Elliptic Curve Cryptography and Security of Embedded Devices

Ph.D. Defense

Vincent Verneuil

Institut de Mathématiques de Bordeaux
Inside Secure

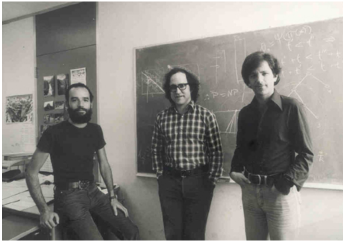
June 13th, 2012

Outline

Outline

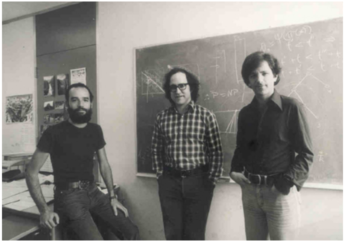
Outline

RSA (Rivest-Shamir-Adleman)



A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, 1978.

RSA (Rivest-Shamir-Adleman)



A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, 1978.

Key generation

- pick at random two primes p and q, and compute $n = p \times q$
- choose e and compute d such that: $e \times d \equiv 1 \mod (p-1)(q-1)$

Public key

 = $\{n, e\}$

Private key

 = $\{p, q, d\}$

RSA (Rivest-Shamir-Adleman)

Encryption / Decryption

To encrypt a message m:

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

To decrypt c:

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

RSA (Rivest-Shamir-Adleman)

Encryption / Decryption

To encrypt a message m:

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

To decrypt c:

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

Security assumption

Given 🔑 = $\{n, e\}$, how to recover $d = e^{-1} \mod (p-1)(q-1)$?

Factorize n to recover p and q !

Elliptic Curve Cryptography

Independently introduced by Koblitz and Miller in 1985.

Elliptic Curve Equation

Let $\mathbb{K}$ be a field, and $\mathscr{E}/\mathbb{K}$ an elliptic curve.
Then the set of $\mathbb{K}$-rational points $\mathscr{E}(\mathbb{K}) \subset \mathbb{P}^2(\mathbb{K})$
is an abelian group, with neutral element $\mathscr{O}$.

On a field $\mathbb{K} = \mathbb{F}_p$, $p > 3$, it has an affine equation:

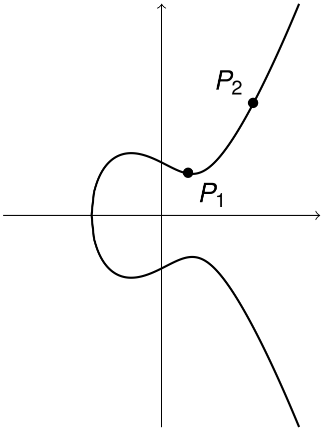
$$y^2 = x^3 + ax + b$$

Elliptic Curve Group Law

Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, $P_1, P_2 \neq \mathcal{O}$.

$P_3 = P_1 + P_2$ is given by:

$$\begin{cases} x_3 = m^2 - x_1 - x_2 \\ y_3 = m(x_1 - x_3) - y_1 \end{cases}$$



$\mathbb{K} = \mathbb{R}$

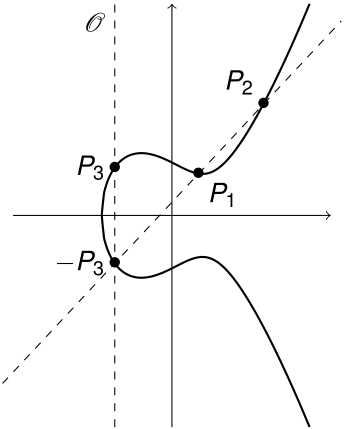
Elliptic Curve Group Law

Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, $P_1, P_2 \neq \mathscr{O}$.

$P_3 = P_1 + P_2$ is given by:

$$\begin{cases} x_3 = m^2 - x_1 - x_2 \\ y_3 = m(x_1 - x_3) - y_1 \end{cases}$$

$$m = \frac{y_2 - y_1}{x_2 - x_1} \quad \text{if} \quad P_1 \neq \pm P_2$$



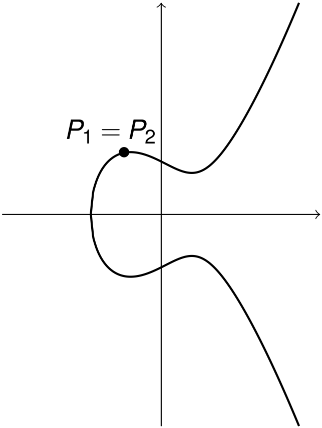
$\mathbb{K} = \mathbb{R}$

Elliptic Curve Group Law

Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, $P_1, P_2 \neq \mathscr{O}$.

$P_3 = P_1 + P_2$ is given by:

$$\begin{cases} x_3 = m^2 - x_1 - x_2 \\ y_3 = m(x_1 - x_3) - y_1 \end{cases}$$



$\mathbb{K} = \mathbb{R}$

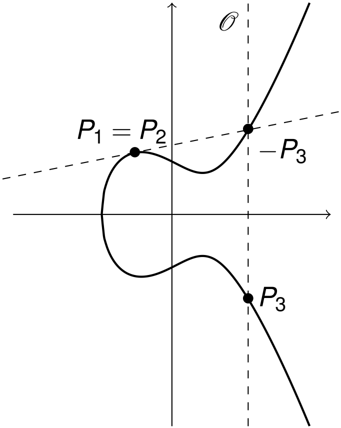
Elliptic Curve Group Law

Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, $P_1, P_2 \neq \mathcal{O}$.

$P_3 = P_1 + P_2$ is given by:

$$\begin{cases} x_3 = m^2 - x_1 - x_2 \\ y_3 = m(x_1 - x_3) - y_1 \end{cases}$$

$$m = \frac{3{x_1}^2 + a}{2y_1} \quad \text{if} \quad P_1 = P_2$$



$\mathbb{K} = \mathbb{R}$

Scalar Multiplication

Given a point P in $\mathscr{E}(\mathbb{K})$ and a positive integer d, we denote $dP = \underbrace{P + P + \cdots + P}_{d \text{ times}}$.

Scalar Multiplication

Given a point P in $\mathscr{E}(\mathbb{K})$ and a positive integer d,
we denote $dP = \underbrace{P + P + \cdots + P}_{d \text{ times}}$.

Elliptic Curve Discrete Logarithm Problem (ECDLP)

Given P in $\mathscr{E}(\mathbb{K})$ and dP, $1 \leq d \leq \#\mathscr{E}(\mathbb{K})$,
find d ?

Much harder than or factoring (which can be solved in subexponential time).

Cryptosystems Comparison

Estimated equivalent key lengths for ECC and RSA:

Security level	80	112	128	192	256
ECC	160	224	256	384	512
RSA	1024	2048	3072	8192	15360

- Very interesting in embedded devices having limited resources.

Outline

Embedded Devices Constraints

Efficiency

- Most transactions have to take less than 500 ms
- Small amount of RAM
- Very low power (hence low frequency) for contactless devices

Embedded Devices Constraints

Efficiency

- Most transactions have to take less than 500 ms
- Small amount of RAM
- Very low power (hence low frequency) for contactless devices

Arithmetic optimizations

- Exponentiation / scalar multiplication
- Group operations and point representation
- Modular arithmetic

$\mathbb{F}_p$ Operations Theoretical Cost

Expensive operations

Significant operations

Negligible operations

$\mathbb{F}_p$ Operations Theoretical Cost

Expensive operations

- Inversion (I)

Significant operations

Negligible operations

$\mathbb{F}_p$ Operations Theoretical Cost

Expensive operations

- Inversion (I)

Significant operations

- Multiplication (M)
- Squaring (S, $S/M \approx 0.8$)

Negligible operations

$\mathbb{F}_p$ Operations Theoretical Cost

Expensive operations

- Inversion (I)

Significant operations

- Multiplication (M)
- Squaring (S, $S/M \approx 0.8$)

Negligible operations

- Addition (A)
- Subtraction (A)
- Negation (N)

$\mathbb{F}_p$ Operations Theoretical Cost

Expensive operations

- Inversion (I)

Significant operations

- Multiplication (M)
- Squaring (S, $S/M \approx 0.8$)

Negligible operations

- Addition (A)
- Subtraction (A)
- Negation (N)

For ECC keylengths,
$A/M \approx 0.2$ and $N/M \approx 0.1$
on most smart cards.

Left-to-right

$$m^d = m^{d_0} \times \left(m^{d_1} \times \left(\dots \left(m^{d_{\ell-1}} \right)^2 \dots \right)^2 \right)^2$$

Input: $m, n, d \in \mathbb{N}$
Output: $m^d \mod n$
$a \leftarrow 1$
for $i = \ell - 1$ **to** 0 **do**
  $a \leftarrow a^2 \mod n$
  if $d_i = 1$ **then**
    $a \leftarrow a \times m \mod n$
return a

Right-to-left

$$m^d = m^{d_{\ell-1} 2^{\ell-1}} \times m^{d_{\ell-2} 2^{\ell-2}} \times \dots \times m^{d_0}$$

Input: $m, n, d \in \mathbb{N}$
Output: $m^d \mod n$
$a \leftarrow 1$; $b \leftarrow m$
for $i = 0$ **to** $\ell - 1$ **do**
  if $d_i = 1$ **then**
    $a \leftarrow a \times b \mod n$
  $b \leftarrow b^2 \mod n$
return a

Scalar Multiplication Algorithms

Double and add

Left-to-right

$dP = d_0 P + 2(d_1 P + 2(\ldots + 2(d_{\ell-1} P)\ldots))$

Input: $P \in \mathscr{E}(\mathbb{K}), d \in \mathbb{N}$
Output: dP
$R \leftarrow \mathscr{O}$
for $i = \ell - 1$ **to** 0 **do**
  $R \leftarrow 2R$
  if $d_i = 1$ **then**
    $R \leftarrow R + P$
return R

Right-to-left

$dP = d_{\ell-1} 2^{\ell-1} P + d_{\ell-2} 2^{\ell-2} P + \ldots + d_0 P$

Input: $P \in \mathscr{E}(\mathbb{K}), d \in \mathbb{N}$
Output: dP
$R \leftarrow \mathscr{O}$; $Q \leftarrow P$
for $i = 0$ **to** $\ell - 1$ **do**
  if $d_i = 1$ **then**
    $R \leftarrow R + Q$
  $Q \leftarrow 2Q$
return R

Non-Adjacent Form (NAF)

Signed representation minimizing the number of non-zero digits (1/3 vs 1/2).

Hence minimize the number of additions.

Refined Algorithms

Non-Adjacent Form (NAF)

Signed representation minimizing the number of non-zero digits (1/3 vs 1/2).

Hence minimize the number of additions.

Sliding window algorithms

Precompute $3P, 5P, \ldots$ to process several scalar bits at a time.

Can be combined with the NAF method.

Refined Algorithms

Non-Adjacent Form (NAF)

Signed representation minimizing the number of non-zero digits (1/3 vs 1/2).

Hence minimize the number of additions.

Sliding window algorithms

Precompute $3P, 5P, \ldots$ to process several scalar bits at a time.

Can be combined with the NAF method.

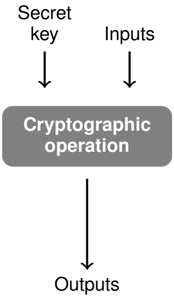
Co-Z Addition

Euclidean Addition Chains [Meloni, WAIFI 2007]

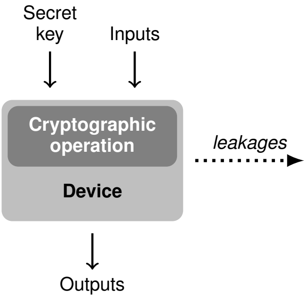
Co-Z binary ladder [Goundar, Joye & Miyaji, CHES 2010]

Outline

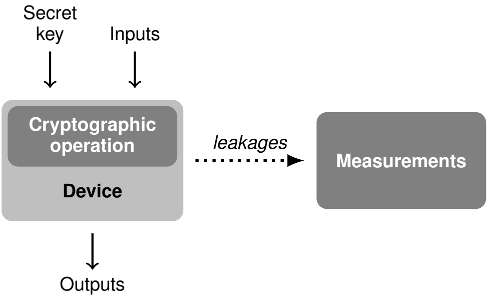
Side-Channel Analysis Framework



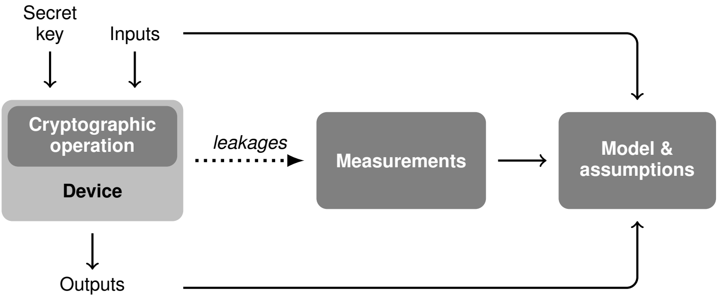
Side-Channel Analysis Framework



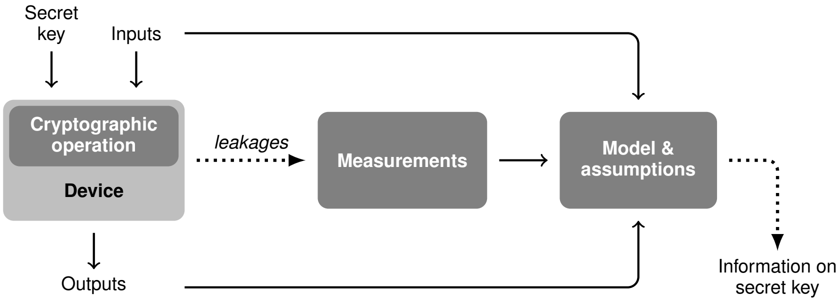
Side-Channel Analysis Framework



Side-Channel Analysis Framework



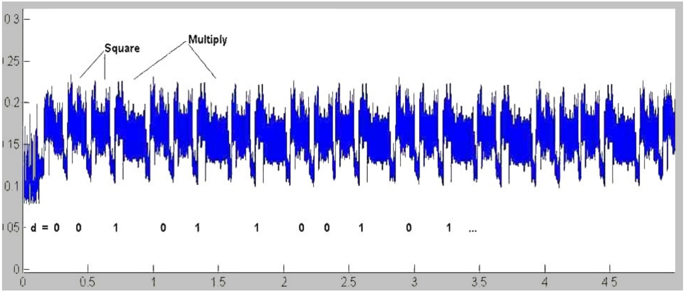
Side-Channel Analysis Framework



Simple Side-Channel Analysis (SSCA)

Left-to-right square & multiply

Side-channel leakage: power, EM, etc.



The whole exponent may be recovered using a single trace.

Square & multiply:

Regular Exponentiation

Left-to-right algorithms

Square & multiply:

Square & multiply always:

Regular Exponentiation

Left-to-right algorithms

Square & multiply:

Square & multiply always:

Montgomery ladder:

Regular Exponentiation Algorithms

Left-to-right

"Montgomery ladder"

Input: $m, n, d \in \mathbb{N}$
Output: $m^d \bmod n$

$R_0 \leftarrow 1$
$R_1 \leftarrow m$
for $i = \ell - 1$ **to** 0 **do**
  $R_{1-d_i} \leftarrow R_0 \times R_1 \bmod n$
  $R_{d_i} \leftarrow {R_{d_i}}^2 \bmod n$
return R_0

Right-to-left

"Joye ladder"

Input: $m, n, d \in \mathbb{N}$
Output: $m^d \bmod n$

$R_0 \leftarrow 1$
$R_1 \leftarrow m$
for $i = 0$ **to** $\ell - 1$ **do**
  $R_{1-d_i} \leftarrow {R_{1-d_i}}^2 \bmod n$
  $R_{1-d_i} \leftarrow R_{1-d_i} \times R_{d_i} \bmod n$
return R_0

Regular Scalar Multiplication

Left-to-right algorithms

Double & add:

Regular Scalar Multiplication

Left-to-right algorithms

Double & add:

Double & add always:

Regular Scalar Multiplication

Left-to-right algorithms

Double & add:

Double & add always:

Montgomery ladders:

Regular Scalar Multiplication Algorithms

Left-to-right

"Montgomery ladder"

Input: $P \in \mathscr{E}(\mathbb{K}), d \in \mathbb{N}$
Output: dP

1: $R_0 \leftarrow \mathscr{O}$
2: $R_1 \leftarrow P$
3: **for** $i = \ell - 1$ **to** 0 **do**
4: $\quad R_{1-d_i} \leftarrow R_0 + R_1$
5: $\quad R_{d_i} \leftarrow 2R_{d_i}$
6: **return** R_0

Right-to-left

"Joye ladder"

Input: $P \in \mathscr{E}(\mathbb{K}), d \in \mathbb{N}$
Output: dP

1: $R_0 \leftarrow \mathscr{O}$
2: $R_1 \leftarrow P$
3: **for** $i = 0$ **to** $\ell - 1$ **do**
4: $\quad R_{1-d_i} \leftarrow 2R_{1-d_i}$
5: $\quad R_{1-d_i} \leftarrow R_{1-d_i} + R_{d_i}$
6: **return** R_0

Regular Atomic Exponentiation

Square & multiply:

Regular Atomic Exponentiation

Square & multiply:

Atomic multiply always:

Regular Atomic Scalar Multiplication

Double & add:

Regular Atomic Scalar Multiplication

Double & add:

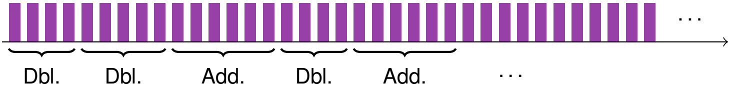
Atomic add always (with a unified group addition):

Regular Atomic Scalar Multiplication

Double & add:

Atomic add always (with a unified group addition):

Atomic scalar multiplication using a smaller pattern:



Leakage on Manipulated Data

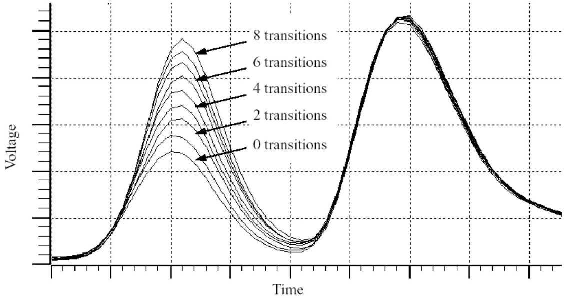


FIGURE 2. **Number of Bit Transitions versus Power Consumption**
These results show how the data effects the power levels. The nine overlayed waveforms correspond to the power traces of different data being accessed by an LDA instruction. These results were obtained by averaging the power signals across 500 samples in order to reduce the noise content. The difference in voltage between i transitions and $i+1$ transitions is about 6.5 mV.

Leakage on Manipulated Data

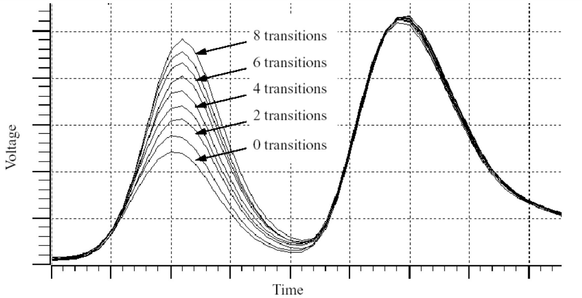


FIGURE 2. **Number of Bit Transitions versus Power Consumption**
These results show how the data effects the power levels. The nine overlayed waveforms correspond to the power traces of different data being accessed by an LDA instruction. These results were obtained by averaging the power signals across 500 samples in order to reduce the noise content. The difference in voltage between *i* transitions and *i*+1 transitions is about 6.5 mV.

Noise is generally too high to exploit this leakage directly ☹

Leakage on Manipulated Data

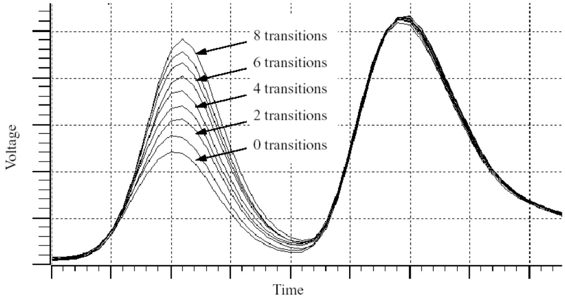


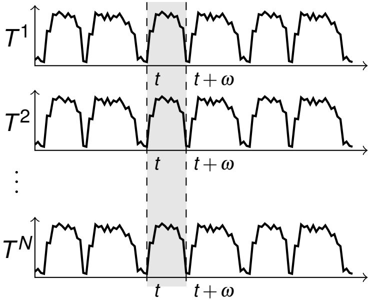
FIGURE 2. **Number of Bit Transitions versus Power Consumption**
These results show how the data effects the power levels. The nine overlayed waveforms correspond to the power traces of different data being accessed by an LDA instruction. These results were obtained by averaging the power signals across 500 samples in order to reduce the noise content. The difference in voltage between *i* transitions and *i*+1 transitions is about 6.5 mV.

Noise is generally too high to exploit this leakage directly ☹

➧ Many acquisitions are used to reduce noise influence

Differential Analysis Principle

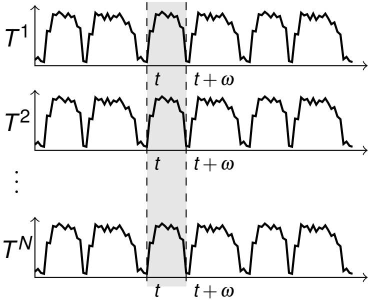
Measure N times a side-channel leakage with different data involved and consider the traces $T^1, T^2, \ldots, T^n$.



Differential Analysis Principle

Measure N times a side-channel leakage with different data involved and consider the traces $T^1, T^2, \ldots, T^n$.

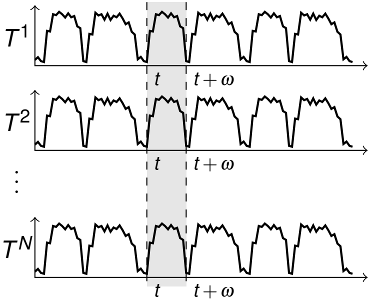
- align vertically the traces on the targeted operation using signal processing tools



Differential Analysis Principle

Measure N times a side-channel leakage with different data involved and consider the traces $T^1, T^2, \ldots, T^n$.

- align vertically the traces on the targeted operation using signal processing tools
- perform statistical treatment between traces, known inputs or outputs and a guess on a few key bits

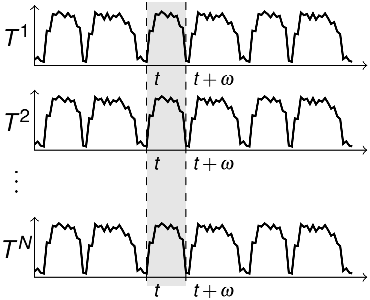


Differential Analysis Principle

Measure N times a side-channel leakage with different data involved and consider the traces $T^1, T^2, \ldots, T^n$.

- align vertically the traces on the targeted operation using signal processing tools
- perform statistical treatment between traces, known inputs or outputs and a guess on a few key bits

➧ Validate the guess or not



Differential Side-Channel Analysis

Original method introduced in [Kocher, Jaffe & Jun, CRYPTO'99]

- Hamming weight leakage model
- Difference of means as a distinguisher

Differential Side-Channel Analysis

Original method introduced in [Kocher, Jaffe & Jun, CRYPTO'99]

- Hamming weight leakage model
- Difference of means as a distinguisher

Correlation analysis introduced in [Brier, Clavier & Olivier, CHES 2004]

- Hamming weight/distance leakage model
- Pearson correlation factor as a distinguisher

Countermeasures for RSA Exponentiation

- Exponent blinding $d' = d + r(p-1)(q-1)$

Countermeasures for RSA Exponentiation

- Exponent blinding $d' = d + r(p-1)(q-1)$
- Message/ciphertext additive blinding $m' = m + rn \bmod cn$, $r < c$

Countermeasures for RSA Exponentiation

- Exponent blinding $d' = d + r(p-1)(q-1)$
- Message/ciphertext additive blinding $m' = m + rn \bmod cn$, $r < c$
- Message/ciphertext multiplicative blinding $m' = r^e m \bmod n$, result recovered as $r^{-1}(m')^d \bmod n$

Countermeasures for Scalar Multiplication

From [Coron, CHES'99]:

- Scalar blinding $d' = d + r\#\mathscr{E}(\mathbb{F}_p)$

Countermeasures for Scalar Multiplication

From [Coron, CHES'99]:

- Scalar blinding $d' = d + r\#\mathscr{E}(\mathbb{F}_p)$
- Base point projective coordinates blinding $(r^2X : r^3Y : rZ)$

Countermeasures for Scalar Multiplication

From [Coron, CHES'99]:

- Scalar blinding $d' = d + r\#\mathscr{E}(\mathbb{F}_p)$
- Base point projective coordinates blinding $(r^2X : r^3Y : rZ)$
- Input point blinding $Q = d(P + R)$, result recovered as $Q - S$ with $S = dR$

Outline

Our Contribution

- New atomic pattern for right-to-left scalar multiplication implementation
- Fastest implementation for standard curves considering addition cost $A/M \geq 0.1$

Our Contribution

- New atomic pattern for right-to-left scalar multiplication implementation
- Fastest implementation for standard curves considering addition cost $A/M \geq 0.1$

Theoretical comparison ($S/M = 0.8$, $A/M = 0.2$)

Previous right-to-left NAF atomic scalar multiplication: - 20 % (M/bit)

Best previous scalar multiplication (Co-Z Montgomery ladder $(X:Z)$-only): - 3.6 % (M/bit)

Atomic Right-to-Left Scalar Multiplication

Mixed coordinates

■ [
- Multiplication
- Addition
- Negation
- Addition

]

Operations expression using the atomic pattern

Addition :	■■■■■■■■■■■■■■■■	[11M+5S]
Doubling :	■■■■■■■■	[3M+5S]

Atomic Right-to-Left Scalar Multiplication

Mixed coordinates

■ = [
- Multiplication
- Addition
- Negation
- Addition

]

■ = [
- Squaring
- Addition
- Negation
- Addition

]

Operations expression using the atomic pattern

Addition :	■■■■■■■■■■■■■■■■	[11M+5S]
Doubling :	■■■■■■■■	[3M+5S]

Atomic Right-to-Left Scalar Multiplication

Mixed coordinates

- Multiplication
- Addition
- Negation
- Addition

- Squaring
- Addition
- Negation
- Addition

Operations expression using the atomic pattern

Addition : [11M+5S]

Doubling : [3M+5S]

Atomic Right-to-Left Scalar Multiplication

Mixed coordinates

- Multiplication
- Addition
- Negation
- Addition

- Squaring
- Addition
- Negation
- Addition

Operations expression using the atomic pattern

Addition :	[11M+5S]
Doubling :	[3M+5S]

Extended pattern :

Atomic Right-to-Left Scalar Multiplication

- Sq.
- Add.
- Neg.
- Add.
- Mult.
- Add.
- Neg.
- Add.
- Mult.
- Add.
- Neg.
- Add.
- Mult.
- Add.
- Neg.
- Add.
- Sq.
- Add.
- Neg.
- Add.
- Mult.
- Add.
- Neg.
- Add.
- Mult.
- Add.
- Neg.
- Add.
- Mult.
- Add.
- Neg.
- Add.

Atomic Right-to-Left Scalar Multiplication

	Add. 1	Add. 2
▶Sq.	$R_1 \leftarrow {Z_2}^2$	$R_6 \leftarrow {R_4}^2$
▶Add.	$\star$	$\star$
▶Neg.	$\star$	$\star$
▶Add.	$\star$	$\star$
▶Mult.	$R_2 \leftarrow X_1 \cdot R_1$	$R_5 \leftarrow Z_1 \cdot Z_2$
▶Add.	$\star$	$\star$
▶Neg.	$\star$	$\star$
▶Add.	$\star$	$\star$
▶Mult.	$R_1 \leftarrow R_1 \cdot Z_2$	$Z_3 \leftarrow R_5 \cdot R_4$
▶Add.	$\star$	$\star$
▶Neg.	$\star$	$\star$
▶Add.	$\star$	$\star$
▶Mult.	$R_3 \leftarrow Y_1 \cdot R_1$	$R_2 \leftarrow R_2 \cdot R_6$
▶Add.	$\star$	$\star$
▶Neg.	$\star$	$R_1 \leftarrow -R_1$
▶Add.	$\star$	$\star$
▶Sq.	$R_1 \leftarrow {Z_1}^2$	$R_5 \leftarrow {R_1}^2$
▶Add.	$\star$	$\star$
▶Neg.	$\star$	$R_3 \leftarrow -R_3$
▶Add.	$\star$	$\star$
▶Mult.	$R_4 \leftarrow R_1 \cdot X_2$	$R_4 \leftarrow R_4 \cdot R_6$
▶Add.	$\star$	$R_6 \leftarrow R_5 + R_4$
▶Neg.	$R_4 \leftarrow -R_4$	$R_2 \leftarrow -R_2$
▶Add.	$R_4 \leftarrow R_2 + R_4$	$R_6 \leftarrow R_6 + R_2$
▶Mult.	$R_1 \leftarrow Z_1 \cdot R_1$	$R_3 \leftarrow R_3 \cdot R_4$
▶Add.	$\star$	$X_3 \leftarrow R_2 + R_6$
▶Neg.	$\star$	$\star$
▶Add.	$\star$	$R_2 \leftarrow X_3 + R_2$
▶Mult.	$R_1 \leftarrow R_1 \cdot Y_2$	$R_1 \leftarrow R_1 \cdot R_2$
▶Add.	$\star$	$Y_3 \leftarrow R_3 + R_1$
▶Neg.	$R_1 \leftarrow -R_1$	$\star$
▶Add.	$R_1 \leftarrow R_3 + R_1$	$\star$

Atomic Right-to-Left Scalar Multiplication

	Add. 1	Add. 2	Dbl.
▶Sq.	$R_1 \leftarrow Z_2{}^2$	$R_6 \leftarrow R_4{}^2$	$R_1 \leftarrow X_1{}^2$
▶Add.	$\star$	$\star$	$R_2 \leftarrow Y_1 + Y_1$
▶Neg.	$\star$	$\star$	$\star$
▶Add.	$\star$	$\star$	$\star$
▶Mult.	$R_2 \leftarrow X_1 \cdot R_1$	$R_5 \leftarrow Z_1 \cdot Z_2$	$Z_2 \leftarrow R_2 \cdot Z_1$
▶Add.	$\star$	$\star$	$R_4 \leftarrow R_1 + R_1$
▶Neg.	$\star$	$\star$	$\star$
▶Add.	$\star$	$\star$	$\star$
▶Mult.	$R_1 \leftarrow R_1 \cdot Z_2$	$Z_3 \leftarrow R_5 \cdot R_4$	$R_3 \leftarrow R_2 \cdot Y_1$
▶Add.	$\star$	$\star$	$R_6 \leftarrow R_3 + R_3$
▶Neg.	$\star$	$\star$	$\star$
▶Add.	$\star$	$\star$	$\star$
▶Mult.	$R_3 \leftarrow Y_1 \cdot R_1$	$R_2 \leftarrow R_2 \cdot R_6$	$R_2 \leftarrow R_6 \cdot R_3$
▶Add.	$\star$	$\star$	$R_1 \leftarrow R_4 + R_1$
▶Neg.	$\star$	$R_1 \leftarrow -R_1$	$\star$
▶Add.	$\star$	$\star$	$R_1 \leftarrow R_1 + W_1$
▶Sq.	$R_1 \leftarrow Z_1{}^2$	$R_5 \leftarrow R_1{}^2$	$R_3 \leftarrow R_1{}^2$
▶Add.	$\star$	$\star$	$\star$
▶Neg.	$\star$	$R_3 \leftarrow -R_3$	$\star$
▶Add.	$\star$	$\star$	$\star$
▶Mult.	$R_4 \leftarrow R_1 \cdot X_2$	$R_4 \leftarrow R_4 \cdot R_6$	$R_4 \leftarrow R_6 \cdot X_1$
▶Add.	$\star$	$R_6 \leftarrow R_5 + R_4$	$R_5 \leftarrow W_1 + W_1$
▶Neg.	$R_4 \leftarrow -R_4$	$R_2 \leftarrow -R_2$	$R_4 \leftarrow -R_4$
▶Add.	$R_4 \leftarrow R_2 + R_4$	$R_6 \leftarrow R_6 + R_2$	$R_3 \leftarrow R_3 + R_4$
▶Mult.	$R_1 \leftarrow Z_1 \cdot R_1$	$R_3 \leftarrow R_3 \cdot R_4$	$W_2 \leftarrow R_2 \cdot R_5$
▶Add.	$\star$	$X_3 \leftarrow R_2 + R_6$	$X_2 \leftarrow R_3 + R_4$
▶Neg.	$\star$	$\star$	$R_2 \leftarrow -R_2$
▶Add.	$\star$	$R_2 \leftarrow X_3 + R_2$	$R_6 \leftarrow R_4 + X_2$
▶Mult.	$R_1 \leftarrow R_1 \cdot Y_2$	$R_1 \leftarrow R_1 \cdot R_2$	$R_4 \leftarrow R_6 \cdot R_1$
▶Add.	$\star$	$Y_3 \leftarrow R_3 + R_1$	$\star$
▶Neg.	$R_1 \leftarrow -R_1$	$\star$	$R_4 \leftarrow -R_4$
▶Add.	$R_1 \leftarrow R_3 + R_1$	$\star$	$Y_2 \leftarrow R_4 + R_2$

Atomic Right-to-Left Scalar Multiplication

	Add. 1	Add. 2	Dbl.
▶Sq.	$R_1 \leftarrow {Z_2}^2$	$R_6 \leftarrow {R_4}^2$	$R_1 \leftarrow {X_1}^2$
▶Add.	$\star$	$\star$	$R_2 \leftarrow Y_1 + Y_1$
▶Neg.	$\star$	$\star$	$\star$
▶Add.	$\star$	$\star$	$\star$
▶Mult.	$R_2 \leftarrow X_1 \cdot R_1$	$R_5 \leftarrow Z_1 \cdot Z_2$	$Z_2 \leftarrow R_2 \cdot Z_1$
▶Add.	$\star$	$\star$	$R_4 \leftarrow R_1 + R_1$
▶Neg.	$\star$	$\star$	$\star$
▶Add.	$\star$	$\star$	$\star$
▶Mult.	$R_1 \leftarrow R_1 \cdot Z_2$	$Z_3 \leftarrow R_5 \cdot R_4$	$R_3 \leftarrow R_2 \cdot Y_1$
▶Add.	$\star$	$\star$	$R_6 \leftarrow R_3 + R_3$
▶Neg.	$\star$	$\star$	$\star$
▶Add.	$\star$	$\star$	$\star$
▶Mult.	$R_3 \leftarrow Y_1 \cdot R_1$	$R_2 \leftarrow R_2 \cdot R_6$	$R_2 \leftarrow R_6 \cdot R_3$
▶Add.	$\star$	$\star$	$R_1 \leftarrow R_4 + R_1$
▶Neg.	$\star$	$R_1 \leftarrow -R_1$	$\star$
▶Add.	$\star$	$\star$	$R_1 \leftarrow R_1 + W_1$
▶Sq.	$R_1 \leftarrow {Z_1}^2$	$R_5 \leftarrow {R_1}^2$	$R_3 \leftarrow {R_1}^2$
▶Add.	$\star$	$\star$	$\star$
▶Neg.	$\star$	$R_3 \leftarrow -R_3$	$\star$
▶Add.	$\star$	$\star$	$\star$
▶Mult.	$R_4 \leftarrow R_1 \cdot X_2$	$R_4 \leftarrow R_4 \cdot R_6$	$R_4 \leftarrow R_6 \cdot X_1$
▶Add.	$\star$	$R_6 \leftarrow R_5 + R_4$	$R_5 \leftarrow W_1 + W_1$
▶Neg.	$R_4 \leftarrow -R_4$	$R_2 \leftarrow -R_2$	$R_4 \leftarrow -R_4$
▶Add.	$R_4 \leftarrow R_2 + R_4$	$R_6 \leftarrow R_6 + R_2$	$R_3 \leftarrow R_3 + R_4$
▶Mult.	$R_1 \leftarrow Z_1 \cdot R_1$	$R_3 \leftarrow R_3 \cdot R_4$	$W_2 \leftarrow R_2 \cdot R_5$
▶Add.	$\star$	$X_3 \leftarrow R_2 + R_6$	$X_2 \leftarrow R_3 + R_4$
▶Neg.	$\star$	$\star$	$R_2 \leftarrow -R_2$
▶Add.	$\star$	$R_2 \leftarrow X_3 + R_2$	$R_6 \leftarrow R_4 + X_2$
▶Mult.	$R_1 \leftarrow R_1 \cdot Y_2$	$R_1 \leftarrow R_1 \cdot R_2$	$R_4 \leftarrow R_6 \cdot R_1$
▶Add.	$\star$	$Y_3 \leftarrow R_3 + R_1$	$\star$
▶Neg.	$R_1 \leftarrow -R_1$	$\star$	$R_4 \leftarrow -R_4$
▶Add.	$R_1 \leftarrow R_3 + R_1$	$\star$	$Y_2 \leftarrow R_4 + R_2$

Atomic Right-to-Left Scalar Multiplication

	Add. 1	Add. 2	Dbl.
▶Sq.	$R_1 \leftarrow {Z_2}^2$	$R_6 \leftarrow {R_4}^2$	$R_1 \leftarrow {X_1}^2$
▶Add.	$\star$	$\star$	$R_2 \leftarrow Y_1 + Y_1$
▶Mult.	$R_2 \leftarrow X_1 \cdot R_1$	$R_5 \leftarrow Z_1 \cdot Z_2$	$Z_2 \leftarrow R_2 \cdot Z_1$
▶Add.	$\star$	$\star$	$R_4 \leftarrow R_1 + R_1$
▶Mult.	$R_1 \leftarrow R_1 \cdot Z_2$	$Z_3 \leftarrow R_5 \cdot R_4$	$R_3 \leftarrow R_2 \cdot Y_1$
▶Add.	$\star$	$\star$	$R_6 \leftarrow R_3 + R_3$
▶Mult.	$R_3 \leftarrow Y_1 \cdot R_1$	$R_2 \leftarrow R_2 \cdot R_6$	$R_2 \leftarrow R_6 \cdot R_3$
▶Add.	$\star$	$\star$	$R_1 \leftarrow R_4 + R_1$
▶Neg.	$\star$	$R_1 \leftarrow -R_1$	$\star$
▶Add.	$\star$	$\star$	$R_1 \leftarrow R_1 + W_1$
▶Sq.	$R_1 \leftarrow {Z_1}^2$	$R_5 \leftarrow {R_1}^2$	$R_3 \leftarrow {R_1}^2$
▶Neg.	$\star$	$R_3 \leftarrow -R_3$	$\star$
▶Mult.	$R_4 \leftarrow R_1 \cdot X_2$	$R_4 \leftarrow R_4 \cdot R_6$	$R_4 \leftarrow R_6 \cdot X_1$
▶Add.	$\star$	$R_6 \leftarrow R_5 + R_4$	$R_5 \leftarrow W_1 + W_1$
▶Neg.	$R_4 \leftarrow -R_4$	$R_2 \leftarrow -R_2$	$R_4 \leftarrow -R_4$
▶Add.	$R_4 \leftarrow R_2 + R_4$	$R_6 \leftarrow R_6 + R_2$	$R_3 \leftarrow R_3 + R_4$
▶Mult.	$R_1 \leftarrow Z_1 \cdot R_1$	$R_3 \leftarrow R_3 \cdot R_4$	$W_2 \leftarrow R_2 \cdot R_5$
▶Add.	$\star$	$X_3 \leftarrow R_2 + R_6$	$X_2 \leftarrow R_3 + R_4$
▶Neg.	$\star$	$\star$	$R_2 \leftarrow -R_2$
▶Add.	$\star$	$R_2 \leftarrow X_3 + R_2$	$R_6 \leftarrow R_4 + X_2$
▶Mult.	$R_1 \leftarrow R_1 \cdot Y_2$	$R_1 \leftarrow R_1 \cdot R_2$	$R_4 \leftarrow R_6 \cdot R_1$
▶Add.	$\star$	$Y_3 \leftarrow R_3 + R_1$	$\star$
▶Neg.	$R_1 \leftarrow -R_1$	$\star$	$R_4 \leftarrow -R_4$
▶Add.	$R_1 \leftarrow R_3 + R_1$	$\star$	$Y_2 \leftarrow R_4 + R_2$

Atomic Right-to-Left Scalar Multiplication

	Add. 1	Add. 2	Dbl.
Sq.	$R_1 \leftarrow Z_2{}^2$	$R_1 \leftarrow R_6{}^2$	$R_1 \leftarrow X_1{}^2$
Add.	$\star$	$\star$	$R_2 \leftarrow Y_1 + Y_1$
Mult.	$R_2 \leftarrow Y_1 \cdot Z_2$	$R_4 \leftarrow R_5 \cdot R_1$	$Z_2 \leftarrow R_2 \cdot Z_1$
Add.	$\star$	$\star$	$R_4 \leftarrow R_1 + R_1$
Mult.	$R_5 \leftarrow Y_2 \cdot Z_1$	$R_5 \leftarrow R_1 \cdot R_6$	$R_3 \leftarrow R_2 \cdot Y_1$
Add.	$\star$	$\star$	$R_6 \leftarrow R_3 + R_3$
Mult.	$R_3 \leftarrow R_1 \cdot R_2$	$R_1 \leftarrow Z_1 \cdot R_6$	$R_2 \leftarrow R_6 \cdot R_3$
Add.	$\star$	$\star$	$R_1 \leftarrow R_4 + R_1$
Add.	$\star$	$\star$	$R_1 \leftarrow R_1 + W_1$
Sq.	$R_4 \leftarrow Z_1{}^2$	$R_6 \leftarrow R_2{}^2$	$R_3 \leftarrow R_1{}^2$
Mult.	$R_2 \leftarrow R_5 \cdot R_4$	$Z_3 \leftarrow R_1 \cdot Z_2$	$R_4 \leftarrow R_6 \cdot X_1$
Add.	$\star$	$R_1 \leftarrow R_4 + R_4$	$R_5 \leftarrow W_1 + W_1$
Sub.	$R_2 \leftarrow R_2 - R_3$	$R_6 \leftarrow R_6 - R_1$	$R_3 \leftarrow R_3 - R_4$
Mult.	$R_5 \leftarrow R_1 \cdot X_1$	$R_1 \leftarrow R_5 \cdot R_3$	$W_2 \leftarrow R_2 \cdot R_5$
Sub.	$\star$	$X_3 \leftarrow R_6 - R_5$	$X_2 \leftarrow R_3 - R_4$
Sub.	$\star$	$R_4 \leftarrow R_4 - X_3$	$R_6 \leftarrow R_4 - X_2$
Mult.	$R_6 \leftarrow X_2 \cdot R_4$	$R_3 \leftarrow R_4 \cdot R_2$	$R_4 \leftarrow R_6 \cdot R_1$
Sub.	$R_6 \leftarrow R_6 - R_5$	$Y_3 \leftarrow R_3 - R_1$	$Y_2 \leftarrow R_4 - R_2$

Atomic Right-to-Left Scalar Multiplication

	Add. 1	Add. 2	Dbl.
Sq.	$R_1 \leftarrow {Z_2}^2$	$R_1 \leftarrow {R_6}^2$	$R_1 \leftarrow {X_1}^2$
Add.	$\star$	$\star$	$R_2 \leftarrow Y_1 + Y_1$
Mult.	$R_2 \leftarrow Y_1 \cdot Z_2$	$R_4 \leftarrow R_5 \cdot R_1$	$Z_2 \leftarrow R_2 \cdot Z_1$
Add.	$\star$	$\star$	$R_4 \leftarrow R_1 + R_1$
Mult.	$R_5 \leftarrow Y_2 \cdot Z_1$	$R_5 \leftarrow R_1 \cdot R_6$	$R_3 \leftarrow R_2 \cdot Y_1$
Add.	$\star$	$\star$	$R_6 \leftarrow R_3 + R_3$
Mult.	$R_3 \leftarrow R_1 \cdot R_2$	$R_1 \leftarrow Z_1 \cdot R_6$	$R_2 \leftarrow R_6 \cdot R_3$
Add.	$\star$	$\star$	$R_1 \leftarrow R_4 + R_1$
Add.	$\star$	$\star$	$R_1 \leftarrow R_1 + W_1$
Sq.	$R_4 \leftarrow {Z_1}^2$	$R_6 \leftarrow {R_2}^2$	$R_3 \leftarrow {R_1}^2$
Mult.	$R_2 \leftarrow R_5 \cdot R_4$	$Z_3 \leftarrow R_1 \cdot Z_2$	$R_4 \leftarrow R_6 \cdot X_1$
Add.	$\star$	$R_1 \leftarrow R_4 + R_4$	$R_5 \leftarrow W_1 + W_1$
Sub.	$R_2 \leftarrow R_2 - R_3$	$R_6 \leftarrow R_6 - R_1$	$R_3 \leftarrow R_3 - R_4$
Mult.	$R_5 \leftarrow R_1 \cdot X_1$	$R_1 \leftarrow R_5 \cdot R_3$	$W_2 \leftarrow R_2 \cdot R_5$
Sub.	$\star$	$X_3 \leftarrow R_6 - R_5$	$X_2 \leftarrow R_3 - R_4$
Sub.	$\star$	$R_4 \leftarrow R_4 - X_3$	$R_6 \leftarrow R_4 - X_2$
Mult.	$R_6 \leftarrow X_2 \cdot R_4$	$R_3 \leftarrow R_4 \cdot R_2$	$R_4 \leftarrow R_6 \cdot R_1$
Sub.	$R_6 \leftarrow R_6 - R_5$	$Y_3 \leftarrow R_3 - R_1$	$Y_2 \leftarrow R_4 - R_2$

8 multiplications $\rightarrow$ 6 multiplications + 2 squarings

Atomic Right-to-Left Scalar Multiplication

	Add. 1	Add. 2	Dbl.
Sq.	$R_1 \leftarrow {Z_2}^2$	$R_1 \leftarrow {R_6}^2$	$R_1 \leftarrow {X_1}^2$
Add.	$\star$	$\star$	$R_2 \leftarrow Y_1 + Y_1$
Mult.	$R_2 \leftarrow Y_1 \cdot Z_2$	$R_4 \leftarrow R_5 \cdot R_1$	$Z_2 \leftarrow R_2 \cdot Z_1$
Add.	$\star$	$\star$	$R_4 \leftarrow R_1 + R_1$
Mult.	$R_5 \leftarrow Y_2 \cdot Z_1$	$R_5 \leftarrow R_1 \cdot R_6$	$R_3 \leftarrow R_2 \cdot Y_1$
Add.	$\star$	$\star$	$R_6 \leftarrow R_3 + R_3$
Mult.	$R_3 \leftarrow R_1 \cdot R_2$	$R_1 \leftarrow Z_1 \cdot R_6$	$R_2 \leftarrow R_6 \cdot R_3$
Add.	$\star$	$\star$	$R_1 \leftarrow R_4 + R_1$
Add.	$\star$	$\star$	$R_1 \leftarrow R_1 + W_1$
Sq.	$R_4 \leftarrow {Z_1}^2$	$R_6 \leftarrow {R_2}^2$	$R_3 \leftarrow {R_1}^2$
Mult.	$R_2 \leftarrow R_5 \cdot R_4$	$Z_3 \leftarrow R_1 \cdot Z_2$	$R_4 \leftarrow R_6 \cdot X_1$
Add.	$\star$	$R_1 \leftarrow R_4 + R_4$	$R_5 \leftarrow W_1 + W_1$
Sub.	$R_2 \leftarrow R_2 - R_3$	$R_6 \leftarrow R_6 - R_1$	$R_3 \leftarrow R_3 - R_4$
Mult.	$R_5 \leftarrow R_1 \cdot X_1$	$R_1 \leftarrow R_5 \cdot R_3$	$W_2 \leftarrow R_2 \cdot R_5$
Sub.	$\star$	$X_3 \leftarrow R_6 - R_5$	$X_2 \leftarrow R_3 - R_4$
Sub.	$\star$	$R_4 \leftarrow R_4 - X_3$	$R_6 \leftarrow R_4 - X_2$
Mult.	$R_6 \leftarrow X_2 \cdot R_4$	$R_3 \leftarrow R_4 \cdot R_2$	$R_4 \leftarrow R_6 \cdot R_1$
Sub.	$R_6 \leftarrow R_6 - R_5$	$Y_3 \leftarrow R_3 - R_1$	$Y_2 \leftarrow R_4 - R_2$

8 multiplications $\rightarrow$ 6 multiplications + 2 squarings

16 additions $\rightarrow$ 6 additions + 4 subtractions

Atomic Right-to-Left Scalar Multiplication

	Add. 1	Add. 2	Dbl.
Sq.	$R_1 \leftarrow {Z_2}^2$	$R_1 \leftarrow {R_6}^2$	$R_1 \leftarrow {X_1}^2$
Add.	$\star$	$\star$	$R_2 \leftarrow Y_1 + Y_1$
Mult.	$R_2 \leftarrow Y_1 \cdot Z_2$	$R_4 \leftarrow R_5 \cdot R_1$	$Z_2 \leftarrow R_2 \cdot Z_1$
Add.	$\star$	$\star$	$R_4 \leftarrow R_1 + R_1$
Mult.	$R_5 \leftarrow Y_2 \cdot Z_1$	$R_5 \leftarrow R_1 \cdot R_6$	$R_3 \leftarrow R_2 \cdot Y_1$
Add.	$\star$	$\star$	$R_6 \leftarrow R_3 + R_3$
Mult.	$R_3 \leftarrow R_1 \cdot R_2$	$R_1 \leftarrow Z_1 \cdot R_6$	$R_2 \leftarrow R_6 \cdot R_3$
Add.	$\star$	$\star$	$R_1 \leftarrow R_4 + R_1$
Add.	$\star$	$\star$	$R_1 \leftarrow R_1 + W_1$
Sq.	$R_4 \leftarrow {Z_1}^2$	$R_6 \leftarrow {R_2}^2$	$R_3 \leftarrow {R_1}^2$
Mult.	$R_2 \leftarrow R_5 \cdot R_4$	$Z_3 \leftarrow R_1 \cdot Z_2$	$R_4 \leftarrow R_6 \cdot X_1$
Add.	$\star$	$R_1 \leftarrow R_4 + R_4$	$R_5 \leftarrow W_1 + W_1$
Sub.	$R_2 \leftarrow R_2 - R_3$	$R_6 \leftarrow R_6 - R_1$	$R_3 \leftarrow R_3 - R_4$
Mult.	$R_5 \leftarrow R_1 \cdot X_1$	$R_1 \leftarrow R_5 \cdot R_3$	$W_2 \leftarrow R_2 \cdot R_5$
Sub.	$\star$	$X_3 \leftarrow R_6 - R_5$	$X_2 \leftarrow R_3 - R_4$
Sub.	$\star$	$R_4 \leftarrow R_4 - X_3$	$R_6 \leftarrow R_4 - X_2$
Mult.	$R_6 \leftarrow X_2 \cdot R_4$	$R_3 \leftarrow R_4 \cdot R_2$	$R_4 \leftarrow R_6 \cdot R_1$
Sub.	$R_6 \leftarrow R_6 - R_5$	$Y_3 \leftarrow R_3 - R_1$	$Y_2 \leftarrow R_4 - R_2$

8 multiplications $\rightarrow$ 6 multiplications + 2 squarings

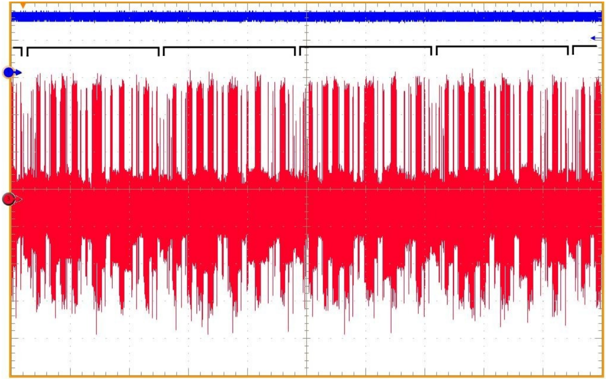
16 additions $\rightarrow$ 6 additions + 4 subtractions

8 negations $\rightarrow$ 0

Implementation

192 bits ECDSA @ 30 MHz (CPU) & 50 MHz (CC)

Original : 35 ms, Improved : 30 ms (- 14.5 %)
Comparable RAM ($\approx$ 500 Bytes) and Code size ($\approx$ 3 KB)



Outline

Introduction
- RSA and Elliptic Curve Cryptography
- Scalar Multiplication Implementation
- Side-Channel Analysis

Improved Atomic Pattern for Scalar Multiplication

Square Always Exponentiation

Horizontal Correlation Analysis

Long-Integer Multiplication Blinding and Shuffling

Collision-Correlation Analysis on AES

Conclusion

Our Contribution

- New atomic algorithms using squarings only
- Immune to attacks distinguishing squarings from multiplications
- Better efficiency than regular ladders
- Exponentiation algorithms for parallelized squarings with best performances to our knowledge

Exponentiation Cost Summary

Algorithm	Cost / bit	$S/M = 1$	$S/M = .8$	# reg
Square & multiply [1,2,3]	$0.5M + 1S$	$1.5M$	$1.3M$	2
Multiply always [2,3]	$1.5M$	$1.5M$	$1.5M$	2
Regular ladders	$1M + 1S$	$2M$	$1.8M$	2

[1] algorithm unprotected towards the SPA
[2] algorithm sensitive to S−M discrimination
[3] possible sliding window optimization

Replacing Multiplications by Squarings

$$x \times y = \frac{(x+y)^2 - x^2 - y^2}{2} \tag{1}$$

$$x \times y = \left(\frac{x+y}{2}\right)^2 - \left(\frac{x-y}{2}\right)^2 \tag{2}$$

Regular Atomic Exponentiation

Square & multiply:

Atomic Multiply always:

Regular Atomic Exponentiation

Square & multiply:

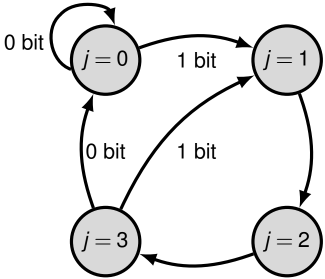
Atomic Multiply always:

Atomic Square always:

Atomic Left-to-Right Algorithm

Input: $m, n, d \in \mathbb{N}$
Output: $m^d \bmod n$

$R_0 \leftarrow 1$; $R_1 \leftarrow m$; $R_2 \leftarrow 1$
$R_3 \leftarrow m^2/2 \bmod n$
$j \leftarrow 0$; $i \leftarrow k-1$
while $i \geq 0$ **do**
    $R_{M_{j,0}} \leftarrow R_{M_{j,1}} + R_{M_{j,2}} \bmod n$
    $R_{M_{j,3}} \leftarrow {R_{M_{j,3}}}^2 \bmod n$
    $R_{M_{j,4}} \leftarrow R_{M_{j,5}}/2 \bmod n$
    $R_{M_{j,6}} \leftarrow R_{M_{j,7}} - R_{M_{j,8}} \bmod n$
    $j \leftarrow d_i(1 + (j \bmod 3))$
    $i \leftarrow i - M_{j,9}$
return R_0

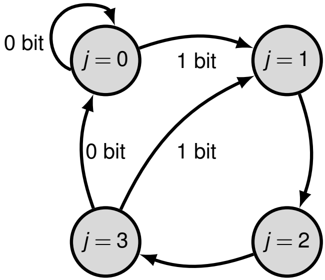


$$M = \begin{pmatrix} 1 & 1 & 1 & 0 & 2 & 1 & 1 & 1 & 2 & 1 \\ 2 & 0 & 1 & 2 & 2 & 2 & 2 & 2 & 3 & 0 \\ 1 & 1 & 3 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 3 & 3 & 3 & 0 & 3 & 3 & 1 & 1 & 3 & 1 \end{pmatrix}$$

Atomic Right-to-Left Algorithm

Input: $m, n, d \in \mathbb{N}$
Output: $m^d \bmod n$

$R_0 \leftarrow m$; $R_1 \leftarrow 1$; $R_2 \leftarrow 1$
$i \leftarrow 0$; $j \leftarrow 0$
while $i \leq k-1$ **do**
  $j \leftarrow d_i(1 + (j \bmod 3))$
  $R_{M_{j,0}} \leftarrow R_{M_{j,1}} + R_0 \bmod n$
  $R_{M_{j,2}} \leftarrow R_{M_{j,3}} / 2 \bmod n$
  $R_{M_{j,4}} \leftarrow R_{M_{j,5}} - R_{M_{j,6}} \bmod n$
  $R_{M_{j,3}} \leftarrow {R_{M_{j,3}}}^2 \bmod n$
  $i \leftarrow i + M_{j,7}$
return R_1



$$M = \begin{pmatrix} 0 & 0 & 2 & 0 & 0 & 0 & 2 & 1 \\ 2 & 1 & 2 & 2 & 1 & 0 & 1 & 0 \\ 0 & 2 & 1 & 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 & 1 \end{pmatrix}$$

Cost Comparison

Algorithm	Cost / bit	$S/M = 1$	$S/M = .8$	# reg
Square & multiply [1,2,3]	$0.5M + 1S$	$1.5M$	$1.3M$	2
Multiply always [2,3]	$1.5M$	$1.5M$	$1.5M$	2
Regular ladder	$1M + 1S$	$2M$	$1.8M$	2
L.-to-r. square always[3]	$2S$	$2M$	**1.6M**	4
R.-to-l. square always[3]	$2S$	$2M$	**1.6M**	3

$\rightarrow$ **11 % speed-up** over Montgomery ladder

[1] algorithm unprotected towards the SPA
[2] algorithm sensitive to S – M discrimination
[3] possible sliding window optimization

AT90SC chip @ 30MHz with AdvX arithmetic coprocessor:

Algorithm	Key len. (b)	Code (B)	RAM (B)	Timing (ms)
Mont. ladder	512	360	128	30
	1024	360	256	200
	2048	360	512	1840
Square Always	512	510	192	28
	1024	510	384	190
	2048	510	768	1740

→ **5 %** practical speed-up obtained in practice

Parallelization

Motivation:

- Many devices are equipped with multi-core processors
- Parallelized Montgomery ladder : $1M$ / bit
- Squarings are independent in equations (1) and (2)

Motivation:

- Many devices are equipped with multi-core processors
- Parallelized Montgomery ladder : $1M$ / bit
- Squarings are independent in equations (1) and (2)

We study how to optimize square always algorithms if **two parallel squarings** are available using space/time trade-offs.

Cost Summary

We demonstrate that the cost of our parallelized algorithm using λ extra registers tends to:

$$\left(1+\frac{1}{4\lambda+2}\right)S$$

Algorithm	General cost	$S/M=1$	$S/M=0.8$
Parallel Montgomery ladder	$1M$	$1M$	$1M$
Parallel square always $\lambda=1$	$7S/6$	$1.17M$	**0.93M**
Parallel square always $\lambda=2$	$11S/10$	$1.10M$	**0.88M**
Parallel square always $\lambda=3$	$15S/14$	$1.07M$	**0.86M**
$\vdots$	$\vdots$	$\vdots$	$\vdots$
Parallel square always $\lambda\to\infty$	$1S$	$1M$	**0.8M**

Outline

Our Contribution

- New differential analysis on exponentiation using a single trace
- Any exponentiation algorithm can be subject to this attack
- Circumvent the exponent blinding countermeasure
- Require the knowledge of underlying modular multiplication implementation

Modular Multiplication Implementation

Schoolbook long-integer multiplication $x \times y$ in base b with $x, y < b^k$

Input: $x = (x_{k-1} x_{k-2} \dots x_0)_b$, $y = (y_{k-1} y_{k-2} \dots y_0)_b$
Output: $x \times y$
Uses: $w = (w_{2k-1} w_{2k-2} \dots w_0)$

$w \leftarrow (00 \dots 0)$
for $i = 0$ **to** $k-1$ **do**
    $c \leftarrow 0$
    for $j = 0$ **to** $k-1$ **do**
        $(uv)_b \leftarrow w_{i+j} + x_i \times y_j + c$
        $w_{i+j} \leftarrow v$
        $c \leftarrow u$
    $w_{i+k} \leftarrow c$
return w

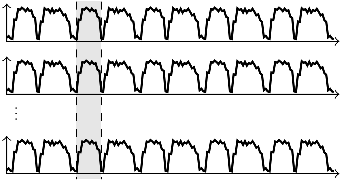
Modular Multiplication Implementation

Rows and columns

$$
\begin{array}{ccccccccccc}
 & & & & & & x_{k-1} & \cdots & x_2 & x_1 & x_0 \\
\times & & & & & & y_{k-1} & \cdots & y_2 & y_1 & y_0 \\
\hline
+ & & & & & & x_0 y_{k-1} & \cdots & x_0 y_2 & x_0 y_1 & x_0 y_0 \\
+ & & & & & x_1 y_{k-1} & x_1 y_{k-2} & \cdots & x_1 y_1 & x_1 y_0 & \\
+ & & & & x_2 y_{k-1} & x_2 y_{k-2} & x_2 y_{k-3} & \cdots & x_2 y_0 & & \\
\vdots & & & & & \iddots & & & & & \\
+ & & x_{k-2} y_{k-1} & \cdots & x_{k-2} y_2 & x_{k-2} y_1 & x_{k-2} y_0 & & & & \\
+ & x_{k-1} y_{k-1} & x_{k-1} y_{k-2} & \cdots & x_{k-1} y_1 & x_{k-1} y_0 & & & & & \\
\hline
w_{2k-1} & w_{2k-2} & w_{2k-3} & & & \cdots & & & w_2 & w_1 & w_0
\end{array}
$$

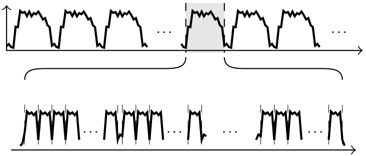
Horizontal Correlation Analysis

Vertical:



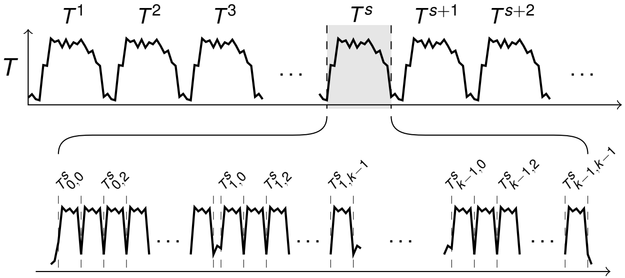
- Uses N segments from different traces.

Horizontal:



- Uses k^2 segments from a single trace.

Horizontal Side-Channel Analysis



We target single-multiplication segments $T^s_{i,j}$ of the s-th modular multiplication inside a single leakage trace T.

Horizontal Correlation Analysis

Considering an atomic multiply-always implementation:

Input: $m, n, d \in \mathbb{N}$
Output: $m^d \bmod n$

$R_0 \leftarrow 1$
$R_1 \leftarrow m$
$i \leftarrow \ell - 1$
$t \leftarrow 0$
while $i \geq 0$ **do**
    $R_0 \leftarrow R_0 \times R_t \bmod n$
    $t \leftarrow t \oplus d_i$
    $i \leftarrow i - 1 + t$
return R_0

Horizontal Correlation Analysis

Considering an atomic multiply-always implementation:

Input: $m, n, d \in \mathbb{N}$
Output: $m^d \bmod n$

$R_0 \leftarrow 1$
$R_1 \leftarrow m$
$i \leftarrow \ell - 1$
$t \leftarrow 0$
while $i \geq 0$ **do**
    $R_0 \leftarrow R_0 \times R_t \bmod n$
    $t \leftarrow t \oplus d_i$
    $i \leftarrow i - 1 + t$
return R_0

- Execute a single RSA signature $m^d \bmod n$ and collect the execution power trace T.

Horizontal Correlation Analysis

Considering an atomic multiply-always implementation:

Input: $m, n, d \in \mathbb{N}$
Output: $m^d \bmod n$

```
R_0 ← 1
R_1 ← m
i ← ℓ − 1
t ← 0
while i ≥ 0 do
    R_0 ← R_0 × R_t mod n
    t ← t ⊕ d_i
    i ← i − 1 + t
return R_0
```

- Execute a single RSA signature $m^d \bmod n$ and collect the execution power trace T.
- Assuming u most significant bits of d are known by the attacker:

$$d = (d_{\ell-1} \dots d_{\ell-u}\ d_{\ell-(u+1)} \dots d_1 d_0)$$

Horizontal Correlation Analysis

Considering an atomic multiply-always implementation:

Input: $m, n, d \in \mathbb{N}$
Output: $m^d \bmod n$

$R_0 \leftarrow 1$
$R_1 \leftarrow m$
$i \leftarrow \ell - 1$
$t \leftarrow 0$
while $i \geq 0$ **do**
    $R_0 \leftarrow R_0 \times R_t \bmod n$
    $t \leftarrow t \oplus d_i$
    $i \leftarrow i - 1 + t$
return R_0

- Execute a single RSA signature $m^d \bmod n$ and collect the execution power trace T.
- Assuming u most significant bits of d are known by the attacker:
  $$d = (d_{\ell-1} \dots d_{\ell-u}\ d_{\ell-(u+1)} \dots d_1 d_0)$$
- Let $R_0^{(u)}$ denote the value of R_0 after processing the u-th bit of d:
  $$R_0^{(u)} = m^{d_{\ell-1} \dots d_{\ell-u}} \bmod n$$

Horizontal Correlation Analysis

Let $v = u + \mathrm{HW}(d_{\ell-1} \ldots d_{\ell-u})$ i.e. u-th bit $\longleftrightarrow$ multiplication T^v

$$
\begin{tikzcd}
 & \boxed{T^{v+1}} & \boxed{T^{v+2}} & \\
 & R_0^{(u)} \times R_0^{(u)} \arrow[r] & {R_0^{(u)}}^2 \times m & \cdots \\
R_0^{(u)} \arrow[ur, "d_{\ell-u-1}=1"] \arrow[dr, "d_{\ell-u-1}=0"'] & & & \\
 & R_0^{(u)} \times R_0^{(u)} \arrow[r, "d_{\ell-u-2}=0{,}1"] & {R_0^{(u)}}^2 \times {R_0^{(u)}}^2 & \cdots
\end{tikzcd}
$$

Horizontal Correlation Analysis

Let $v = u + \mathrm{HW}(d_{\ell-1} \dots d_{\ell-u})$ i.e. u-th bit $\longleftrightarrow$ multiplication T^v

$$\begin{tikzcd}
 & \boxed{T^{v+1}} & & \boxed{T^{v+2}} & \\
 & R_0^{(u)} \times R_0^{(u)} \arrow[rr] & & {R_0^{(u)}}^2 \times m & \cdots \\
R_0^{(u)} \arrow[ur, "d_{\ell-u-1}=1"] \arrow[dr, "d_{\ell-u-1}=0"'] & & & & \\
 & R_0^{(u)} \times R_0^{(u)} \arrow[rr, "d_{\ell-u-2}=0{,}1"] & & {R_0^{(u)}}^2 \times {R_0^{(u)}}^2 & \cdots
\end{tikzcd}$$

- Compute correlation between:
  - trace segments $T_{i,j}^{v+2}$ and values $D_j = m_j$ or
  - trace segments $T_{i,j}^{v+2}$ and values $D_{i,j} = R_{0,i}^{(u)} \times m_j$

Horizontal Correlation Analysis

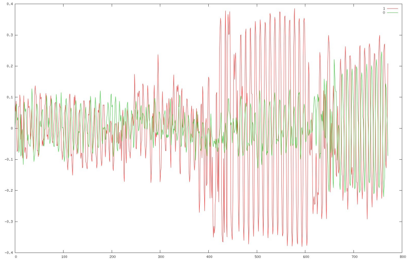
Let $v = u + \mathrm{HW}(d_{\ell-1} \ldots d_{\ell-u})$ i.e. u-th bit $\longleftrightarrow$ multiplication T^v

$$
\begin{tikzcd}
 & \boxed{T^{v+1}} & \boxed{T^{v+2}} & \\
 & R_0^{(u)} \times R_0^{(u)} \arrow[r] & {R_0^{(u)}}^2 \times m & \cdots \\
R_0^{(u)} \arrow[ur, "d_{\ell-u-1}=1"] \arrow[dr, "d_{\ell-u-1}=0"'] & & & \\
 & R_0^{(u)} \times R_0^{(u)} \arrow[r, "d_{\ell-u-2}=0{,}1"] & {R_0^{(u)}}^2 \times {R_0^{(u)}}^2 & \cdots
\end{tikzcd}
$$

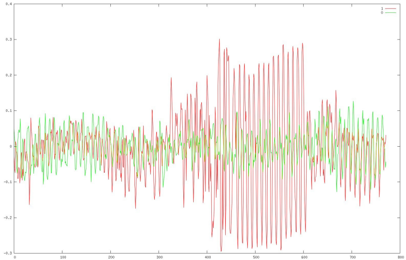
- Compute correlation between:
  - trace segments $T_{i,j}^{v+2}$ and values $D_j = m_j$ or
  - trace segments $T_{i,j}^{v+2}$ and values $D_{i,j} = R_{0,i}^{(u)} \times m_j$
- If correlation peak: $d_{\ell-(u+1)} = 1$, or $d_{\ell-(u+1)} = 0$ otherwise.

Experimental Results

Correlation trace result on series of traces $T_{i,j}^{v+2}$ with $D_j = m_j$



Correlation trace result on series of segments $T_{i,j}^{v+2}$ with $D_{i,j} = R_{0,i}^{(u)} \times m_j$



Outline

Our Contribution

- New countermeasure against differential analysis for RSA and ECC
- Designed to protect from horizontal analysis
- Implemented at the multi-precision multiplication level

Long-Integer Multiplication

Shuffling rows and blinding columns

Let us shuffle the rows of the multiplication:

$$
\begin{array}{ccccccccccc}
 & & & & & & x_{k-1} & \cdots & x_2 & x_1 & x_0 \\
\times & & & & & & y_{k-1} & \cdots & y_2 & y_1 & y_0 \\
\hline
+ & & & & & & x_0 y_{k-1} & \cdots & x_0 y_2 & x_0 y_1 & x_0 y_0 \\
+ & & & & & x_1 y_{k-1} & x_1 y_{k-2} & \cdots & x_1 y_1 & x_1 y_0 & \\
+ & & & & x_2 y_{k-1} & x_2 y_{k-2} & x_2 y_{k-3} & \cdots & x_2 y_0 & & \\
\vdots & & & & & \iddots & & & & & \\
+ & & x_{k-2} y_{k-1} & \cdots & x_{k-2} y_2 & x_{k-2} y_1 & x_{k-2} y_0 & & & & \\
+ & x_{k-1} y_{k-1} & x_{k-1} y_{k-2} & \cdots & x_{k-1} y_1 & x_{k-1} y_0 & & & & & \\
\hline
w_{2k-1} & w_{2k-2} & w_{2k-3} & & & \cdots & & & w_2 & w_1 & w_0
\end{array}
$$

Long-Integer Multiplication

Shuffling rows and blinding columns

Choose at random a permutation α of $(0, 1, \ldots, k-1)$ and compute:

$$(c, w_{\alpha(i)+j})_b = w_{\alpha(i)+j} + x_{\alpha(i)} \times y_j + c$$

$$\begin{array}{cccccccccc}
 & & & & & & x_{k-1} & \ldots & x_2 & x_1 & x_0 \\
\times & & & & & & y_{k-1} & \ldots & y_2 & y_1 & y_0 \\
\hline
+ & & x_{k-2}y_{k-1} & \ldots & x_{k-2}y_2 & x_{k-2}y_1 & x_{k-2}y_0 & & & & \\
+ & & & & & & x_0y_{k-1} & \ldots & x_0y_2 & x_0y_1 & x_0y_0 \\
+ & & & & x_2y_{k-1} & x_2y_{k-2} & x_2y_{k-3} & \ldots & x_2y_0 & & \\
\vdots & & & & & \vdots & & & & & \\
+ & x_{k-1}y_{k-1} & x_{k-1}y_{k-2} & \ldots & x_{k-1}y_1 & x_{k-1}y_0 & & & & & \\
+ & & & & & x_1y_{k-1} & x_1y_{k-2} & \ldots & x_1y_1 & x_1y_0 & \\
\hline
w_{2k-1} & w_{2k-2} & w_{2k-3} & & & \ldots & & & w_2 & w_1 & w_0
\end{array}$$

Long-Integer Multiplication

Shuffling rows and blinding columns

Choose at random a permutation α of $(0, 1, \ldots, k-1)$ and compute:

$$(c, w_{\alpha(i)+j})_b = w_{\alpha(i)+j} + x_{\alpha(i)} \times y_j + c$$

Still necessary to blind columns:

For each row $\alpha(i)$, choose at random a word r,
compute and store $r \times x_{\alpha(i)}$,
blind each single-precision multiplication:

$$(c, w_{\alpha(i)+j})_b = w_{\alpha(i)+j} + x_{\alpha(i)} \times (y_j - r) + r \times x_{\alpha(i)} + c$$

Long-Integer Multiplication

Shuffling rows and blinding columns

Choose at random a permutation α of $(0, 1, \ldots, k-1)$ and compute:

$$(c, w_{\alpha(i)+j})_b = w_{\alpha(i)+j} + x_{\alpha(i)} \times y_j + c$$

Still necessary to blind columns:

For each row $\alpha(i)$, choose at random a word r,
compute and store $r \times x_{\alpha(i)}$,
blind each single-precision multiplication:

$$(c, w_{\alpha(i)+j})_b = w_{\alpha(i)+j} + x_{\alpha(i)} \times (y_j - r) + r \times x_{\alpha(i)} + c$$

➧ Provides $k!$ different sequences of single-precision multiplications.

Long-Integer Multiplication

Shuffling rows and blinding columns

Choose at random a permutation α of $(0, 1, \ldots, k-1)$ and compute:

$$(c, w_{\alpha(i)+j})_b = w_{\alpha(i)+j} + x_{\alpha(i)} \times y_j + c$$

Still necessary to blind columns:

For each row $\alpha(i)$, choose at random a word r,
compute and store $r \times x_{\alpha(i)}$,
blind each single-precision multiplication:

$$(c, w_{\alpha(i)+j})_b = w_{\alpha(i)+j} + x_{\alpha(i)} \times (y_j - r) + r \times x_{\alpha(i)} + c$$

- Provides $k!$ different sequences of single-precision multiplications.
- Requires k extra multiplications and 3 extra words of storage.

Long-Integer Multiplication

Shuffling rows and blinding columns

Choose at random a permutation α of $(0, 1, \dots, k-1)$ and compute:

$$(c, w_{\alpha(i)+j})_b = w_{\alpha(i)+j} + x_{\alpha(i)} \times y_j + c$$

Still necessary to blind columns:

For each row $\alpha(i)$, choose at random a word r,
compute and store $r \times x_{\alpha(i)}$,
blind each single-precision multiplication:

$$(c, w_{\alpha(i)+j})_b = w_{\alpha(i)+j} + x_{\alpha(i)} \times (y_j - r) + r \times x_{\alpha(i)} + c$$

- Provides $k!$ different sequences of single-precision multiplications.
- Requires k extra multiplications and 3 extra words of storage.
- Saves $k+1$ multiplications and $4k-1$ words of storage compared to the full blinding countermeasure.

Long-Integer Multiplication

Shuffling rows and columns

Let us now shuffle the rows and the columns of the multiplication:

Long-Integer Multiplication

Shuffling rows and columns

Let us now shuffle the rows and the columns of the multiplication:

Choose at random two permutations α, β of $(0, 1, \ldots, k-1)$ and compute:

$$(c_{\beta(j)}, w_{\alpha(i)+\beta(j)})_b = w_{\alpha(i)+\beta(j)} + x_{\alpha(i)} \times y_{\beta(j)}$$

Carry propagation is more complicated and requires a k-word array c.

Long-Integer Multiplication

Shuffling rows and columns

Let us now shuffle the rows and the columns of the multiplication:

Choose at random two permutations α, β of $(0, 1, \ldots, k-1)$ and compute:

$$(c_{\beta(j)}, w_{\alpha(i)+\beta(j)})_b = w_{\alpha(i)+\beta(j)} + x_{\alpha(i)} \times y_{\beta(j)}$$

Carry propagation is more complicated and requires a k-word array c.

➧ Provides $(k!)^2$ different sequences of single-precision multiplications.

Let us now shuffle the rows and the columns of the multiplication:

Choose at random two permutations α, β of $(0, 1, \ldots, k-1)$ and compute:

$$(c_{\beta(j)}, w_{\alpha(i)+\beta(j)})_b = w_{\alpha(i)+\beta(j)} + x_{\alpha(i)} \times y_{\beta(j)}$$

Carry propagation is more complicated and requires a k-word array c.

- Provides $(k!)^2$ different sequences of single-precision multiplications.
- Requires no extra multiplication but k extra words of storage.

Long-Integer Multiplication

Shuffling rows and columns

Let us now shuffle the rows and the columns of the multiplication:

Choose at random two permutations α, β of $(0, 1, \ldots, k-1)$ and compute:

$$(c_{\beta(j)}, w_{\alpha(i)+\beta(j)})_b = w_{\alpha(i)+\beta(j)} + x_{\alpha(i)} \times y_{\beta(j)}$$

Carry propagation is more complicated and requires a k-word array c.

- Provides $(k!)^2$ different sequences of single-precision multiplications.
- Requires no extra multiplication but k extra words of storage.
- Saves k multiplications but uses additional storage compared to the previous countermeasure.

Long-Integer Multiplication

For instance, using a 32-bit multiplier:

bit length	$k!$	$(k!)^2$
256	$\approx 2^{15}$	$\approx 2^{30}$
512	$\approx 2^{44}$	$\approx 2^{88}$
1024	$\approx 2^{117}$	$\approx 2^{235}$

Long-Integer Multiplication

For instance, using a 32-bit multiplier:

bit length	$k!$	$(k!)^2$
256	$\approx 2^{15}$	$\approx 2^{30}$
512	$\approx 2^{44}$	$\approx 2^{88}$
1024	$\approx 2^{117}$	$\approx 2^{235}$

➧ Also compatible with interleaved multiplications and reductions.

Long-Integer Multiplication

For instance, using a 32-bit multiplier:

bit length	$k!$	$(k!)^2$
256	$\approx 2^{15}$	$\approx 2^{30}$
512	$\approx 2^{44}$	$\approx 2^{88}$
1024	$\approx 2^{117}$	$\approx 2^{235}$

- Also compatible with interleaved multiplications and reductions.
- Studying the cost of these countermeasures for hardware implementations requires further investigation.

Outline

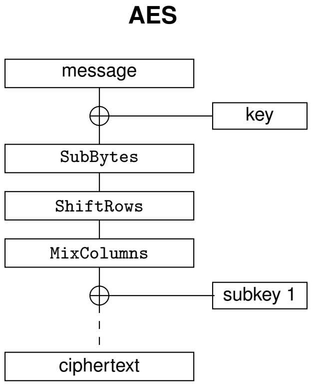
Our Contribution

- Improved collision-correlation techniques on AES defeating some first-order protected implementations
- Need less than 1500 acquisitions in our experiments
- No need to establish a consumption model for correlation

AES Overview

We focus on AES-128:

- message $M = (m_0\, m_1\, \dots\, m_{15})$
- key $K = (k_0\, k_1\, \dots\, k_{15})$
- ciphertext $C = (c_0\, c_1\, \dots\, c_{15})$
- for $i \in [0, 15]$ we denote $x_i = m_i \oplus k_i$

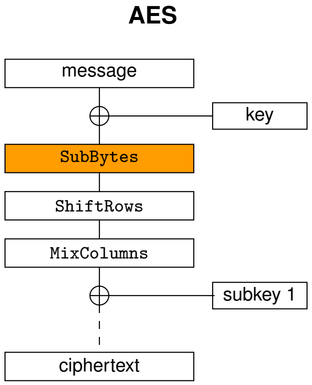


AES Overview

We focus on AES-128:

- message $M = (m_0\, m_1\, \ldots\, m_{15})$
- key $K = (k_0\, k_1\, \ldots\, k_{15})$
- ciphertext $C = (c_0\, c_1\, \ldots\, c_{15})$
- for $i \in [0, 15]$ we denote $x_i = m_i \oplus k_i$

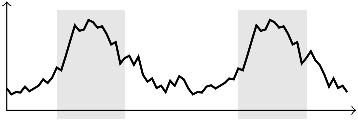
Our attack targets the first round `SubBytes` function



Principle

Detect internal collisions between data processed in blinded S-Boxes in the first AES round:

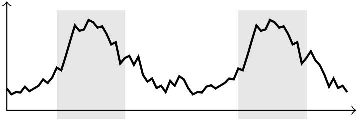
$$\mathit{data}1 \oplus \mathit{mask} \quad = \quad \mathit{data}2 \oplus \mathit{mask}$$



Principle

Detect internal collisions between data processed in blinded S-Boxes in the first AES round:

$$data1 \oplus mask \quad = \quad data2 \oplus mask$$

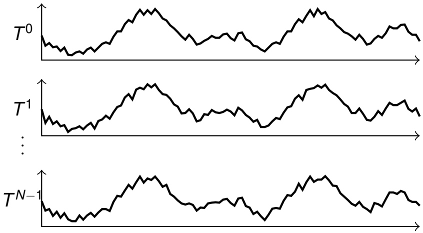


Two protections against first-order attacks are considered:

1. substitution table masking: $S'(x_i \oplus u) = S(x_i) \oplus v$, with $u \neq v$
   same masks u and v for all bytes
2. masked pseudo-inversion in $\mathbb{F}_{2^8}$: $I'(x_i \oplus u_i) = I(x_i) \oplus u_i$, for $0 \leq i \leq 15$
   16 different masks but same input and output masks

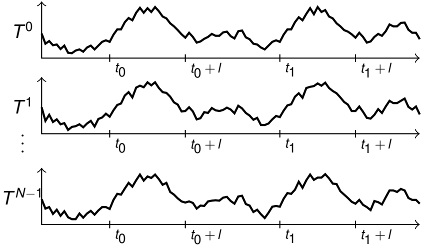
Collision-Correlation Analysis

- Encrypt N times the same message M
- Collect the power traces T^n, $0 \leq n \leq N-1$



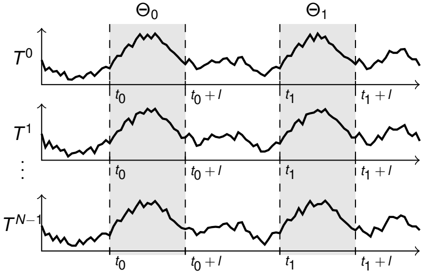
Collision-Correlation Analysis

- Encrypt N times the same message M
- Collect the power traces T^n, $0 \leq n \leq N-1$
- Consider two instructions whose processing starts at times t_0 and t_1
  l points are acquired per instruction processing



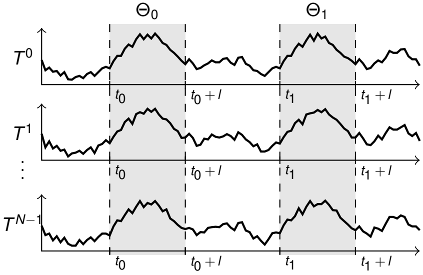
Collision-Correlation Analysis

- Encrypt N times the same message M
- Collect the power traces T^n, $0 \leq n \leq N-1$
- Consider two instructions whose processing starts at times t_0 and t_1
  l points are acquired per instruction processing
- Construct the two series $\Theta_0 = (T^n_{t_0})_n$ and $\Theta_1 = (T^n_{t_1})_n$ of power consumptions segments



Collision-Correlation Analysis

- Encrypt N times the same message M
- Collect the power traces T^n, $0 \leq n \leq N-1$
- Consider two instructions whose processing starts at times t_0 and t_1
  l points are acquired per instruction processing
- Construct the two series $\Theta_0 = (T^n_{t_0})_n$ and $\Theta_1 = (T^n_{t_1})_n$ of power consumptions segments

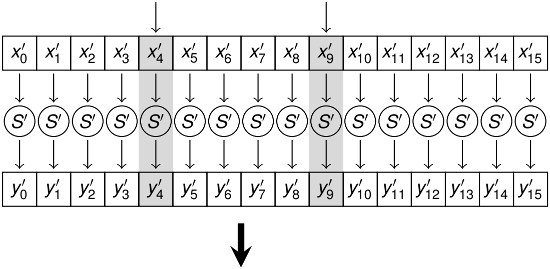


- Apply a statistical treatment to (Θ_0, Θ_1) to identify if same data was involved in $T^n_{t_0}$ and $T^n_{t_1}$
- We choose the Pearson correlation factor $$\hat{\rho}_{\Theta_0,\Theta_1}(t) = \frac{\mathrm{cov}(\Theta_0(t), \Theta_1(t))}{\sigma_{\Theta_0(t)}\sigma_{\Theta_1(t)}}$$

First Attack Description (1)

Principle: detect when two `SubBytes` inputs (and outputs) are equal in first AES round

$$m_4 \oplus k_4 \oplus u \quad = \quad m_9 \oplus k_9 \oplus u$$



$$k_4 \oplus k_9 \quad = \quad m_4 \oplus m_9$$

Result: provide a relation between two key bytes

First Attack Description (2)

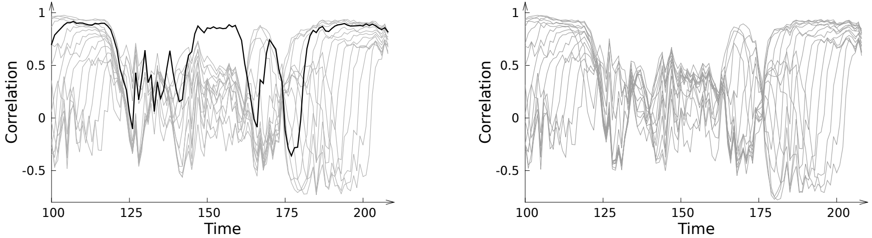
- Encrypt N times the same message M and collect the N traces of first AES round
- For the 120 possible pairs (i_1, i_2) compute $\hat{\rho}_{\Theta_{i_1},\Theta_{i_2}}(t)$
- When a correlation peak appears a relation between k_{i_1} and k_{i_2} is found
- Repeat for several random messages M until enough relations are found

First Attack Description (2)

- Encrypt N times the same message M and collect the N traces of first AES round
- For the 120 possible pairs (i_1, i_2) compute $\hat{\rho}_{\Theta_{i_1}, \Theta_{i_2}}(t)$
- When a correlation peak appears a relation between k_{i_1} and k_{i_2} is found
- Repeat for several random messages M until enough relations are found

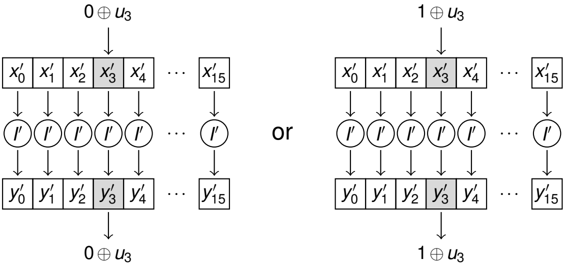
➧On average 59 messages are needed
Total number of traces $= 59 \times N$

Correlation traces obtained on real traces for $N = 25$



Total number of acquisitions : $25 \times 59 \approx 1500$

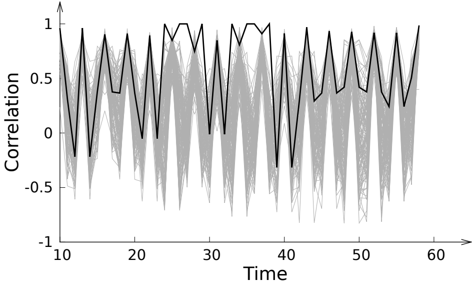
Previous attack cannot be applied to masked inversion if masks are different for each byte



Collision between input and output reveals one key byte except one bit:

$$k_i = m_i \qquad \text{or} \qquad k_i = m_i \oplus 1$$

Correlation traces obtained on simulated traces for the pseudo-inversion of the first byte in $GF(2^8)$ with $N = 16$



Outline

Conclusion

Concrete results of this thesis:

- 4 publications in international conferences (CHES, INDOCRYPT, CARDIS, ICICS)
- 4 patent registrations

Personal benefits:

- Research with industrial constraints is motivating
- Both implementation and side-channel analysis covered in this research
- Both high and low-level implementation studied
- Both public and private-key cryptography investigated