

# Analyse de la consommation instantannée de courant sur RSA

Kévin Boyeldieu  
Titouan Tanguy  
TLS-SEC

Mars 2018

## Abstract

Le but de ce challenge est de montrer comment l'analyse de la consommation instantannée de courant d'un cryptosystème cryptographiquement sûr peut le compromettre si son implémentation n'est pas rigoureuse. Nous vous proposons de réaliser à travers une simulation web chaque étape de l'attaque (de la préparation du circuit électronique à l'analyse sur l'oscilloscope) afin de casser l'algorithme RSA implémenté sur un Arduino Nano. Parviendrez vous à retrouver la clé privée?

## Description du tutoriel

L'objectif de ce tutoriel est de réaliser chacune des étapes d'une attaque par consommation instantannée de courant afin de prendre conscience des dangers d'une implémentation non rigoureuse d'un algorithme mathématiquement sûr. Ces différentes étapes sont les suivantes :

- 1- réfléchir à un moyen de mesurer la consommation instantannée de courant du circuit;
- 2,3- préparer le circuit en y soudant et dessoudant certains composants;
- 4- analyser les résultats obtenus sur l'oscilloscope pour casser l'algorithme implémenté sur la carte.

Nous vous proposons dans ce tutoriel de casser l'algorithme RSA, implémenté sur une carte Arduino Nano. Les spécifications de cette carte sont fournies dans le fichier *arduino\_nano\_specs.png*. La documentation du micro contrôleur utilisé par l'Arduino Nano est également jointe, dans le pdf *ATmega328:P.pdf*. Pour vous faciliter la tâche, chaque étape devra être correcte et validée afin de passer à la suivante. Le tutoriel se termine lorsque vous parvenez à retrouver la clé privée utilisée par le cryptosystème.

Pour comprendre comment exploiter les résultats lors de la dernière étape de l'attaque, voici le code contenu dans la carte:

```
c : message chiffré
m : message déchiffré
d : clé privée
n : modulo

while true:
    // RSA : calcul de  $c^d \bmod n$ 
    r = 1
    pour chaque bit de d:
        r = r*r mod n
        if bit == 1:
            r = r*c mod n
    m = r
```

En outre, on admettra que le calcul d'une multiplication modulaire nécessite environ 100 cycles d'horloges.

## Solution du tutoriel

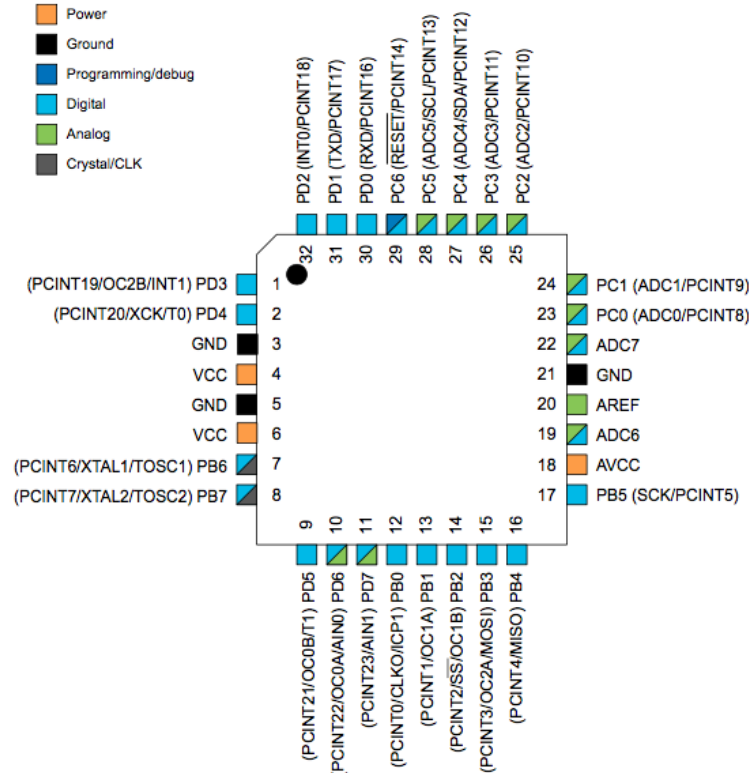
Comme expliqué dans la description du tutoriel, le but pour l'utilisateur est de retrouver la clé privée RSA utilisée par le cryptosystème.

Pour ce faire la marche à suivre est la suivante :

**1-** C'est la loi d'Ohm  $U = R * I$  qui va nous permettre de mesurer la consommation instantanée de courant du circuit. La loi d'Ohm énonce que la différence de potentiel ou tension aux bornes d'un résistor est proportionnelle à l'intensité du courant électrique qui la traverse. Ainsi, la tension mesurée par l'oscilloscope peut être directement reliée au courant. Il s'agit maintenant de placer une résistance au bon endroit sur notre circuit.

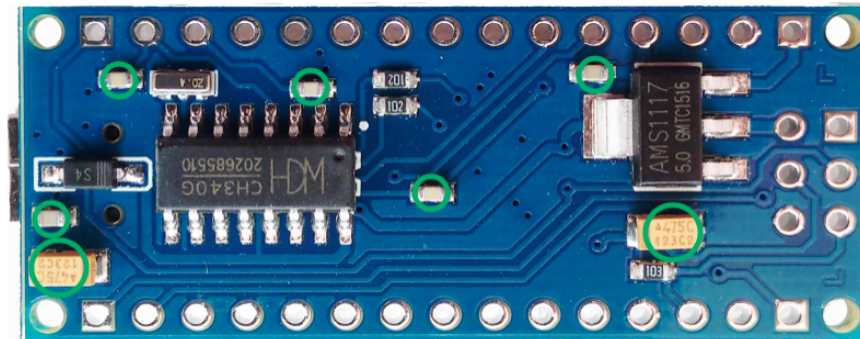
**2-** Pour réduire le bruit et obtenir des mesures significatives, il faut placer cette résistance sur la ligne d'alimentation au plus près du micro contrôleur. La documentation permet de retrouver les *pins* VCC (alimentation) de la puce. Il s'agit donc de dessouder l'un de ces *pins* (4 ou 6) et d'y insérer la résistance. Dans notre simulation, cela consiste à positionner les deux extrémités de la résistance soit sur le *pin* 4, soit sur le 6. Notons que le *pin* AVCC correspond à l'alimentation analogique et ne conviendrait pas pour nos mesures.

Figure 5-3. 32-pin TQFP Top View



**3-** Pour amplifier les différences de consommation instantanée de courant, il est

nécessaire d'enlever les condensateurs positionnés à l'arrière du circuit. Ces composants électroniques ont la propriété de pouvoir stocker des charges électriques et ainsi stabiliser une alimentation électrique (ils se déchargent lors des chutes de tension et se rechargent lors des pics de tension). Il est donc impératif de dessouder les 7 condensateurs (entourés en vert) présents à l'arrière de l'Arduino Nano.



4- Nous pouvons dès à présent passer à l'analyse des traces obtenues sur l'oscilloscope. Il s'agit dans un premier temps de régler les sensibilités verticales (V/div) et horizontales (s/div). La fréquence de l'horloge de l'Arduino Nano est de 16MHz. Pour pouvoir observer quelque chose, il faut donc choisir une sensibilité horizontale plus grande que 16MHz. Il est indiqué dans la description du challenge qu'une multiplication modulaire nécessite en moyenne 100 cycles d'horloge. On choisit donc une sensibilité horizontale d'environ  $100/16000000 \simeq 6e-6$ . Concernant la sensibilité verticale, la documentation indique que la tension d'alimentation de l'Arduino Nano est de 7-12V, donc 2-3V/div devrait convenir. Une fois ces réglages effectués, il faut se pencher sur l'implémentation de l'algorithme RSA mis sur la carte. On remarque qu'il n'est pas symétrique : le traitement effectué est différent en fonction du bit de clé traité. Si le bit courant vaut 1 alors deux opérations sont effectuées (une mise au carré, puis une multiplication). S'il vaut 0, alors une seule opération est effectuée (une mise au carré). D'où le nom de l'algorithme utilisé : *Square and Multiply*. Il s'agit de donc de repérer ces deux motifs (voir Figure 1) afin de reconstruire la clé privée. En analysant les quatre traces fournies, on trouve : 0b001010011000001100011000101100110000000010000110. Ce résultat doit être donné en hexadécimal, soit 0x298318b30086.

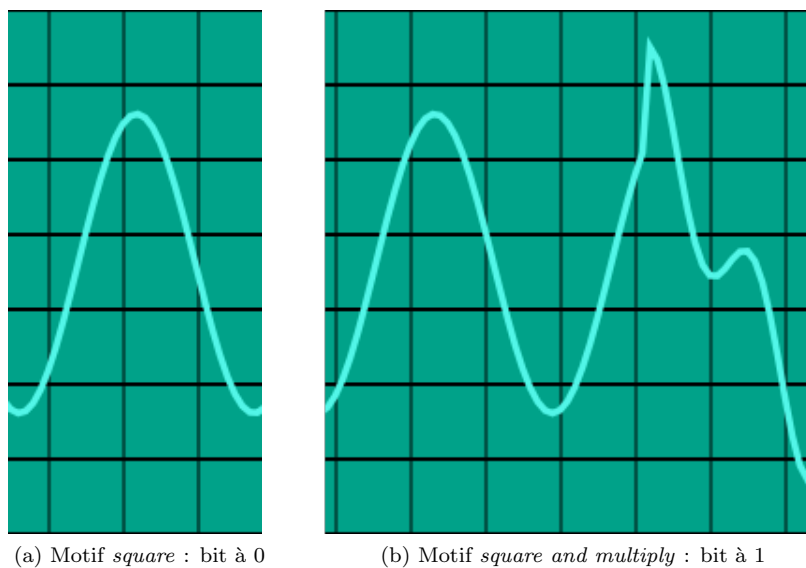


Figure 1: Motifs à identifier sur l'oscilloscope

## Procédure d'installation

Ce tutoriel nécessite une connexion internet (les scripts javascripts utilisés ne sont pas inclus dans les sources fournies) et un navigateur de version récente :

- Chrome version 63 ou 64;
- Firefox 57 ou 58 ( Firefox ESR n'est pas supporté)
- Internet Explorer 11 et Edge 40 ou 41
- Safari 10 ou 11 (Macintosh seulement)

Pour démarrer le tutoriel, il suffit d'ouvrir dans l'un de ces navigateurs web la page *page0.html*.