IDEMIA

FA-LLLING FOR RSA

Lattice-based Fault Attacks against RSA Encryption and Signature

Guillaume Barbu
IDEMIA Cryptography & Security Lab

September 13, 2022

OUTLINE

INTRODUCTION

1

FAULT ATTACKS & FAULT MODELS

Fault Attacks have long proven their devastating effects on unprotected crypto implementations

To simulate the effect of a fault as well as to design and assess countermeasures, we use fault models such as ***stuck-at-0***, ***stuck-at-1***, ***bitflip*** or ***random-byte***

Through these fault models, we implicitly assume that a fault will only alter a bit, a byte or at most a word of a value that is being manipulated

This can be somehow limiting and possibly lead to ignore some real-world effects

FAULT ATTACKS + LATTICE REDUCTION = FA-LLL

At CT-RSA 2022, Cao *et al.* proposed several attacks using the following strategy:

- Fault several executions, assuming **bounded random additive errors** :

$$\tilde{x}_i = x_i + \varepsilon_i, \varepsilon_i < \mathcal{B}$$

- Use the faulty outputs to exhibit a Hidden Number Problem instance and solve it thanks to lattice reduction algorithms to recover the secret value

They applied it to several deterministic elliptic curve signature schemes

Interestingly, this allows using larger, unknown (random, or not) faults

HIDDEN NUMBER PROBLEM AND LATTICE REDUCTION

Hidden Number Problem

Given the l MSBs (y_i) of random multiples (t_i) of a secret α modulo a prime p, find α

$$0 \leq t_i \cdot \alpha \bmod p - y_i \leq p/2^l$$

It is possible to solve this HNP by searching for short vectors in the lattice spawn by the following matrix

$$\mathbf{M} = \begin{pmatrix} p & 0 & \dots & 0 & 0 \\ 0 & \ddots & & \vdots & \vdots \\ \vdots & & p & 0 & \vdots \\ t_1 & \dots & t_\delta & 2^{-l} & 0 \\ -y_1 & \dots & -y_\delta & 0 & 1 \end{pmatrix}$$

HIDDEN NUMBER PROBLEM AND LATTICE REDUCTION

Indeed from the coordinate vector $\mathbf{x} = (h_1, \ldots, h_\delta, \alpha, 1)$ and recalling

$$\mathbf{M} = \begin{pmatrix} p & 0 & \ldots & 0 & 0 \\ 0 & \ddots & & \vdots & \vdots \\ \vdots & & p & 0 & \vdots \\ t_1 & \ldots & t_\delta & 2^{-l} & 0 \\ -y_1 & \ldots & -y_\delta & 0 & 1 \end{pmatrix}$$

We have that $\mathbf{v} = \mathbf{xM} = (h_1 p + t_1 \alpha - y_1, \ldots, h_\delta n + t_\delta \alpha - y_\delta, 2^{-l}\alpha, 1)$ is a vector of $\mathcal{L}(\mathbf{M})$ and we can expect it to be short since $0 \leq t_i \cdot \alpha \bmod p - y_i \leq p/2^l$

Lattice reduction algorithms such as LLL or BKZ are then likely to exhibit $\mathbf{v}$ and thus, reveal α

LATTICE-BASED FAULT ATTACK AGAINST RSA ENCRYPTION

2

RSA ENCRYPTION

Let $(n = pq, e)$ be an RSA public key and m a secret message

RSA encryption of m is done as follows:

$$c = m^e \bmod n$$

m can then only be recovered with the knowledge of the private key $d = e^{-1} \bmod \varphi(n)$

$$m = c^d \bmod n$$

The only known attack against the public exponentiation is a perturbation of the modulus n (Berzati *et al.* 2008)

We then assume a simple left-to-right Square & Multiply exponentiation

MODULAR EXPONENTIATION

Algorithm 1: L2R Square & Multiply

Inputs: $m, e = (e_{k-1}, \ldots, e_0)_2, n$
Result: $m^e \bmod n$
$A \leftarrow 1 \mod n$
for $i = k - 1$ ***to*** 0 **do**
  $A \leftarrow A^2 \mod n$
  if $(e_i = 1)$ **then**
    $A \leftarrow A \cdot m \mod n$
  end
end
return A

MODULAR EXPONENTIATION

Algorithm 2: L2R Square & Multiply

Inputs: $m, e = (e_{k-1}, \ldots, e_0)_2, n$
Result: $m^e \bmod n$
$A \leftarrow 1 \mod n$
for $i = k - 1$ ***to*** 0 **do**
  $A \leftarrow A^2 \mod n$
  if $(e_i = 1)$ **then**
    $A \leftarrow A \cdot m \mod n$
  end
end
return A

FAULT ATTACK ON RSA ENCRYPTION

For the last iteration of the loop, $e_0 = 1$ and so :

$$A \leftarrow A \cdot m = m^{e-1} \cdot m \bmod n$$

We consider in the following a **small** additive fault on the second operand of the multiplication:

$$\tilde{A} \leftarrow A \cdot \tilde{m} = m^{e-1} \cdot (m + \varepsilon_i) \bmod n$$

As a result, we get a faulty ciphertext

$$\begin{aligned} \tilde{c}_i &= m^{e-1} \cdot (m + \varepsilon_i) \bmod n \\ &= c + m^{e-1} \cdot \varepsilon_i \bmod n \end{aligned}$$

We can then build several equations of the form

$$\begin{aligned} \varepsilon_i &= (\tilde{c}_i - c)(m^{e-1})^{-1} \bmod n \\ &= (\tilde{c}_i - c)m^{1-e} \bmod n \end{aligned}$$

FROM FAULTY OUTPUTS TO AN HNP INSTANCE

Hidden Number Problem

These equations can be seen as an instance of the HNP:

$$0 \leq \varepsilon_i = \alpha \cdot t_i \bmod n \leq \mathcal{B},$$

with

- $t_i = \tilde{c}_i - c \bmod n$
- $\alpha = m^{1-e} \bmod n$, the hidden number.

We can now remark that

$$c \cdot \alpha = m^e \cdot m^{1-e} = m \bmod n$$

That is to say, one can decrypt c by multiplying it by α

SIMULATIONS

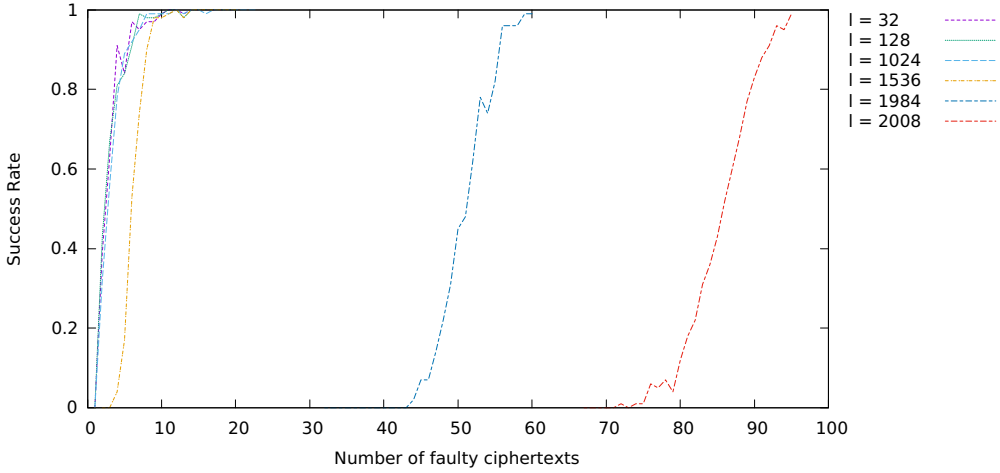

Figure: FA-LLL on RSA encryption success rate depending on error size and available number of faulty ciphertexts.

VARIATIONS

Note that m^{e-1} can be recovered directly by skipping the last multiplication of the exponentiation loop

The attack can also be adapted by faulting any operand of the square or multiply operation, or even by skipping entirely the last loop iteration

$$((m^{\frac{e-1}{2}})^2)^{-1} \cdot c = (m^{e-1})^{-1} \cdot c = m^{1-e} \cdot m^e = m \bmod n$$

We illustrate here the case of the classic left-to-right *S&M* algorithm, but parsing the exponent the other way around or even using windowing does not prevent the attack

COUNTERMEASURES & LIMITATIONS

Obviously, traditional RSA fault countermeasures should be effective at detecting our attack

We note that, in the context of encryption, repeating the operation is an efficient approach since the public exponent is usually small

Yet, another factor can make the attack fail:

- using a correct padding scheme
- PKCS #1 v2.2 mandates the introduction of random values in the message encoding when using the public exponent
- the assumption that the same message is encrypted is not met anymore

Finally, another limitation is that lattice attacks are not robust to errors. So if the fault model is not correct, the attack will fail

CONCLUSION
3

CONCLUSION

FA on RSA encryption

- Can be obtained with a standard FA or with the help of lattice reduction
- Extends the known attack surface for modular exponentiation with a public key
- Seems to withstand usual SCA countermeasures

FA on RSA signature

- Extends the known attack surface for CRT-based signature generation
- Prevented by SCA countermeasures on the recombination

In both case using a padding scheme introducing some random makes the attack fail

FA-LLL can be a promising tool

- To find new attack paths
- To exploit faulty outputs with a quite large error

Thank You !

IDEMIA

Join us on

www.idemia.com