M2 Cyber security - Smart Card Security - 2019
Evaluations de la sécurité d'une carte bancaire
Bank card security evaluation

1. Introduction [2/20]

Votre but est d'évaluer la sécurité d'une carte bancaire à puce qu'un développeur vous a fourni. *The goal is to evaluate the security of a bank card that a developer has given you.*

1.1. Citer des schémas de certifications / organismes de certification que le dévelopeur voudrait obtenir. *Cite certification schemes that a developer would like to get.*

1.2. Décrivez succintement (avec un schéma par exemple) comment se déroule une évauation sécuritaire entre le développeur, la laboratoire d'évaluation et l'organisme de certification *Quickly describe (with a diagram for instance) how a security evaluation takes place between the developer, the certification body and your evaluation lab.*

2. Attaque sur la validation du PIN / Verify PIN [9/20]

Le développeur a intégré une authentification de l'utilisateur à base d'un code PIN à 4 chiffres. Le but de votre attaque est de s'authentifier sans connaître le vrai code PIN (i.e. le retour de la fonction `Verify_PIN` doit être `true`). Le nombre d'essai est limité à 100, et contrôlé par la variable `tries` qui est stockée en mémoire non volatile. Lorsque cette variable passe en-dessous de 0, la fonction `kill_card()` tue la carte. Dans le cadre de l'évaluation, vous disposez du code C utilisé pour cette fonction. Il est écrit ci-dessous:

The developer integrated a user authentication with a 4-digit PIN code. The goal of the attack is to authenticate yourself without knowing the real PIN code (i.e. function `Verify_PIN` shall return `true`). Number of tries is limited to 100 and controlled by variable `tries` stored in a Non Volatile Memory. When this variable is below 0, the function `kill_card()` destroys the smartcard. For this evaluation, the developer gave you the C code used in this function. It is reproduced below:

```
1  const int true = 1;
2  const int false = 0;
3  extern void kill_card(void);
4  int tries = 100; // permanently stored & updated in a Non Volatile Memory
5  int correct_PIN[4] = {9,8,7,6};
6
7  int verifyPIN_0(int user_PIN[]){
8    if (tries < 0) kill_card();
9    for (int i=0; i < 4; i++) {
10     if(user_PIN[i] != correct_PIN[i]) {
11       tries --;
12       return false; } }
13   return true;}
```

2.1. Proposer une attaque pour trouver le PIN correct sans tuer aucune carte. *Propose an attack to retrieve the correct PIN without killing a single chip.*

2.2. En vous aidant de la table fournie à la fin de l'énoncé, proposez une quotation argumentée d'une telle attaque. *With the help for the table at the last page of this test, propose an argued rating for this attack.*

Suite à l'attaque précédente, le développeur décide d'améliorer le code. Sa nouvelle version est la suivante: *After the previous successful attack, the developer improved its code. Its new version is below:*

```
1  const int true = 0x1234;
2  const int false = 0x9876;
3  extern void kill_card(void);
4  int tries = 100; // permanently stored & updated in a Non Volatile Memory
5  int correct_PIN[4] = {9,8,7,6};
6
7  int verifyPIN_1(int user_PIN[]){
8    int correct_digits = 0;
9    if (tries < 0) kill_card();
10   tries--;
11   for (int i=0; i < 4; i++) {
12     if(user_PIN[i] != correct_PIN[i])
13       correct_digits --;
14     else
15       correct_digits ++;}
16   if (correct_digits == 4){
17     tries ++;
18     return true;}
19   else
20     return false;}
```

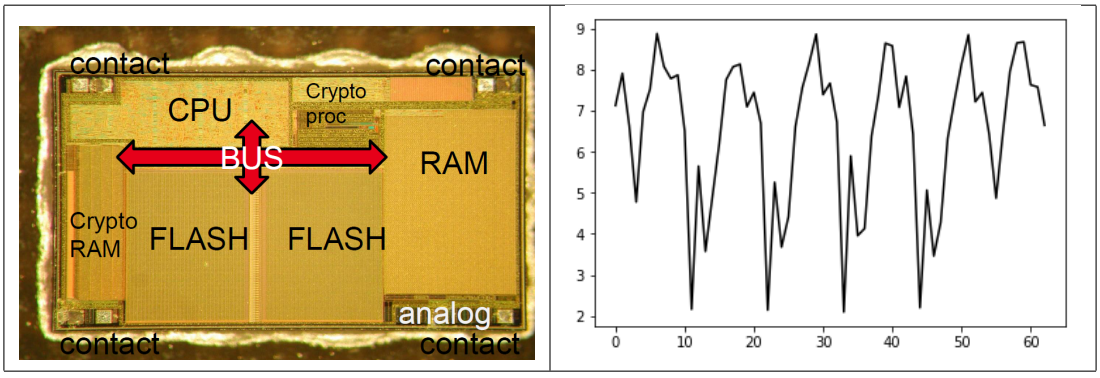
Expliquer certaines contremesures aux lignes indiquées ci-dessous. Pourquoi ont-elles été insérées dans le code et contre quels type d'attaque ont-elles été placées: *Explain the countermeasures at the following lines. Why and against which kind of attacks have they been inserted ?*

2.3. lines 1-2

2.4. lines 12 to 15

2.5. lines 9 and 16

2.6. La figure suivante montre une image du composant et une acquisition des émissions électromagnétiques du composant pendant que le code précédent se déroule. Proposer une attaque simple en faute sur ce code qui permette d'être authentifié sans connaître le code PIN. Indiquer où et quand réaliser l'attaque sur ce composant. *The figure below shows a chip picture as well as electromagnetic leakages recorded when the preceding code is run on the chip. Propose a single fault attack that allows to be authenticated without knowning the PIN code. Indicate where and when to attack the chip.*



2.7. Coter cette attaque en utilisant la table fournie en annexe à la fin de l'énoncé. *Rate this attack*

3. Cryptographie asymétrique Asymmetric cryptography (4/20)

L'authentification mutuelle de la carte est effectuée avec un chiffrement RSA. Le composant calcule une exponentiation modulaire $C = M^d$ où d est la clé privée que vous voulez retrouver. Vous connaissez l'algorithme utilisé qui est décrit ci-dessous.

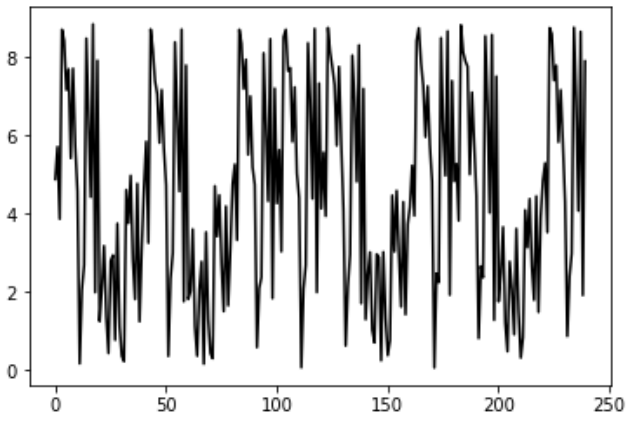
Mutual authentication is done with an RSA ciphering. The chip performs a modular exponentiation $C = M^d$ *where* d *is the private key you want to retrieve. The algorithm used is described below:*

Algorithm 1 Calculate $C = M \times d \mod n$ with $d = [d_{n-1}..d_0]$ d_0 being the Least Significant Bit. ex: d=0d13=0b1101 = [1101]

```
C ← 1, R ← M
for i = 0 to n − 1 do
    if d_i = 1 then
        C ← C × R mod n
    end if
    R ← R^2 mod n
end for
return Q
```

3.1. Dérouler chaque boucle de l'algorithme et vérifier qu'il fonctionne pour calculer C^{11} (11 en base décimale). *Describe every loops of this algorithm for a given key $d = 11$ in decimal.*

3.2. Retrouver un octet de la clé privée en s'aidant de la courbe de consommation de la figure ci-dessous. *Retrieve one byte of the private key with the help of the following consumption curve.*



4. Symetric cryptography [Barème: 5/20]

La puce utilise un chiffrement symétrique mais aucun document ne vous indique sa nature. *The chip has a symmetric cipher scheme but no document describes it.*

4.1. Comment feriez-vous pour retrouver le type d'algorithme utilisé ? Plusieurs réponses sont possibles, suivant la puissance de l'attaquant. *How would you find out which algorithm is used ? Several answers can be given, depending on the attacker's strength.*

Une partie de l'algorithme utilisé a finalement été retrouvé et est expliqué ci-dessous. L'idée est de retrouver la clé privée par une observation des canaux auxiliaires en ciblant la consommation du composant, le message pouvant être choisi par l'attaquant.

A part of the algorithm has eventually been retrieved and is explained below. The idea is to find the secret key with a side-channel attack targetting chip consumption, knowing the message can be chosen by the attacker.

4.2. Décrivez un banc dédié à ce type d'attaque.*Describe a test bench dedicated to side-channel attack.*

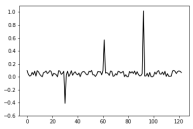
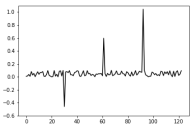
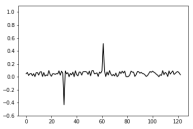
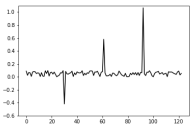
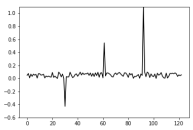
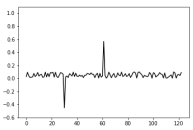
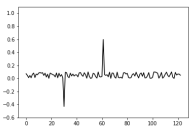
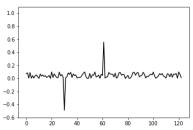
The algorithm is:

- messsage (m): 3 bits, key (k): 2 bits
- $C(m,k) =$ SBox(m xor k)
- SBox[] = [0x5, 0x3, 0x2, 0x7, 0x0, 0x1, 0x2, 0x4]

4.3. Compléter le tableau suivant avec la valeur manquante pour C(m,k). *Complete the following table with the missing value for C(m,k).*

k/m	000	001	010	011	100	101	110	111
00	101	011	010	111	000	001	?	100
01	011	101	111	010	001	000	100	010
10	010	111	101	011	010	100	000	001
11	111	010	011	101	100	010	001	000

1000 courbes ont été acquises et la moyenne de chaque courbe a été calculée pour chaque valeur possible du message (0b000, 0b001 . . . , 0b111). *Using the set of acquired curves for different messages (0b000, 0b001 . . . , 0b111) , the following means have been computed:*

message	000	001	010	011
Moyennes (Means)				
message	100	101	110	111
Moyennes (Means)				

4.4. Trouver sur les courbes quels sont les points (positions temporelles) où il pourrait y avoir des fuites ? *Find the time positions on the curve where the chip may leak ?*

4.5. Mener une DPA en ciblant le bit 1 (celui du milieu : 0bx**X**x?) *Attack the chip with a DPA targeting the bit 1 (the middle one : 0bx**X**x?)*

Factors	Identification	Exploitation
Elapsed time		
< one hour	0	0
< one day	1	3
< one week	2	4
< one month	3	6
> one month	5	8
Not practical	*	*
Expertise		
Layman	0	0
Proficient	2	2
Expert	5	4
Multiple Expert	7	6
Knowledge of the TOE		
Public	0	0
Restricted	2	2
Sensitive	4	3
Critical	6	5
Very critical hardware design	9	na
Access to TOE		
< 10 samples	0	0
< 100 samples	2	4
> 100 samples	3	6
Not practical	*	*
Equipment		
None	0	0
Standard	1	2
Specialized (1)	3	4
Bespoke	5	6
Multiple Bespoke	7	8