

Royaume du Maroc
UNIVERSITÉ MOHAMED V - RABAT

ECOLE NATIONALE SUPÉRIEURE D'INFORMATIQUE
ET D'ANALISE DES SYSTÈMES



Mini-projet cryptographie

L'attaque Simple Power Analysis sur RSA

Filière : Sécurité des systèmes d'information (SSI)

Réalisé par :

AGOULZI Imane
JOUIJATE Rim

Professeur :

BELKASMI Mostafa

Encadrant :

EL GAABOURI Ismail

Année universitaire : 2022 - 2023

Table des matières

Introduction générale	1
1 Contexte générale du projet	2
1.1 Introduction	2
1.2 L'algorithme RSA	2
1.3 Exponentiation modulaire	3
1.4 Simple Power Analysis	3
1.5 Conclusion	4
2 Démonstration	5
2.1 Introduction	5
2.2 Présentation de l'idée	5
2.3 La mise en oeuvre de l'idée	5
2.4 Résultats et discussion	6
2.5 Conclusion	9
Conclusion	10

Introduction générale

La cryptographie est une science qui étudie les méthodes de communication sécurisées en présence d'adversaires qui cherchent à intercepter ou altérer les informations échangées. Elle repose sur des algorithmes mathématiques permettant de chiffrer les messages transmis et de déchiffrer ces messages grâce à une clé secrète. La cryptographie assure la confidentialité, l'authenticité et l'intégrité des données, et est utilisée dans de nombreux domaines, tels que la sécurité informatique, la communication sans fil, la banque en ligne, le commerce électronique, etc.

La cryptographie symétrique (dite à clé secrète) et la cryptographie asymétrique (dite à clé publique) sont deux types de cryptographie, la première permet de chiffrer et de déchiffrer les messages à l'aide d'une même clé secrète partagée entre les deux parties communicantes, par ailleurs dans la deuxième, chaque partie communicante a une clé publique connue par tout le monde et une clé privée secrète, une pour chiffrer et l'autre pour déchiffrer.

RSA (Rivest-Shamir-Adleman), qui fait partie de la cryptographie asymétrique, est l'un des algorithmes de cryptographie les plus populaires utilisés pour sécuriser les communications électroniques. Il est basé sur la difficulté de factoriser des nombres entiers très grands en leurs facteurs premiers, ce qui rend la recherche de la clé privée très difficile pour un adversaire.

Cependant, même les algorithmes de cryptographie les plus robustes peuvent être vulnérables à des attaques si leur implémentation n'est pas sécurisée. L'attaque SPA sur RSA est l'une de ces attaques qui vise à exploiter les fuites de puissance dans les implantations matérielles de RSA pour récupérer la clé privée. Cette attaque illustre l'importance de la sécurité de l'implémentation dans la cryptographie et met en évidence la nécessité de mesures de protection physique et logique pour sécuriser les dispositifs de cryptographie.

Dans ce document, nous allons découvrir en premier lieu le principe de RSA, puis comment l'attaque SPA exploite l'exponentiation modulaire pour casser le RSA. En deuxième lieu, nous allons présenter une simulation de cette attaque sur un dispositif physique.

Chapitre 1

Contexte générale du projet

1.1 Introduction

Ce chapitre est dédié à l'étude de trois concepts fondamentaux en cryptographie : RSA, l'exponentiation modulaire et SPA. RSA est un algorithme de chiffrement asymétrique largement utilisé pour sécuriser les échanges de données sur Internet. L'exponentiation modulaire est une opération mathématique clé utilisée dans RSA, mais qui peut être vulnérable à des attaques du type SPA.

1.2 L'algorithme RSA

L'algorithme RSA (Rivest-Shamir-Adleman) est un algorithme de cryptographie asymétrique inventé en 1978 par R. Rivest, A. Shamir et L. Adleman. Cet algorithme est utilisé pour chiffrer et déchiffrer les données.

C'est la première instance connue des méthodes de cryptographie à clé publique, en effet, il utilise deux clés, une clé publique pour le chiffrement et une clé privée pour le déchiffrement.

L'algorithme fonction comme suit :

- La sélection de deux grands nombres premiers aléatoires p et q .
- Le calcul de modulo public $N = p * q$. La taille de N , au sens binaire, est noté n .
- On choisit l'exposant public e qui vérifie la condition $\text{pgcd}(e; L(N)) = 1$ avec $L(N)$ désigne l'indicatrice d'Euler du modulo N . Elle est calculée par $L(N) = (p - 1)(q - 1)$.

Pour chiffrer un message, on utilise **la clé publique (N, e)** du destinataire, qui va être utiliser dans une exponentiation modulaire avec l'exposant public e : $C = m^{**} e \bmod N$.

Le message chiffré ne peut être déchiffré que par la personne possédant **la clé privée d** correspondante qui est calculé comme suit : $d = e^{-1} \bmod L(N)$. L'opération de déchiffrement est réalisée de manière similaire en utilisant l'exposant privé d : $m = C^{**} d \bmod N$.

La sécurité théorique de l'algorithme RSA repose sur la difficulté de la factorisation de grands nombres entiers. Pour cela, on choisit des nombres de taille au moins égale à 2048 bits voire même à 4096 bits.

Afin de casser l'algorithme RSA, cela nécessite la factorisation du nombre n , avec les algorithmes classiques le temps que prend cette factorisation croît exponentiellement avec la longueur de la clé.

Finalement, l'algorithme RSA utilise l'exponentiation modulaire pour le chiffrement et le déchiffrement des messages, qui permet de réaliser des calculs avec des nombres très grands, et réduit le risque d'attaques en offrant une forte résistance aux attaques par force brute et aux attaques de factorisation.

1.3 Exponentiation modulaire

L'algorithme de l'exponentiation modulaire est une technique couramment utilisée en cryptographie pour calculer de grandes puissances des nombres avec une réduction modulaire à chaque étape. Cela permet de réduire le risque d'attaque par dépassement de capacité en limitant la taille des nombres intermédiaires à chaque étape. L'algorithme fonctionne en représentant l'exposant comme un nombre binaire, puis en multipliant la base par elle-même et en divisant l'exposant par deux jusqu'à ce que l'exposant devienne zéro. À chaque étape, si le bit le plus à droite de l'exposant est un, la valeur actuelle de la base est multipliée par le résultat.

Dans notre cas, on a utilisé l'algorithme suivant qui correspond à l'exponentiation modulaire binaire :

```
//calculate a^b mod n

Result = 1
//variable pour stocker le résultat final
temp = a
//la base de l'exponentation
While b is not 0
    If b is odd
        Result = Result * temp % n
    b = b/2
    temp = temp * temp % n

return result
```

FIGURE 1.1 – Le pseudo code de l'exponentiation modulaire binaire

L'exponentiation modulaire est généralement effectuée à l'aide de l'algorithme de Square-and-Multiply, qui utilise des opérations de carré et de multiplication pour réduire le nombre d'opérations nécessaires. L'algorithme commence par décomposer l'exposant binaire en une séquence de bits. Ensuite, les bits sont examinés un par un, de gauche à droite. Si le bit est un 0, on effectue une opération de carré, sinon on effectue une opération de multiplication suivie d'une opération de carré.

Les implémentations simples des algorithmes d'exponentiation binaire rendent le système cryptographique vulnérable aux attaques par canaux auxiliaires comme les attaques d'analyse de puissance simple (SPA).

1.4 Simple Power Analysis

Le concept de l'attaque SPA (Simple Power Analysis) a été introduit pour la première fois en 1998 par Paul Kocher. Cette attaque repose sur l'analyse des fuites de puissance de l'appareil lors de l'exécution d'une opération cryptographique, telle que l'exponentiation modulaire utilisée dans RSA.

On appelle attaque par simple analyse du courant (ou SPA), une attaque consistant à analyser la consommation de puissance d'un dispositif cryptographique (comme une carte à puce ou un microcontrôleur) lorsqu'il effectue des opérations cryptographiques.

Par exemple, lors d'une opération cryptographique, la consommation de puissance peut varier en fonction de la clé utilisée, des données d'entrée ou des valeurs intermédiaires générées pendant le calcul. Et donc la courbe de consommation est différente suivant les instructions exécutées et les données manipulées. En analysant cette courbe, un attaquant peut obtenir des informations sur la clé secrète ou d'autres informations sensibles.

L'attaquant peut extraire la clé s'il a un bon oscilloscope avec "zooms in". La figure ci-dessous montre l'allure d'un signal capturé lors d'une attaque SPA. Les oscillations représentent les opérations de calcul effectuées par un appareil pendant une opération cryptographique. D'où on peut tirer les bits de la clé secrète :

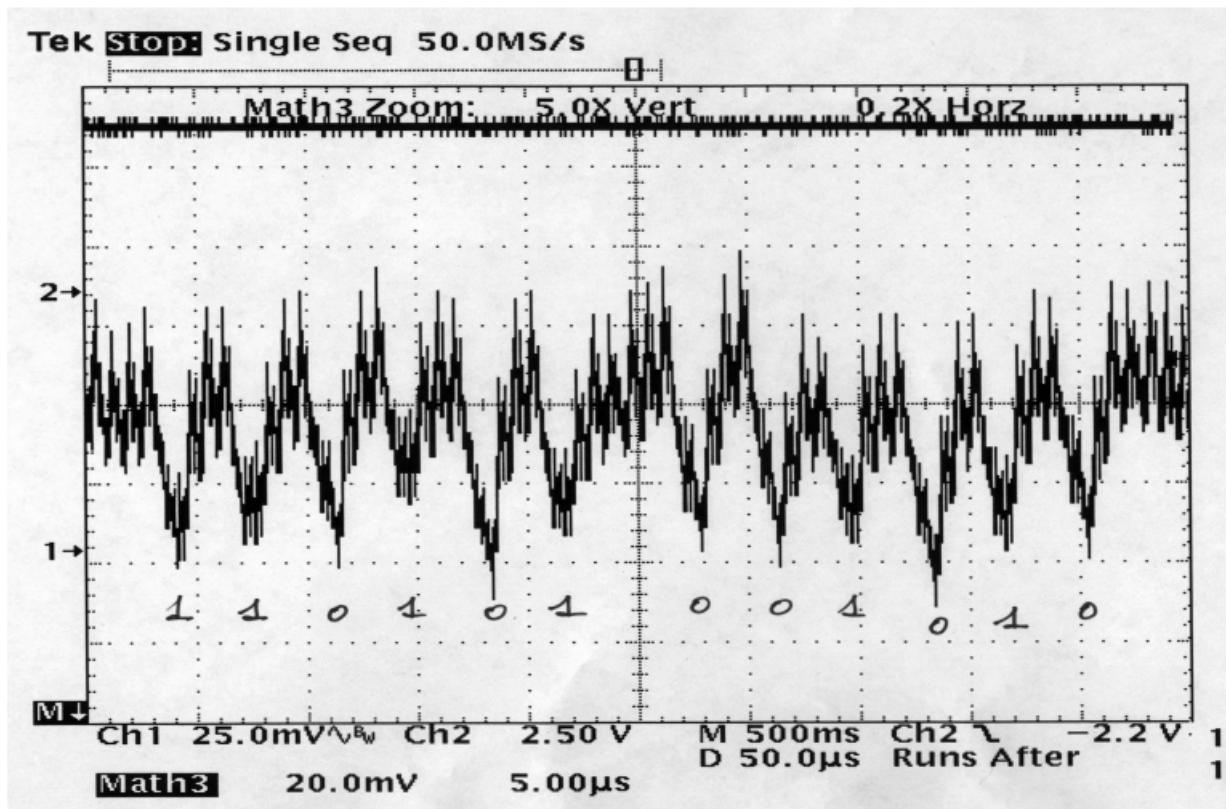


FIGURE 1.2 – Extraction des la clé secrète avec SPA - Exemple

1.5 Conclusion

En conclusion, RSA, l'exponentiation modulaire et SPA sont des concepts importants dans la cryptographie moderne. Toutefois, SPA peut représenter une menace pour la sécurité de RSA en révélant des informations sur la clé privée.

Chapitre 2

Démonstration

2.1 Introduction

Ce chapitre vise à exposer l'idée du projet que nous avons conçu ainsi que les étapes de sa réalisation. Ensuite, une section sera consacrée à l'explication et à la discussion des résultats que nous avons obtenus.

2.2 Présentation de l'idée

Le but principal de notre démonstration est de casser l'algorithme RSA à travers l'attaque SPA en exploitant la faiblesse de l'algorithme de l'exponentiation modulaire lors d'une communication chiffrée entre un émetteur et un récepteur.

Pour ce faire, nous avons utilisé deux cartes Arduino UNO qui représentent respectivement l'émetteur et le récepteur, et communiquent entre eux d'une manière sécurisée grâce à l'algorithme RSA.

Pour simuler l'attaque SPA, dont le but est de trouver la clé privée de récepteurs, nous avons essayé de visualiser les variations de la consommation d'énergie électrique au sein du récepteur à l'aide d'un oscilloscope.

2.3 La mise en oeuvre de l'idée

Le matériel utilisé dans cette démonstration est :

- 2 cartes Aduino UNO avec un microcontrôleur ATmega328.
- Un conducteur ohmique de résistance 100 ohm .
- Un oscilloscope.

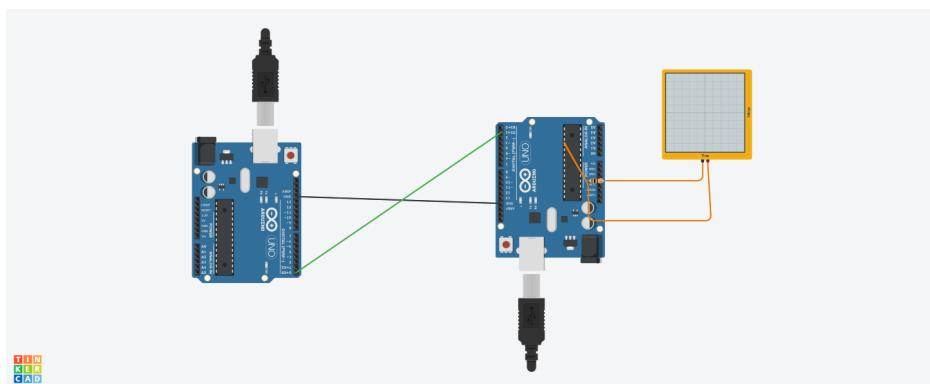


FIGURE 2.1 – Prototype

Nous avons commencé par associer le conducteur ohmique avec le port VCC du microcontrôleur.

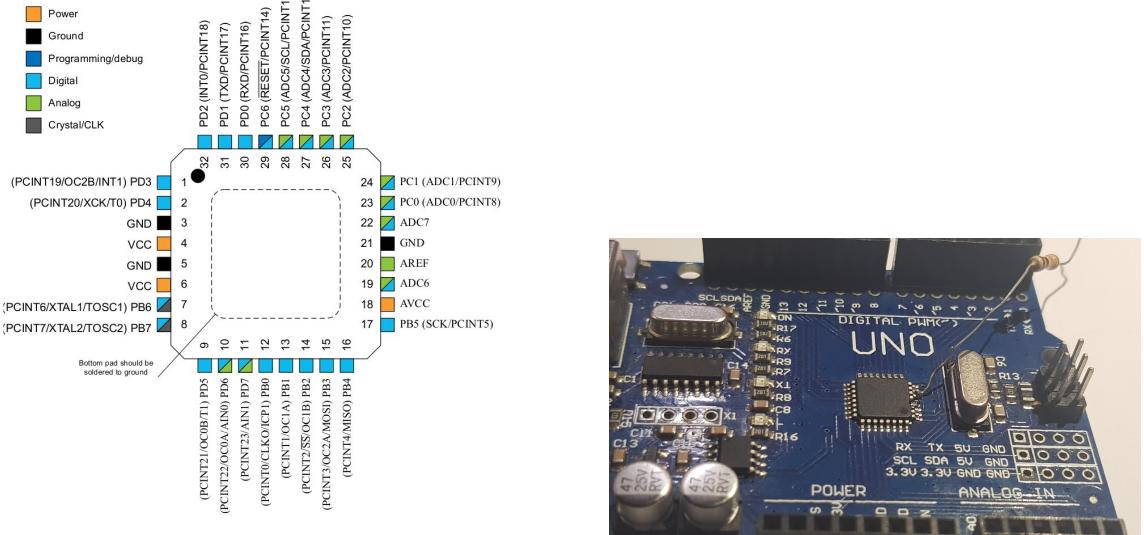


FIGURE 2.2 – État final de récepteur

Ensuite, nous avons mis les codes du programme au sein des deux cartes Arduino, ce qui va assurer l'échange des informations entre l'émetteur et le récepteur.

Les deux utilisent l'algorithme RSA pour le chiffrement et le déchiffrement des messages, les clés privées et publiques sont déjà choisies et affectées dans le code comme donné, pour réussir une communication chiffrée.

Puis, nous avons lié les cartes Arduino afin d'établir la connexion entre eux.

Et finalement, nous avons essayé de visualiser le signal entre les bornes du conducteur ohmique.

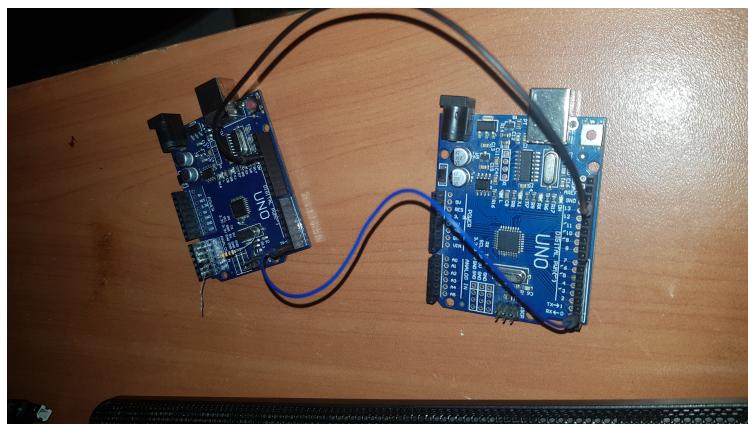


FIGURE 2.3 – Résultat de simulation

2.4 Résultats et discussion

Dans cette section, on va expliquer le graphe qu'on a eu lors de la simulation.

A travers l'oscilloscope, nous avons réussi à visualiser le signal électrique entre les bornes de la résistance. La figure ci-dessous représente l'allure du signal, mais il n'est pas assez bien précis à cause du modèle de l'oscilloscope utilisé.

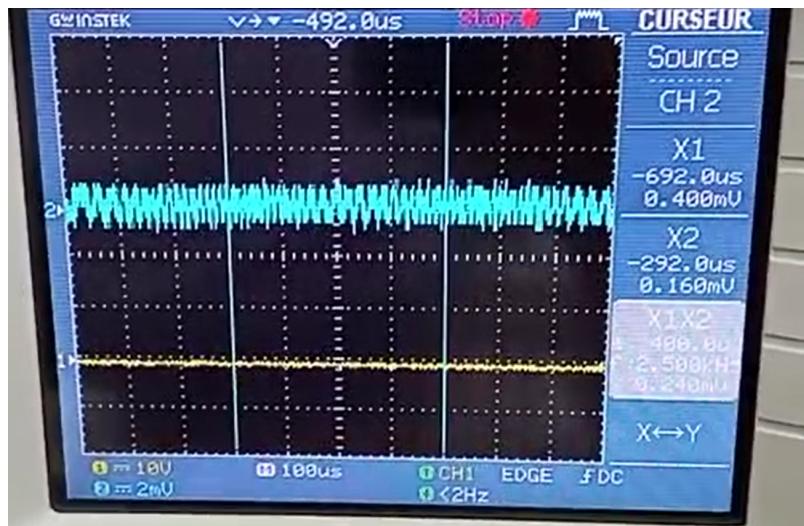


FIGURE 2.4 – signal visualisé par l'oscilloscope

NB : Vu les problèmes techniques mentionnés avant, nous nous avons basé sur un graphe d'une simulation déjà faite :

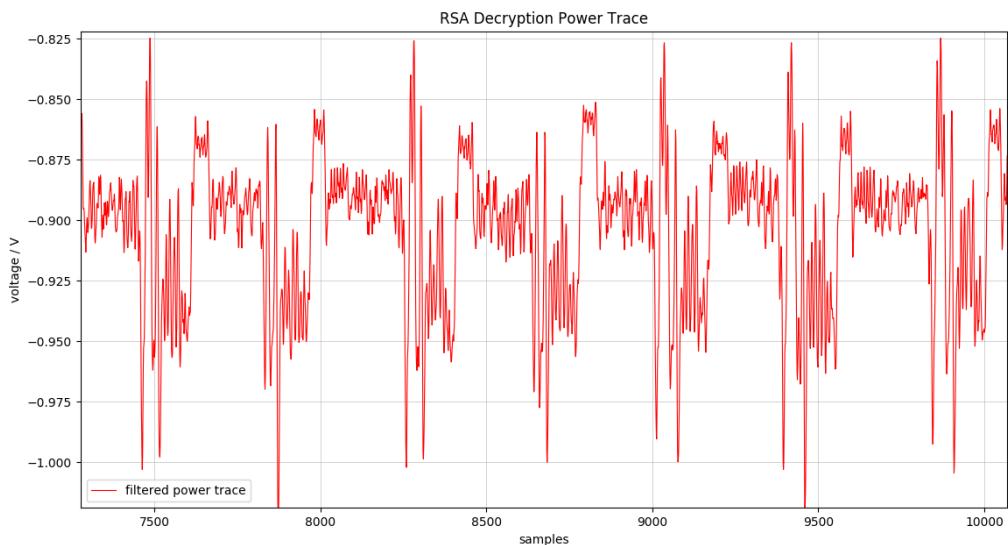


FIGURE 2.5 – Résultat de simulation

Pour interpréter ce graphe et extraire la clé privée du récepteur, et après avoir compris comment se passent les opérations du carré et de la multiplication, nous déduisons que chaque opération consomme un niveau d'énergie différent à toute autre opération de l'exponentiation modulaire. En fait, la multiplication consomme plus d'énergie et plus du temps contrairement aux carrés, ce qui permet de la connaître facilement.

Par ailleurs, pour la figure précédente, la multiplication est celle qui a les amplitudes des oscillations les plus grandes, et le carré est celui pour les amplitudes les plus petites.

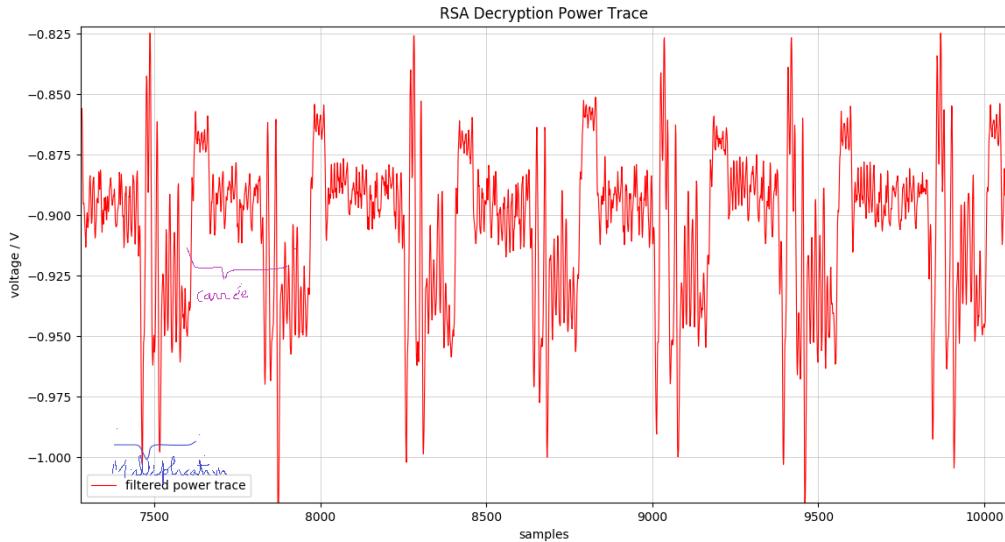


FIGURE 2.6 – La différence entre les deux opérations : multiplication et carré

Aussi, comme déjà mentionné dans l'algorithme de l'exponentiation modulaire, lorsqu'on a le bit 1 cela signifie qu'il y avait eu deux multiplication-carrées, et lorsqu'on a le bit 0, il y avait eu juste une multiplication-carré.

Maintenant, et que nous avons compris la méthode, nous pouvons extraire de la figure la clé privée du récepteur.

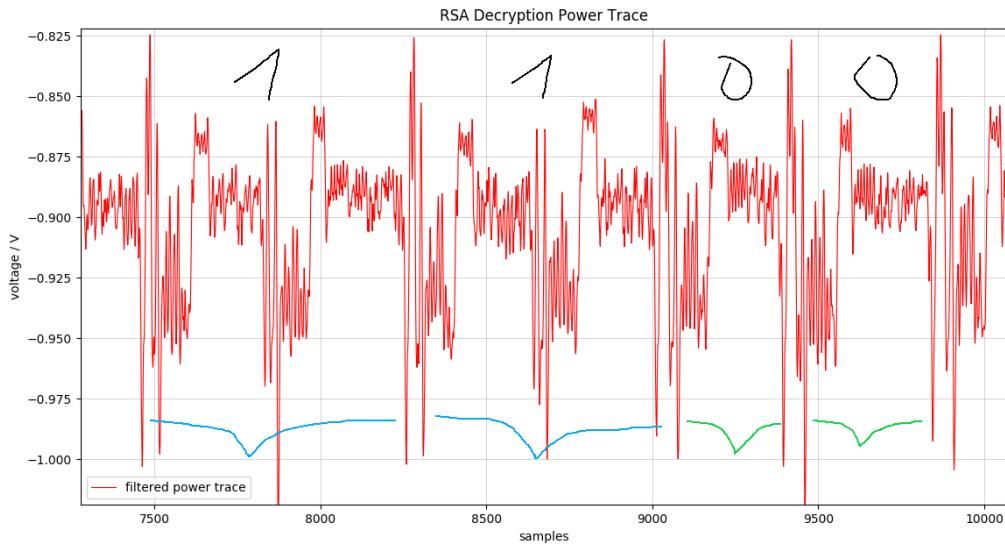


FIGURE 2.7 – Extraction de la clé privée du récepteur

Nous obtenons la valeur binaire **1100** qui correspond en décimal à **12**.

Remarque : Plus la clé est grande, plus son extraction de l'allure de la consommation électrique est difficile manuellement. Mais cela ne pose aucun problème aux attaquants, car avec un simple script - en Python par exemple - ils peuvent extraire les bits 1 et 0 correspondant aux opérations de multiplication-carrée.

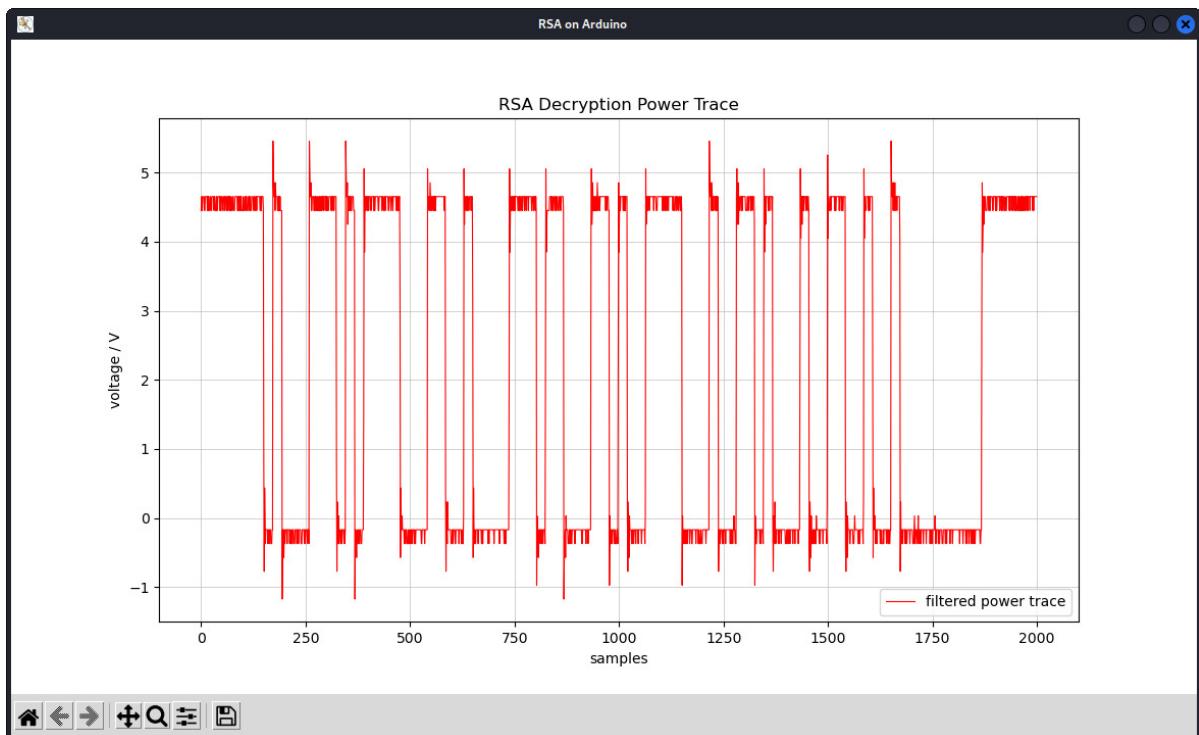


FIGURE 2.8 – La visualisation du signal d'une longue clé

De la figure ci-dessus, on peut extraire la clé à l'aide d'un script Python qui a donné la valeur binaire suivante 100110000101100001111001100011110000101000111101110001 correspondant à 10 720 368 893 529 969 en décimal.

2.5 Conclusion

Dans ce chapitre, nous avons réussi à faire la simulation de l'attaque SPA qui représente une menace potentielle pour la sécurité de l'algorithme RSA. Par conséquent, nous sommes arrivés à extraire la clé privée du récepteur utilisant l'algorithme RSA dans sa communication avec l'émetteur.

Conclusion

En conclusion, notre étude de l'attaque SPA sur les microcontrôleurs nous a montré à quel point cette méthode d'attaque est efficace contre les algorithmes de cryptographie tels que RSA. Grâce à ce travail, nous avons pu simuler cette attaque en mesurant la variation de la consommation d'énergie électrique dans la carte Arduino récepteur à l'aide d'un oscilloscope, et en exploitant la faiblesse de l'algorithme d'exponentiation modulaire pour extraire la clé privée du récepteur.

Cette expérience a mis en évidence la vulnérabilité des systèmes de cryptographie aux attaques SPA, et l'importance de prendre en compte la sécurité physique des dispositifs électroniques lors de la conception des systèmes cryptographiques. Il est donc essentiel de mettre en place des contre-mesures appropriées pour protéger ces systèmes contre les attaques SPA, telles que l'utilisation de techniques de masquage de données, de techniques de brouillage de l'horloge ou de l'utilisation de matériaux absorbants pour réduire les émissions électromagnétiques.

En somme, notre projet a démontré l'importance de la sécurité physique dans la conception des systèmes cryptographiques, et a souligné la nécessité de mettre en place des mesures de sécurité efficaces pour protéger les systèmes de cryptographie contre les attaques SPA.

Bibliographie

- [1] Nadia el mrabet, attaques par canaux cachés. <https://www.emse.fr/~nadia.el-mrabet/Presentation/Cours5_SCA.pdf>.
- [2] Power analysis. <https://en.wikipedia.org/wiki/Power_analysis>.
- [3] Power analysis attacks. <<https://www.youtube.com/watch?v=2iDLfuEBcs8>>.
- [4] Power analysis based side channel attack. <https://www.researchgate.net/publication/322243368_Power_Analysis_Based_Side_Channel_Attack>.
- [5] Rsa algorithm in cryptography. <<https://www.geeksforgeeks.org/rsa-algorithm-cryptography/>>.
- [6] Spa-based adaptive chosen-ciphertext attack on rsa implementation. <https://link.springer.com/content/pdf/10.1007/3-540-45664-3_18.pdf>.
- [7] Tamper-resistant cryptographic hardware. <https://www.researchgate.net/publication/312873398_Tamper-resistant_cryptographic_hardware>.
- [8] Thèse : Contrer l'attaque simple power analysis efficacement dans les applications de la cryptographie asymétrique, algorithmes et implantations. <<https://theses.hal.science/tel-01269753/document>>.
- [9] Thèse : Méthodologie et outils pour la mise en pratique des attaques par collision et attaques horizontales sur l'exponentiation modulaire. <<https://www.theses.fr/2017LYSEM010.pdf>>.
- [10] Wolfgang rankl and wolfgang effing ,smart card handbook, third edition. <>.
- [11] Yossi oren, attacks on secure implementations. <<https://www.youtube.com/watch?v=dTCsycglAsg>>.