PB173 Domain specific development: side-channel analysis

Seminar 3: continuation of the previous seminar “SPA Exercise and Acquisition/ChipWhisperer”

Łukasz Chmielewski (chmiel@fi.muni.cz),
Consultation: in A406 on Fridays 9:30-11:00
(please email Łukasz before coming)

CROCS
Centre for Research on
Cryptography and Security

People

- Łukasz Chmielewski (CRoCS@FI MU)
  - Office hours (consultation): Friday 9:30-11:00, A406
  - Contact me first ↓

  ✉ chmiel@fi.muni.cz

  https://keybase.io/grasshoppper

  @chmiel:fi.muni.cz
- Milan Šorf (CRoCS@FI MU)
  - Office hours (consultation): Friday 16:00-18:00, A403
  - Contact me first ↓

  ✉ xsorf@fi.muni.cz,

  https://keybase.io/milan_s

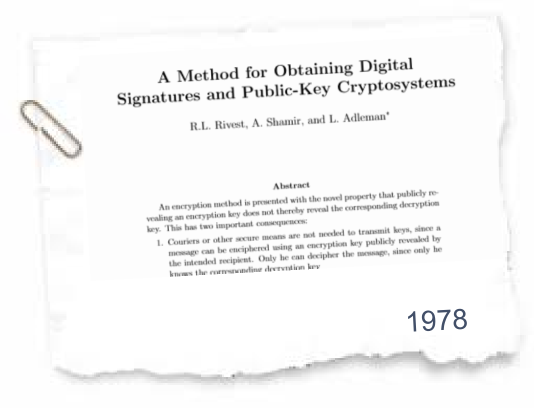
  @xsorf:fi.muni.cz

Solving Exercise: SPA on RSA

RSA

- Two primes p and q

- $N = pq$
- $\varphi(N) = (p-1)(q-1)$
- $e = 3, 5, 7, 17, 257, 65537 \rightarrow \gcd\big(e, \varphi(N)\big) = 1$
- $d = e^{-1} \bmod \varphi(N)$



A Method for Obtaining Digital Signatures and Public-Key Cryptosystems

R.L. Rivest, A. Shamir, and L. Adleman*

Abstract

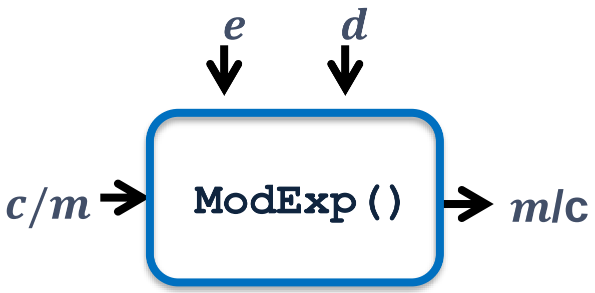
An encryption method is presented with the novel property that publicly revealing an encryption key does not thereby reveal the corresponding decryption key. This has two important consequences:

1. Couriers or other secure means are not needed to transmit keys, since a message can be enciphered using an encryption key publicly revealed by the intended recipient. Only he can decipher the message, since only he knows the corresponding decryption key

1978

Modular Exponentiation:

- Encryption / Verification: $c = m^e \bmod N$
- Decryption / Signature: $m = c^d \bmod N$



RSA Exponentiation (1)

```
ModExp(c) {
        $A = 1$
        for ( $i = n-1$; $i \geq 0$; $i--$)
              $A = A^2 \bmod N$
              if ($d_i == 1$)
                    $A = A*c \bmod N$
              end if
        end for
        return $A = c^d \bmod N$
}
```

$d=(101)=5$

$A = 1$,

$d_2=1$

$A = A^2 \bmod N=1$

$A = A*c \bmod N=c$

$d_1=0$

$A = A^2 \bmod N=c^2$

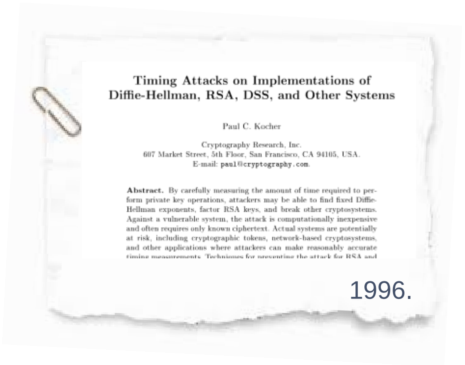
$d_0=1$

$A = A^2 \bmod N=c^4$

$A = A*c \bmod N=c^5$

Simple Power Analysis on RSA

```
ModExp(c) {
    A = 1
    for ( i = n-1; i≥0; i− −)
        A = A^2 mod N                S
        if (d_i = =1)
            A = A*c mod N            M
        end if
    end for
    return A = c^d mod N
}
```



Timing Attacks on Implementations of
Diffie-Hellman, RSA, DSS, and Other Systems

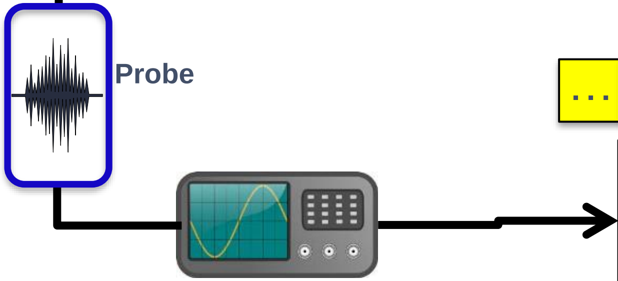
Paul C. Kocher

Cryptography Research, Inc.
607 Market Street, 5th Floor, San Francisco, CA 94105, USA.
E-mail: paul@cryptography.com.

Abstract. By carefully measuring the amount of time required to perform private key operations, attackers may be able to find fixed Diffie-Hellman exponents, factor RSA keys, and break other cryptosystems. Against a vulnerable system, the attack is computationally inexpensive and often requires only known ciphertext. Actual systems are potentially at risk, including cryptographic tokens, network-based cryptosystems, and other applications where attackers can make reasonably accurate timing measurements. Techniques for preventing the attack for RSA and

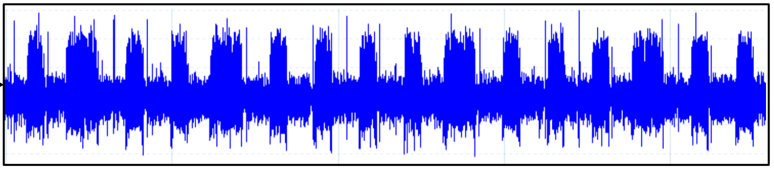
1996.

"By carefully measuring the *amount of time* required to perform private key operations, attackers may be able to find […] RSA keys."

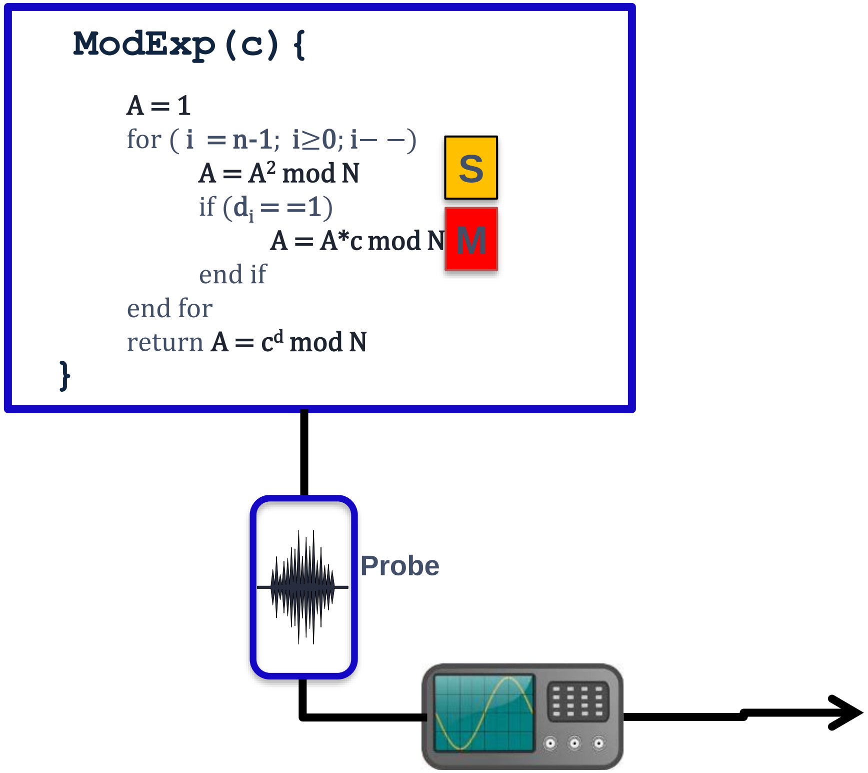


1	0		1	0	0	0		1	0	0		1	0

… S M S S M S S S S M S S S M S S …



Simple Power Analysis on RSA



```
ModExp(c) {
    $A = 1$
    for ( $i = n-1;\ i \geq 0;\ i- -$ )
        $A = A^2 \bmod N$
        if ($d_i = = 1$)
            $A = A*c \bmod N$
        end if
    end for
    return $A = c^d \bmod N$
}
```

This SPA matching does not always need to look this way!
One pattern might correspond multiple operations etc.

RSA Exponentiation (2)

```
ModExp(c) {
        $A = c$
        $j=-1$
        for ( $i = n-1$; $i \geq 0$; $i--$)
              if ($d_i$ ==1):
                    $j = i$
                    break
              end if
        if $j==-1$:
              return 1
        end if
        …
        …
        for ( $i = j-1$; $i \geq 0$; $i--$)
              $A = A^2 \bmod N$
              if ($d_i$ = =1):
                    $A = A*c \bmod N$
              end if
        end for
        return $A = c^d \bmod N$
}
```

$d=(0101)=5$

$j\text{-}1 = 1$

$A = c$

$d_1=0$

$A = A^2 \bmod N=c^2$

$d_0=1$

$A = A^2 \bmod N=c^4$

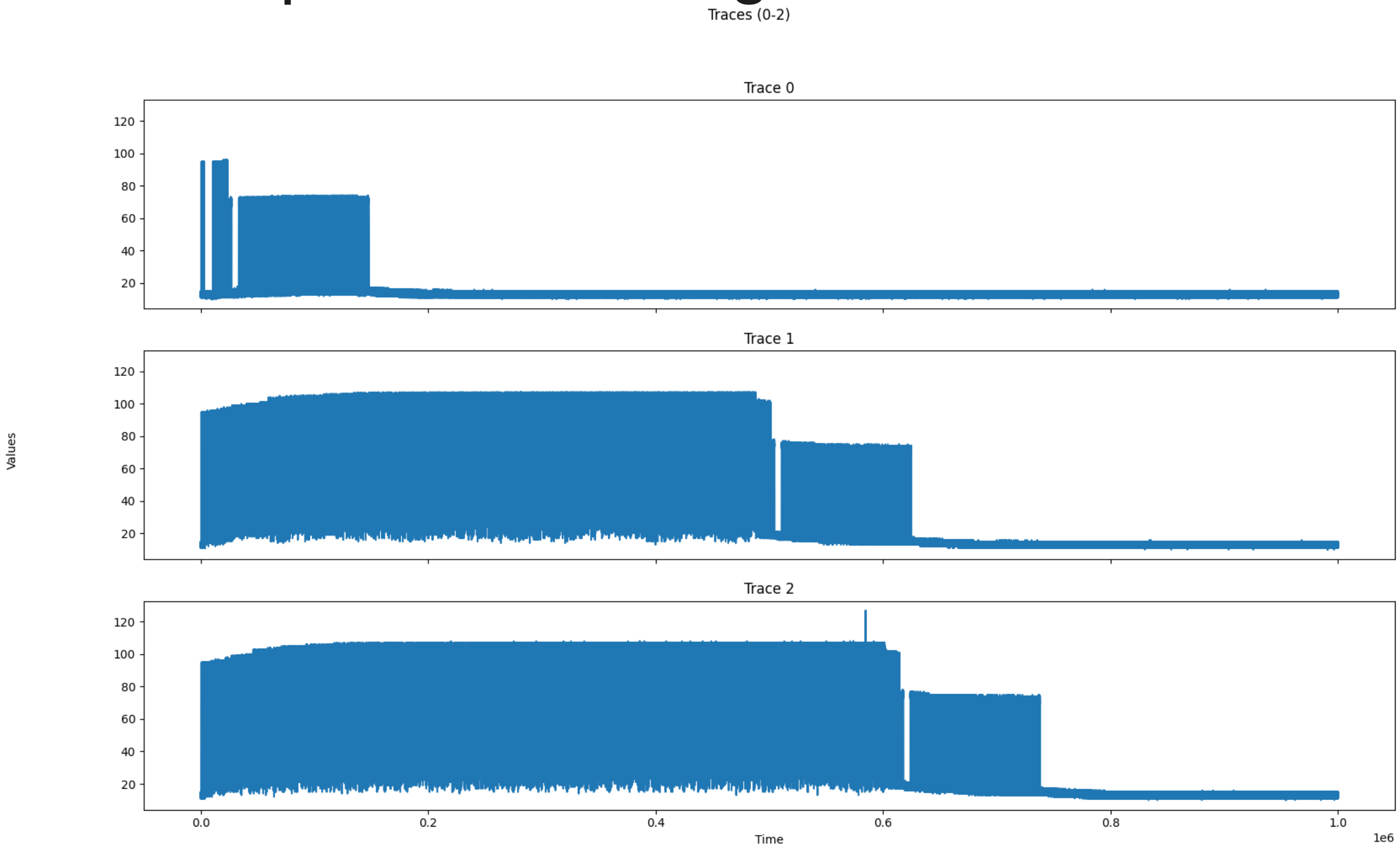
$A = A*c \bmod N=c^5$

Excercise

- RSA_unprotected.trs
- visualize.py
  - python3
  - Install matplotlib (e.g., pip)
  - Install trsfile (available on pip)
  - Feel free to modify the code and ask me questions about that.
- Three different traces
  - Tell me first 20 most significant bits of each exponent.
- Take your time, good luck!
  - I will give some hints during the exercise ☺

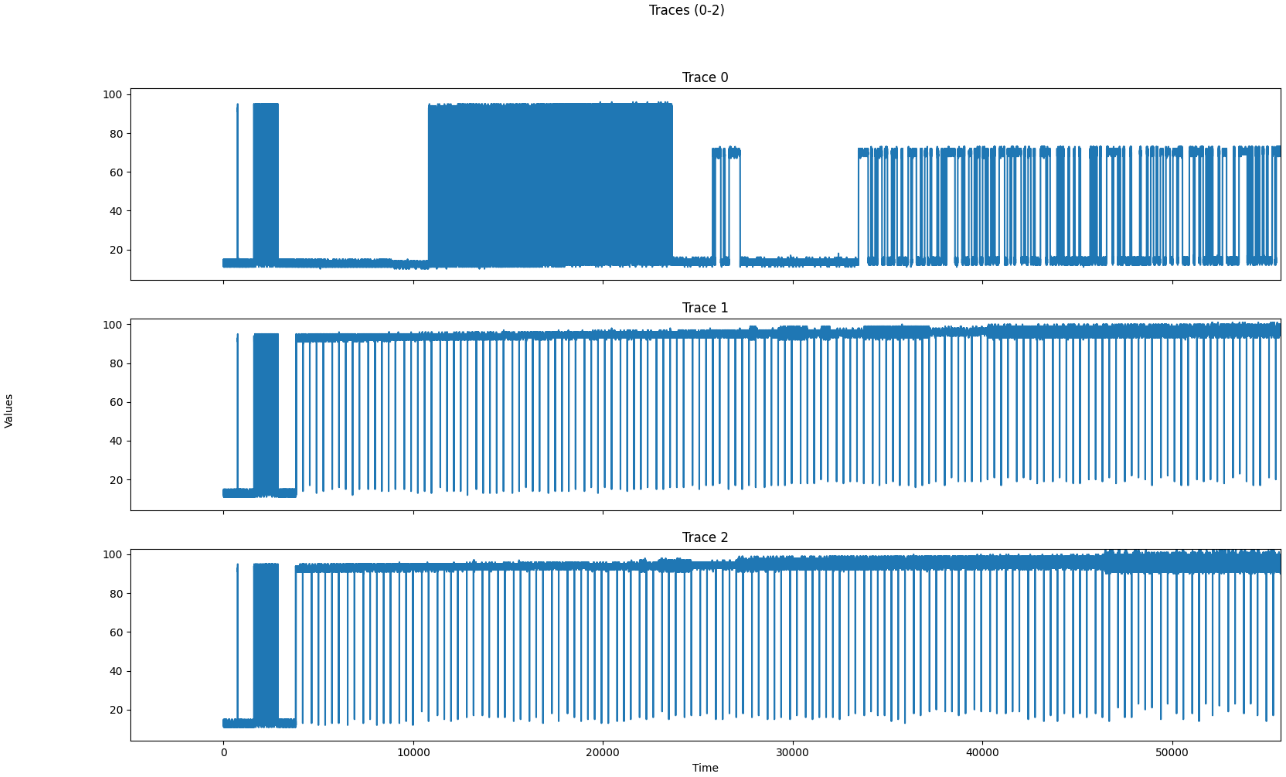
Exercise

- SPA with operation leakage



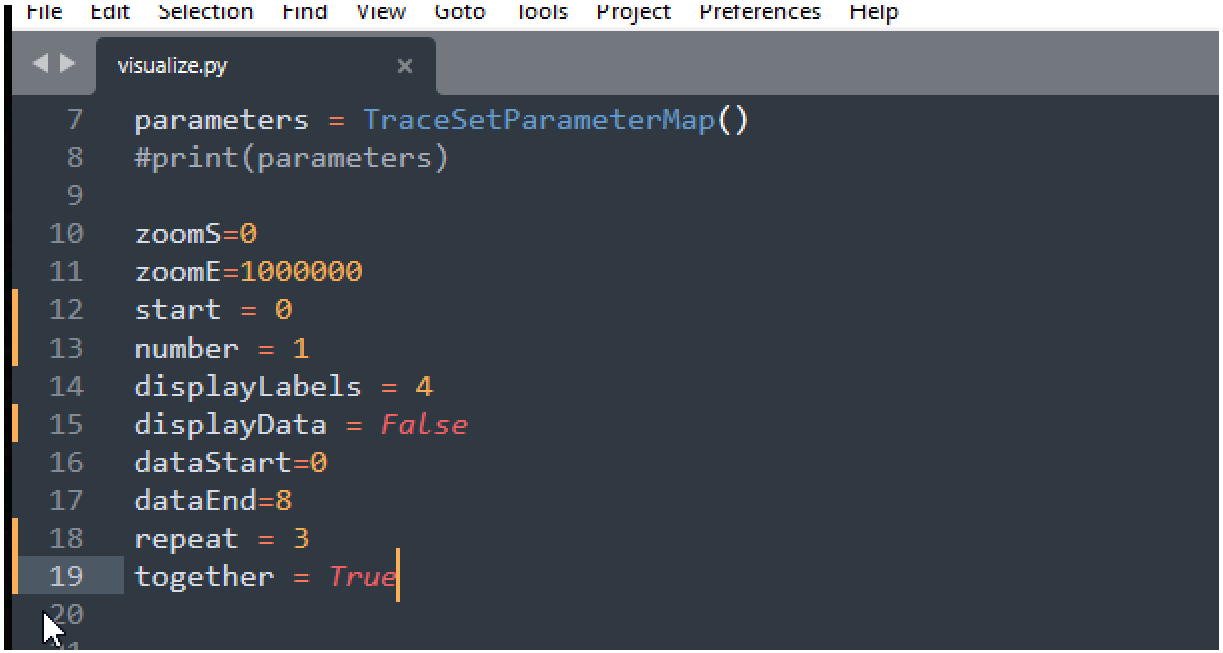
Exercise

- Try to zoom in and find the RSA exponentiation and then get the exponent!



Exercise

- How the visualization script works?



```
parameters = TraceSetParameterMap()
#print(parameters)

zoomS=0
zoomE=1000000
start = 0
number = 1
displayLabels = 4
displayData = False
dataStart=0
dataEnd=8
repeat = 3
together = True
```

Solution?

- Anyone?
- First trace?
- Second trace?
- Third trace?

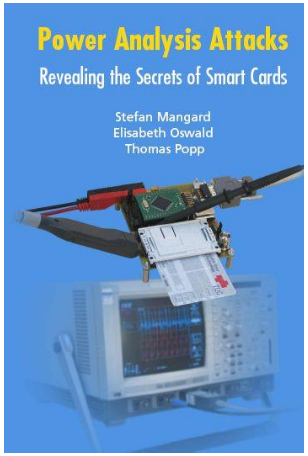
Finishing Acquisition / Chip Whisperer presentation

Homework

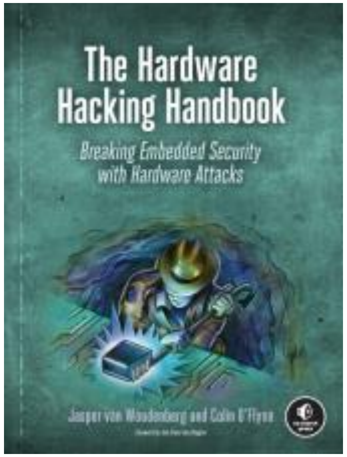
- TODOs before the next seminar:
  - Try to capture side-channel traces
  - Compute input / output correlation
  - Try differential power analysis
- On the next seminar – division of topics
- And in parallel we will also work on Chip Whisperer.

Reading

- For interested people
- Side-Channel Analysis – blue book:
  - http://dpabook.iaik.tugraz.at/
  - The books is available at the uni.
  - Look online



- The Hardware Hacking Handbook:
  - https://nostarch.com/hardwarehacking
  - I have an epub version.


The Hardware Hacking Handbook
Breaking Embedded Security with Hardware Attacks
Jasper van Woudenberg and Colin O'Flynn

Questions