le schéma suivant :

- On chiffre avec l'ensemble des clés $K_{2,i}$ le bloc B_1. Soit $B''_{1,i}$ ces valeurs qu'on sauvegarde dans un tableau.
- On décrypte avec l'ensemble des clés $K_{1,j}$ le bloc B'_1. Soit $B'''_{1,j}$ ces valeurs.
- Pour chacune des valeurs $K_{1,j}$ on cherche la valeur $K_{2,i}$ tel que $B'''_{1,j} = B''_{1,i}$.
- Les couples de clés ($K_{1,j}$; $K_{2,i}$) ainsi obtenues sont des valeurs possibles de clés.

Questions :

1. Quelle est la taille en octets du tableau des clés $K_{2,i}$?
2. Quel est le nombre des couples de clés ($K_{1,j}$; $K_{2,i}$) ?
3. Quel est le nombre de clés testées ?
4. A partir de ces couples comment trouver la bonne valeur de K1 et K2 ?
5. Quel est dans ce cas l'amélioration par rapport à la Brute Force Attack ?

2. Cas réel – Dédouanement avec Signature PKCS7

Une entreprise E2 exporte des marchandises d'un pays P1 à un une entreprise E2 dans un pays P2 en utilisant les services d'un transporteur T. Le transporteur T va dédouaner les marchandises chez la douane D du pays P2, puis les emmener au destinataire E2. On suppose que la douane du pays E1 n'intervient pas dans le processus.
On suppose que les documents accompagnant les marchandises ont été dématérialisés.
Le document d'envoi est signé par E1 en utilisant PKCS7-SignedData.
Le transporteur T va re-signer le document signé en indiquant une date de transfert.
La douane D re-signe le document en indiquant une référence et une date de dédouanement.

a) Quel est le niveau hiérarchique de la signature de T ? (co-signature ou sur-signature) ? (1 ligne)
b) Dans quelle zone PKCS7-SignedData peut être mise la date ? (1 ligne)
c) Quelle partie du standard PKCS décrit la date ? (1 ligne)
d) Quel est le niveau hiérarchique de la signature de D ? (co-signature ou sur-signature) ? (1 ligne)
e) Dans quelle zone PKCS7-SignedData peut être mise la référence et la date de dédouanement ? (1 ligne)
f) Quelle partie du standard PKCS décrit la référence ? (1 ligne)
g) Décrire les diverses preuves que constituent les diverses signatures. (une ligne par signature).

Le transporteur livre la marchandise à E2. Un bon de livraison est signé Pkcs7-SignedData par E2.
T va envoyer les documents signés de dédouanement et un bon de livraison à E1.

h) Que doit inclure le bon de livraison comme information minimale pour être signée ? (une ligne)
i) Pour quelle raison T peut ne pas re-signer les documents. (une ligne)
j) Les deux documents sont transmis sous la forme d'un document unique. Que proposez-vous pour les transformer les deux documents en un document unique sans perdre les signatures ? (une ligne)
k) Le document unique est transmis chiffré à E1. Quelle enveloppe proposez-vous ? (une ligne)

3. Reverse-Engineering de la Carte DX+ d'Etebac5

Le but de cet exercice est de faire le reverse-engineering de la carte DX+ utilisée dans le cadre de l'ancien standard Etebac5. Pour cela on va utiliser la documentation de cette carte et de faire quelques manipulations avec la carte en envoyant des ordres transparents ou **APDU** à cette carte.
Le dialogue avec la se fait via un lecteur Pc/Sc.
Le fichier CHV1 est le fichier contenant le PIN.

MASTER FILE

The Master File contains all the headers of all other files in the card. Depending upon the current access conditions, it is possible to read this file and obtain a directory of the file system of the DX card.

Each header is 16 bytes long and contains the data as shown in table 3-01.

TABLE 3-01 FILE HEADER FORMAT

Byte	Field	Length in bytes
1	the file ID	2
3	the type of file	1
4	pointer to the start of the file	2
6	record length	1
7	maximum size of the file	2
9	reserved for future use	2
11	Access conditions	3
14	invalidation flags	1
15	current end of the file	2

File identity
This is a parameter of the SELECT instruction to make a file the currently selected file. Thus if more than one EF with the same ID were to be created, only the first would be accessible. This field labels the file. The use of each file can also be ascertained from the file identity.

The file identities of regularly used EFs are reserved. The list is shown in table 3-02.

TABLE 3-02 RESERVED FILE IDENTITIES

File ID (hex)	File name	Usage
3F00	Master File	mandatory
0000	EF_{CHV1}	optional
0100	EF_{CHV2}	optional
0001	EF_{KEY1}	optional
0101	EF_{KEY2}	optional
0002	EF_{IC}	mandatory
0003	EF_{ID}	optional
0004	EF_{NAME}	optional
0005	EF_{LANG}	optional
0006	EF_{DIR}	optional
0007	EF_{IP}	mandatory
FFFF		forbidden

File type
This indicates the type of data contained in the file. The DX card offers two types of file. These are specified as:
Elementary Files storing binary information (EF_{BINARY}). No special formatting rules are imposed. The value used for this is 00 (zero).
The directory file (MF). This can be read providing a directory of the DX card. The value for this is 10 (hexadecimal A).

Pointer to start of file
This indicates where the file starts in the EEPROM memory. All instructions which operate on files in utilisation phase use logical addressing (see chapter 5). All addresses are relative to this pointer so address 0 will be the start of the file.

Record length
In the Master File header, this field defines the size of the file headers. For Elementary Files, this field is not used but it should be set to 1 for compatibility with future versions of the DX Smart Card. This will imply that binary EFs are accessed with a record length of one byte.

File size
Elementary Files (EF) have the maximum size specified in the header. This is limited to EEPROM capacity. All accesses to files will be checked with respect to this size to prevent reading outside the file area.

Access conditions
Three bytes are allocated, one nibble for each type of access possible in the DX card. The possible accesses are:
- reading or searching memory
- updating an area (erasure followed by a write)
- first write or execution (create)
- invalidation

Two currently unused nibbles are reserved for future use. The layout is shown in figure 3-01

FIGURE 3-01 ACCESS CONDITION FIELDS

READ	UPDATE	RFU	CREATE/ EXEC	RFU	INVALIDATE

The access conditions are all four bits in length and have the following values:
- 0 – Always possible
- 1 – requires previous presentation of the PIN
- 2-E_H – reserved for future use
- F_H – Never possible.

The control system checks the relevant byte before executing any instruction which results in access to the memory of the DX card.

Transcription Importante:
Une nibble = un digit de 4 bits.
Les nibbles non utilisés sont mis à F
Les droits d'accès sont sur 3 octets formés de 6 nibbles et sont dans l'ordre :

- Read
- Update (Write)
- Rfu = F
- Create ou Exec
- Rfu = F
- Invalidate

Invalidation flags

When a file is to be locked from further modification, it is invalidated by updating this field. The layout is shown in figure 3-02.

FIGURE 3-02 INVALIDATION FLAGS

b7	b6	b5	b4	b3	b2	b1	b0
RFU	RFU	RFU	RFU	RFU	RFU	RWI	INV

Only two bits are used in this byte:
INV set to 0 indicates the file is invalidated
RWI set to 1 indicates the file can be read when invalidated.

End of file

Files may be appended to up to the maximum file size. The end of file indicates the last actual data byte in the file and so points to the first location which will be written to by an append operation (CREATE BINARY).

File access

The file access conditions, which are written in the header of each file, apply to the utilisation phase of the DX card life. The table below, table 4-01, shows how these conditions are modified in the personalisation and invalidation phases. The value KP is the personalisation key which overwrites the fabrication key.

TABLE 4-01 FILE ACCESS CONDITIONS

INSTRUCTION GROUP	UTILISATION	PERSONALISATION	INVALIDATION
READ	always	always	always
	CHV1	presentation of KP	CHV1 or never*
	never	never	never
UPDATE	any condition	presentation of KP	never
CREATE/EXECUTE	any condition	presentation of KP	never
INVALIDATE	any condition	never	never

Fabrication data file

The file EF_{IC} contains the data written to the card by the integrated circuit supplier in the fabrication phase. When the logical file structure is written, the header for this file is written with a pointer to the fabrication data in the fabrication zone. The format of the data in this zone and hence this file is shown in table 3-03. Refer to chapter 2 for a description of these fields.

TABLE 3-03 FABRICATION DATA

Byte	Field	Length in bytes
1	chip manufacturer ID	1
2	type of component	1
3	type of mask	1
4	batch number	1
5-6	reserved for future use	1
7-8	card manufacturer ID	3
9	reserved for future use	1
10-13	serial number of component	4
14	profile	1
15-16	reserved for future use	2

TABLE 8-01 SW2 FOR SECURITY MANAGEMENT ERRORS

SW1 (Hex)	SW2 (Hex)	ERROR DESCRIPTION
98	02	No EF_{CHV} defined
98	04	Access conditions not fulfilled
		PIN verify rejected
		Wrong unblocking key
98	08	The PIN is not blocked
98	10	File currently selected is invalidated
98	40	PIN is blocked
98	80	PIN unblocking key is blocked (PIN is irreversibly blocked)

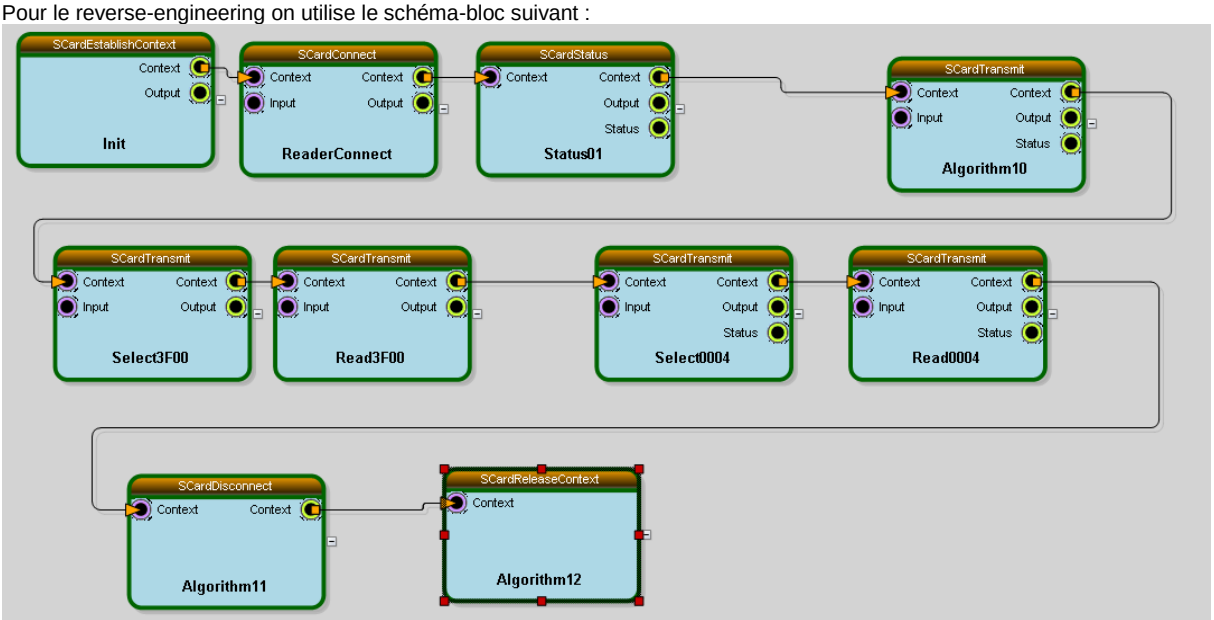
TABLE 8-02 SW2 FOR REFERENCING MANAGEMENT ERRORS

SW1 (Hex)	SW2 (Hex)	ERROR DESCRIPTION
94	00	No EF selected as current
		EF not selected
94	02	Out of range
94	04	not found (ID or pattern)
94	08	Current file-type is inconsistent with the instruction

Rappel du théorème des restes chinois :

N = p *q , S : exposant privé, v exposant publique.
Sp = S mod (p-1)
Sq = S mod (q-1)
p2 = (M mod p) exp Sp mod p
q2 = (M mod q) exp Sq mod q
u = (p exp -1 mod q)
alors Signature = p2 + (u * (q2-p2)) mod q * p

Pour le reverse-engineering on utilise le schéma-bloc suivant :



a) A quoi servent les deux premiers blocs et les deux derniers blocs du schéma ?
b) Le bloc nommé Status01 donne la sortie suivante :

20120402:120042 Entering Status01
Output (15 Bytes) ← réponse de la carte
3B FA 11 00 02 40 60 43 C6 02 F8 03 03 00 00
SW : 90 00

- A quoi peuvent correspondre la suite des octets des valeurs Atrributes ?

c) La trace d'Algorithm10 est la suivante :

```
20120402:120042 Entering Algorithm10(PCSC)
APDU (5 Bytes): 00 B0 00 00 80
Output (128 Bytes)
00 02 00 00 08 01 00 10 00 00 0F FF FF FF 00 10
00 07 00 02 28 01 00 10 00 00 0F FF FF FF 00 10
00 00 00 02 38 01 00 17 00 00 FF FF FF FF 00 17
00 06 00 02 4F 01 00 34 00 00 0F FF FF FF 00 34
00 04 00 02 83 01 00 24 00 00 01 FF FF FF 00 24
00 19 00 02 A7 01 00 84 00 00 1F FF FF FF 00 84
00 1A 00 03 2B 01 01 2A 00 00 1F FF FF FF 01 2A
00 1B 00 04 55 01 01 2A 00 00 1F FF FF FF 01 2A
SW: 90 00
```

- Quel type d'ordre transparent correspond l'APDU ?

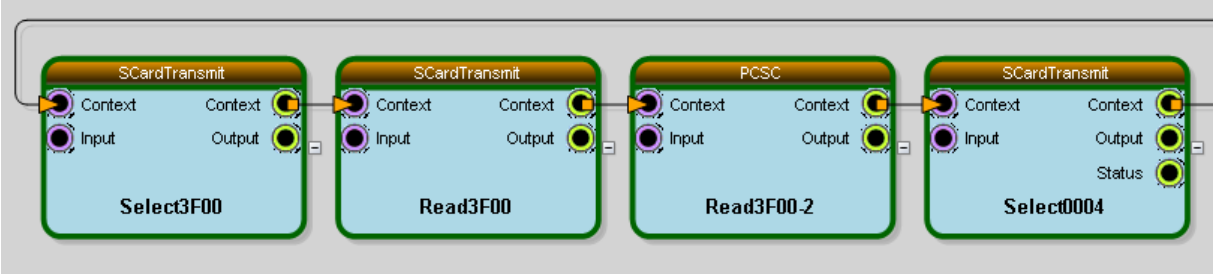
d) La trace de Select3F00 et Read3F00 est la suivante :

```
20120402:120042 Entering Select3F00(PCSC)
APDU (7 Bytes): 00 A4 00 00 02 3F 00
Output (0 Bytes)
SW: 90 00
20120402:120042 Exiting Select3F00(PCSC) ReturnCode=0
20120402:120042 Entering Read3F00(PCSC)
APDU (5 Bytes): 00 B0 FF FF 80
Output (128 Bytes)
00 02 00 00 08 01 00 10 00 00 0F FF FF FF 00 10
00 07 00 02 28 01 00 10 00 00 0F FF FF FF 00 10
00 00 00 02 38 01 00 17 00 00 FF FF FF FF 00 17
00 06 00 02 4F 01 00 34 00 00 0F FF FF FF 00 34
00 04 00 02 83 01 00 24 00 00 01 FF FF FF 00 24
00 19 00 02 A7 01 00 84 00 00 1F FF FF FF 00 84
00 1A 00 03 2B 01 01 2A 00 00 1F FF FF FF 01 2A
00 1B 00 04 55 01 01 2A 00 00 1F FF FF FF 01 2A
SW : 90 00
```

e) Comparer la sortie de Read3F00 et Algorithm10 ? En déduire le premier fichier choisi lors de la mise sous tension de la carte ?

f) On remplace l'APDU 00 B0 FF FF 80 par 00 B0 FF FF 90. La carte répond par SW : 67 00. Quelle est la signification de cet état ?

On insère un bloc supplémentaire Read3F00-2 entre Read3F00 et Select0004. Read3F00-2 envoie le même ordre transparent à la carte que Read3F00.



APDU (5 Bytes):00 B0 FF FF 80
Output (128 Bytes)

```
00 1C 00 05 7F 01 01 2A 00 00 1F FF FF FF 01 2A
00 1D 00 06 A9 01 01 2A 00 00 1F FF FF FF 01 2A
00 0A 00 08 00 01 01 80 00 00 FF F1 FF FF 01 47
00 0B 00 09 80 01 01 80 00 00 FF F1 FF FF 01 48
00 0C 00 0B 00 01 01 80 00 00 FF F1 FF FF 01 48
00 0D 00 0C 80 01 01 80 00 00 FF F1 FF FF 01 49
00 20 00 0E 00 01 00 8C 00 00 1F FF FF FF 00 8C
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

SW : 94 02

- Pourquoi les sorties de Read3F00-2 et Read3F00 sont différentes sachant que c'est le même ordre transparent qui est envoyé à la carte?
- A quoi correspond la valeur de SW obtenue ?
- Quelle est la valeur probable de l'APDU pour que SW revienne à 90 00 ?
- Le droit d'accès Create n'étant pas utilisé, mais Exec peut être utilisé, quels sont les identités des fichiers correspondant à des clés Rsa ?
- **Bonus :** Sachant que les clés Rsa sont sauvegardées sous la forme des théorèmes des restes chinois, Sp, Sq, p2,q2 et u, chaque composante étant précédée par un octet indiquant la taille déduire la taille probable des clés RSA utilisées. (calculer en fonction de la taille de N, la taille de chacune des composantes).
- **Bonus :** Pourquoi il y a une légère différence entre les tailles occupées par les fichiers clés RSA sur la carte ?
- Etebac5 n'utilise pas de certificats mais des accréditations contenant les clés publiques et des identifiants de taille fixe. La taille de ces accréditations est fixe de 298 octets. Quels sont les fichiers accréditations ?
- Par quoi la lecture de ces accréditations est protégée ?
- A partir de ce qui précède définir le mapping de la carte pour les clés Rsa et accréditations incluant uniquement les identifiants et droits d'accès. (pas de taille, pas de type des clés).

g) Quelle est l'identité et la taille du fichier en écriture possible de la carte ? par quoi cette écriture est protégée ?