Fault Attacks on Pairing-Based Cryptography

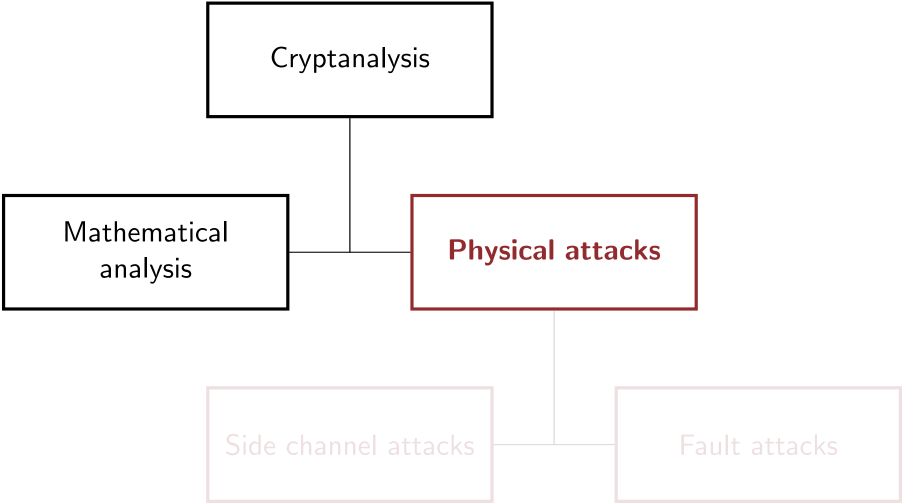
Juliane Krämer

CDC
Technische Universität Darmstadt
Germany

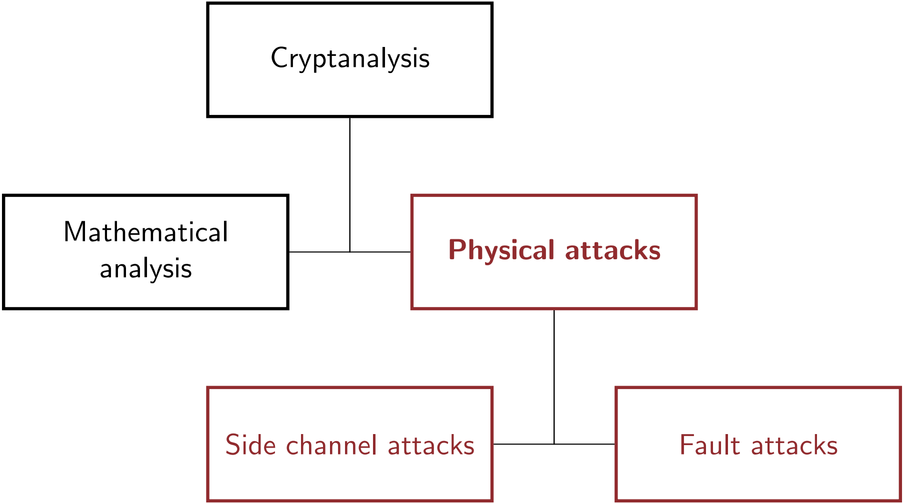
19th Workshop on
Elliptic Curve Cryptography

September 30, 2015

Physical Attacks



Physical Attacks



Bellcore Attack against RSA-CRT (1997) 1/2

- secret primes p, q
- modulus $N = p \cdot q$
- secret key d
- $d_p := d \mod (p-1)$ and $d_q := d \mod (q-1)$

Require: message $m \in \mathbb{Z}_{\mathbb{N}}$
Ensure: $s \equiv m^d \mod N$

1: $S_p := m^{d_p} \mod p$
2: $S_q := m^{d_q} \mod q$
3: $s := CRT(S_p, S_q)$

$$CRT(S_p, S_q) = S_p + p \cdot (p^{-1} \mod q) \cdot (S_q - S_p) \mod N$$

Bellcore Attack against RSA-CRT (1997) 1/2

- secret primes p, q
- modulus $N = p \cdot q$
- secret key d
- $d_p := d \mod (p-1)$ and $d_q := d \mod (q-1)$

Require: message $m \in \mathbb{Z}_{\mathbb{N}}$
Ensure: $s \equiv m^d \mod N$

1: $S_p := m^{d_p} \mod p$
2: $S_q := m^{d_q} \mod q$
3: $s := CRT(S_p, S_q)$

$$CRT(S_p, S_q) = S_p + p \cdot (p^{-1} \mod q) \cdot (S_q - S_p) \mod N$$

Bellcore Attack against RSA-CRT (1997) 2/2

- calculation of the correct value s, using S_p, S_q
- calculation of a faulty value $\tilde{s}$
  - correct computation of S_q
  - fault injection during computation of $S_p \rightarrow \tilde{S}_p$
  - requirement: $\tilde{S}_p \not\equiv S_p \mod p$

$$s \equiv S_p \mod p \qquad s \equiv S_q \mod q$$

$$\tilde{s} \equiv \tilde{S}_p \mod p \qquad \tilde{s} \equiv S_q \mod q$$

$$p \nmid (s - \tilde{s}) \qquad q \mid (s - \tilde{s})$$

$$=> q = \gcd(s - \tilde{s}, N)$$

Bellcore Attack against RSA-CRT (1997) 2/2

- calculation of the correct value s, using S_p, S_q
- calculation of a faulty value $\tilde{s}$
  - correct computation of S_q
  - fault injection during computation of $S_p \rightarrow \tilde{S}_p$
  - requirement: $\tilde{S}_p \not\equiv S_p \mod p$

$$s \equiv S_p \mod p \qquad s \equiv S_q \mod q$$
$$\tilde{s} \equiv \tilde{S}_p \mod p \qquad \tilde{s} \equiv S_q \mod q$$

$$p \nmid (s - \tilde{s}) \qquad q \mid (s - \tilde{s})$$

$$=> q = \gcd(s - \tilde{s}, N)$$

Bellcore Attack against RSA-CRT (1997) 2/2

- calculation of the correct value s, using S_p, S_q
- calculation of a faulty value $\tilde{s}$
  - correct computation of S_q
  - fault injection during computation of $S_p \rightarrow \tilde{S}_p$
  - requirement: $\tilde{S}_p \not\equiv S_p \mod p$

$$s \equiv S_p \mod p \qquad s \equiv S_q \mod q$$

$$\tilde{s} \equiv \tilde{S}_p \mod p \qquad \tilde{s} \equiv S_q \mod q$$

$$p \nmid (s - \tilde{s}) \qquad q \mid (s - \tilde{s})$$

$$=> q = \gcd(s - \tilde{s}, N)$$

Bellcore Attack against RSA-CRT (1997) 2/2

- calculation of the correct value s, using S_p, S_q
- calculation of a faulty value $\tilde{s}$
  - correct computation of S_q
  - fault injection during computation of $S_p \rightarrow \tilde{S}_p$
  - requirement: $\tilde{S}_p \not\equiv S_p \mod p$

$$s \equiv S_p \mod p \qquad s \equiv S_q \mod q$$

$$\tilde{s} \equiv \tilde{S}_p \mod p \qquad \tilde{s} \equiv S_q \mod q$$

$$p \nmid (s - \tilde{s}) \qquad q \mid (s - \tilde{s})$$

$$=> q = \gcd(s - \tilde{s}, N)$$

Bellcore Attack against RSA-CRT (1997) 2/2

- calculation of the correct value s, using S_p, S_q
- calculation of a faulty value $\tilde{s}$
  - correct computation of S_q
  - fault injection during computation of $S_p \rightarrow \tilde{S}_p$
  - requirement: $\tilde{S}_p \not\equiv S_p \mod p$

$$s \equiv S_p \mod p \qquad s \equiv S_q \mod q$$

$$\tilde{s} \equiv \tilde{S}_p \mod p \qquad \tilde{s} \equiv S_q \mod q$$

$$p \not| (s - \tilde{s}) \qquad q \mid (s - \tilde{s})$$

$$=> q = \gcd(s - \tilde{s}, N)$$

First fault attack on ECC

First fault attack on ECC in 2000 (Biehl, Meyer, Müller [BMM00])

Idea for DFA on RSA extended to ECC

Idea:

- Exploit that the curve coefficient a_6 is not used during point addition

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

- Modify the coordinates of a point
- New point will not be on the original curve
- New curve might be cryptographically less secure
- ECDLP might be easier to solve on that curve

First fault attack on ECC

First fault attack on ECC in 2000 (Biehl, Meyer, Müller [BMM00])

Idea for DFA on RSA extended to ECC

Idea:

- Exploit that the curve coefficient a_6 is not used during point addition

$$y^2 + a_1 xy + a_3 y = x^3 + a_2 x^2 + a_4 x + a_6$$

- Modify the coordinates of a point
- New point will not be on the original curve
- New curve might be cryptographically less secure
- ECDLP might be easier to solve on that curve

First fault attack on ECC

Different types of attacks:

- modified base point
  - no check if input point is on the curve
  - one-bit register fault at the beginning of the multiplication process $d \cdot P$
- modified intermediate point
  - register faults during double-and-add algorithm
- applicable to El Gamal decryption and ECDSA
- software simulation
- countermeasure:
  check if input and output points are on the original curve

First fault attack on ECC

Different types of attacks:

- modified base point
  - no check if input point is on the curve
  - one-bit register fault at the beginning of the multiplication process $d \cdot P$
- modified intermediate point
  - register faults during double-and-add algorithm

- applicable to El Gamal decryption and ECDSA
- software simulation
- countermeasure:
  check if input and output points are on the original curve

First fault attack on ECC

Different types of attacks:

- modified base point
    - no check if input point is on the curve
    - one-bit register fault at the beginning of the multiplication process $d \cdot P$
- modified intermediate point
    - register faults during double-and-add algorithm

- applicable to El Gamal decryption and ECDSA
- software simulation
- countermeasure:
  check if input and output points are on the original curve

This talk:

Fault Attacks on Pairings

Preliminary questions:

1. What exactly are pairings?
2. Why are pairings interesting?
3. What is the difference between PBC and ECC?

What exactly are pairings?

Pairing

Elliptic curve E over finite field $\mathbb{F}_q$.
Finite abelian groups $\mathbb{G}_1, \mathbb{G}_2 \leq E$ and $\mathbb{G}_T \leq \mathbb{F}_{q^k}^*$, with k the embedding degree.

A pairing is an efficiently computable, non-degenerate bilinear map

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T.$$

What exactly are pairings?

A pairing is an efficiently computable, non-degenerate bilinear map

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T.$$

Computed in two steps:

1. Evaluation of the Miller function (computation of a for loop)
2. Final Exponentiation

$$e(P, Q) = f_{n,P}(Q)^z$$

Why are pairings interesting?

Identity-based cryptography

- Identity-based cryptography first presented in 1984
- No satisfying realization until 2001
- *Identity-based encryption from the weil pairing*
  by D. Boneh and M. K. Franklin [BF01]
- IBC used for wireless sensor networks

What is the difference between PBC and ECC?

Security of ECC is based on ECDLP.

Elliptic Curve Discrete Logarithm Problem

Elliptic curve E over finite field $\mathbb{F}_q$, $P, Q \in E$ with $Q = n \cdot P$ for $n \in \mathbb{Z}$.

The ECDLP is, given P and Q, to find n.

The difference between ECC and PBC

ECC: ECDLP

$$\text{Given } P \text{ and } Q = n \cdot P, \text{ find } n.$$

Pairings: Pairing Inversion

$$\text{Given } e(P, Q), \text{ find } Q$$

Fault attacks on ECC are not directly applicable to PBC.

The difference between ECC and PBC

ECC: ECDLP

Given P and $Q = n \cdot P$, find n.

Pairings: Pairing Inversion

Given $e(P, Q)$, find Q

Fault attacks on ECC are not directly applicable to PBC.

The difference between ECC and PBC

ECC: ECDLP

$$\text{Given } P \text{ and } Q = n \cdot P, \text{ find } n.$$

Pairings: Pairing Inversion

$$\text{Given } e(P, Q), \text{ find } Q$$

Fault attacks on ECC are not directly applicable to PBC.

One of the input points is secret
$\Rightarrow$ cryptanalysis has to invert two functions

Exponentiation Inversion

Given the output of the pairing as well as $P \in \mathbb{G}_1$ and the final exponent z, find the correct preimage of the final exponentiation, i.e., the field element $f_{n,P}(Q)$.

Miller Inversion

Given n, $P \in \mathbb{G}_1$, and a field element $f_{n,P}(Q)$, find the correct input $Q \in \mathbb{G}_2$.

Related Work (1/3)

First fault attack on PBC (Page, Vercauteren, 2004 [PV04]).

- Reduced Tate pairing
- Duursma-Lee algorithm
- Single fault
- Modified Miller bound n

Related Work (1/3)

Idea [PV04]:

1. Isolate single factor of the Duursma-Lee computation
2. Compute secret point

$$e_n(P,Q) \to e_{n\pm 1}(P,Q)$$

Need two computations whose loop bounds differ exactly by one

$$e_n(P,Q) \to e_{n+r}(P,Q), e_{n+r\pm 1}(P,Q)$$

Repeated fault induction

Assumption:

Values r and $r \pm 1$ can be determined via timing/ power analysis

Related Work (1/3)

Idea [PV04]:

1. Isolate single factor of the Duursma-Lee computation
2. Compute secret point

$$e_n(P, Q) \rightarrow e_{n\pm 1}(P, Q)$$

Need two computations whose loop bounds differ exactly by one

$$e_n(P, Q) \rightarrow e_{n+r}(P, Q), e_{n+r\pm 1}(P, Q)$$

Repeated fault induction

Assumption:

Values r and $r \pm 1$ can be determined via timing/ power analysis

Related Work (1/3)

Idea [PV04]:

1. Isolate single factor of the Duursma-Lee computation
2. Compute secret point

$$e_n(P, Q) \rightarrow e_{n\pm 1}(P, Q)$$

Need two computations whose loop bounds differ exactly by one

$$e_n(P, Q) \rightarrow e_{n+r}(P, Q), e_{n+r\pm 1}(P, Q)$$

Repeated fault induction

Assumption:

Values r and $r \pm 1$ can be determined via timing/ power analysis

Related Work (1/3)

Idea [PV04]:

1. Isolate single factor of the Duursma-Lee computation
2. Compute secret point

$$e_n(P, Q) \rightarrow e_{n\pm 1}(P, Q)$$

Need two computations whose loop bounds differ exactly by one

$$e_n(P, Q) \rightarrow e_{n+r}(P, Q), e_{n+r\pm 1}(P, Q)$$

Repeated fault induction

Assumption:

Values r and $r \pm 1$ can be determined via timing/ power analysis

Related Work (1/3)

Idea [PV04]:

1. Isolate single factor of the Duursma-Lee computation
2. Compute secret point

$$e_n(P, Q) \rightarrow e_{n\pm 1}(P, Q)$$

Need two computations whose loop bounds differ exactly by one

$$e_n(P, Q) \rightarrow e_{n+r}(P, Q), e_{n+r\pm 1}(P, Q)$$

Repeated fault induction

Assumption:
Values r and $r \pm 1$ can be determined via timing/ power analysis

Related Work (2/3)

Importance of the final exponentiation when considering fault attacks on pairings (Whelan, Scott 2007 [WS07])

Data corruption and sign change faults on different types of pairings

Different data is corrupted:

- point P (or intermediate point during computation of $r \cdot P$)
- point Q (or intermediate point during computation of $r \cdot Q$)
- Miller variable

Assumes that the correct timing can be determined via SPA

Related Work (2/3)

Importance of the final exponentiation when considering fault attacks on pairings (Whelan, Scott 2007 [WS07])

Data corruption and sign change faults on different types of pairings

Different data is corrupted:

- point P (or intermediate point during computation of $r \cdot P$)
- point Q (or intermediate point during computation of $r \cdot Q$)
- Miller variable

Assumes that the correct timing can be determined via SPA

Related Work (2/3)

Importance of the final exponentiation when considering fault attacks on pairings (Whelan, Scott 2007 [WS07])

Data corruption and sign change faults on different types of pairings

Different data is corrupted:

- point P (or intermediate point during computation of $r \cdot P$)
- point Q (or intermediate point during computation of $r \cdot Q$)
- Miller variable

Assumes that the correct timing can be determined via SPA

Related Work (2/3)

Data corruption fault attack against a variant of the η_n pairing

- no final exponentiation is required
- data corruption fault in the last iteration of the Miller loop
- system of linear equations

Sign change fault attack on the Weil pairing

- no or a simple final exponentiation
- sign change of y_Q
- reduces to solving a cubic equation

Tate pairing: not vulnerable due to its more complex final exponentiation

„… pairings with either no or a straightforward final exponentiation are less secure than pairings with a more complex final exponentiation when considering such fault attacks“

Related Work (2/3)

Data corruption fault attack against a variant of the η_n pairing

- no final exponentiation is required
- data corruption fault in the last iteration of the Miller loop
- system of linear equations

Sign change fault attack on the Weil pairing

- no or a simple final exponentiation
- sign change of y_Q
- reduces to solving a cubic equation

Tate pairing: not vulnerable due to its more complex final exponentiation

„… pairings with either no or a straightforward final exponentiation are less secure than pairings with a more complex final exponentiation when considering such fault attacks“

Related Work (2/3)

Data corruption fault attack against a variant of the η_n pairing

- no final exponentiation is required
- data corruption fault in the last iteration of the Miller loop
- system of linear equations

Sign change fault attack on the Weil pairing

- no or a simple final exponentiation
- sign change of y_Q
- reduces to solving a cubic equation

Tate pairing: not vulnerable due to its more complex final exponentiation

„... pairings with either no or a straightforward final exponentiation are less secure than pairings with a more complex final exponentiation when considering such fault attacks“

Related Work (3/3)

Targeting the final exponentiation of Tate pairings (Lashermes, Fournier, Goubin, 2013 [LFG13])

$$
\begin{aligned}
t_r : E\left(\mathbb{F}_p\right)[r] \times E\left(\mathbb{F}_{p^k}\right)/rE\left(\mathbb{F}_{p^k}\right) &\to \mu_r \subset \mathbb{F}_{p^k}^* \\
(P, Q) &\mapsto t_r(P, Q) = t(P, Q)^{(p^k-1)/r}.
\end{aligned}
$$

- Recover input to the final exponentiation
- Attack targets an optimized pairing implementation

  Final exponentiation: $t(P, Q)^{\frac{p^k-1}{r}}$ with $\frac{p^k-1}{r} = (p^d - 1) \cdot \frac{p^d+1}{\Phi_k(p)} \cdot \frac{\Phi_k(p)}{r}$
- Needs at least three faulty computations

Related Work (3/3)

$$f_1 = f^{p^d-1}, \quad f_2 = f_1^{\frac{p^d+1}{\Phi_k(p)}}, \quad f_3 = f_2^{\frac{\Phi_k(p)}{r}}$$

Repeated single faults:

1. created on f_1
   $\rightarrow$ find f_1 (with the help of an error-free computation)
2. created during the inversion in the first easy exponentiation
   $\rightarrow$ find a candidate for f
3. created during the inversion in the first easy exponentiation
   $\rightarrow$ find f

Open question: How to reveal the secret input to the pairing?

Related Work (3/3)

$$f_1 = f^{p^d-1}, \quad f_2 = f_1^{\frac{p^d+1}{\Phi_k(p)}}, \quad f_3 = f_2^{\frac{\Phi_k(p)}{r}}$$

Repeated single faults:

1. created on f_1
   $\rightarrow$ find f_1 (with the help of an error-free computation)
2. created during the inversion in the first easy exponentiation
   $\rightarrow$ find a candidate for f
3. created during the inversion in the first easy exponentiation
   $\rightarrow$ find f

Open question: How to reveal the secret input to the pairing?

Related Work (3/3)

$$f_1 = f^{p^d - 1}, \quad f_2 = f_1^{\frac{p^d+1}{\Phi_k(p)}}, \quad f_3 = f_2^{\frac{\Phi_k(p)}{r}}$$

Repeated single faults:

1. created on f_1
   $\rightarrow$ find f_1 (with the help of an error-free computation)
2. created during the inversion in the first easy exponentiation
   $\rightarrow$ find a candidate for f
3. created during the inversion in the first easy exponentiation
   $\rightarrow$ find f

Open question: How to reveal the secret input to the pairing?

Until 2014, all fault attacks on pairing computations
were only theoretically described,
but not practically conducted.

Higher-order fault attacks unrealistic?

If the adversary can inject multiple faults [...], then an attack could be launched. This however, is an unrealistic attack scenario. [WS07]

One possibility to achieve this is to consider double faults [...]. The possibility of this attack scheme is yet to be proven [...]. [LFG13]

[...] how to properly override the Final Exponentiation in conjunction with a fault attack on the Miller Algorithm remains an open problem [...]. [LPE$^+$14]

Higher-order fault attacks unrealistic?

If the adversary can inject multiple faults [...], then an attack could be launched. This however, is an unrealistic attack scenario. [WS07]

One possibility to achieve this is to consider double faults [...]. The possibility of this attack scheme is yet to be proven [...]. [LFG13]

[...] how to properly override the Final Exponentiation in conjunction with a fault attack on the Miller Algorithm remains an open problem [...]. [LPE+14]

Higher-order fault attacks unrealistic?

If the adversary can inject multiple faults [...], then an attack could be launched. This however, is an unrealistic attack scenario. [WS07]

One possibility to achieve this is to consider double faults [...]. The possibility of this attack scheme is yet to be proven [...]. [LFG13]

[...] how to properly override the Final Exponentiation in conjunction with a fault attack on the Miller Algorithm remains an open problem [...]. [LPE$^+$14]

Sketch of the first practical fault attack (FDTC '14)

A Practical Second-Order Fault Attack
against a Real-World Pairing Implementation

joint work with J. Blömer, R. Gomes da Silva, P. Günther, and J.-P. Seifert

- eta pairing
- $P, Q \in E(\mathbb{F}_q)$ mit $\mathbb{F}_q = \mathbb{F}_{2^{271}}$
- $n = 2^{(271+1)/2} + 1$
- $z = (q^4 - 1)/\#E(\mathbb{F}_q)$

Relic Library

ATXMega128A1

HW/SW combination also used on WSNs (TinyPBC)

$$\eta_n(P, Q) = f_{\boldsymbol{n},-P}(\psi(Q))^{\boldsymbol{z}}$$

Sketch of the first practical fault attack (FDTC '14)

A Practical Second-Order Fault Attack
against a Real-World Pairing Implementation

joint work with J. Blömer, R. Gomes da Silva, P. Günther, and J.-P. Seifert

- eta pairing
- $P, Q \in E(\mathbb{F}_q)$ mit $\mathbb{F}_q = \mathbb{F}_{2^{271}}$
- $n = 2^{(271+1)/2} + 1$
- $z = (q^4 - 1)/\#E(\mathbb{F}_q)$

Relic Library

ATXMega128A1

HW/SW combination also used on WSNs (TinyPBC)

$$\eta_n(P, Q) = f_{n,-P}(\psi(Q))^z$$

Sketch of the first practical fault attack (FDTC '14)

A Practical Second-Order Fault Attack
against a Real-World Pairing Implementation

joint work with J. Blömer, R. Gomes da Silva, P. Günther, and J.-P. Seifert

- eta pairing
- $P, Q \in E(\mathbb{F}_q)$ mit $\mathbb{F}_q = \mathbb{F}_{2^{271}}$
- $n = 2^{(271+1)/2} + 1$
- $z = (q^4 - 1)/\#E(\mathbb{F}_q)$

Relic Library

ATXMega128A1

HW/SW combination also used on WSNs (TinyPBC)

$$\eta_n(P, Q) = f_{\boldsymbol{n},-P}(\psi(Q))^{\boldsymbol{z}}$$

Sketch of the first practical fault attack (FDTC '14)

A Practical Second-Order Fault Attack
against a Real-World Pairing Implementation

joint work with J. Blömer, R. Gomes da Silva, P. Günther, and J.-P. Seifert

- eta pairing
- $P, Q \in E(\mathbb{F}_q)$ mit $\mathbb{F}_q = \mathbb{F}_{2^{271}}$
- $n = 2^{(271+1)/2} + 1$
- $z = (q^4 - 1)/\#E(\mathbb{F}_q)$

Relic Library

ATXMega128A1

HW/SW combination also used on WSNs (TinyPBC)

$$\eta_n(P, Q) = f_{\boldsymbol{n},-P}(\psi(Q))^{\boldsymbol{z}}$$

Sketch of the first practical fault attack

$$\eta_n(P, Q) = f_{n,-P}(\psi(Q))^z$$

1. Disturbance of Miller function computation
2. Skipping of the final exponentiation

$$\begin{aligned}\alpha &= f_{n',-P}(\psi(Q)) \\ &= l_{[2]P',-P}(\psi(Q)) \cdot g_{P'}(\psi(Q)) \cdot g_{[2^{-1}]P'}(\psi(Q))^2.\end{aligned}$$

Sketch of the first practical fault attack

$$\eta_n(P, Q) = f_{\boldsymbol{n},-P}(\psi(Q))^{\boldsymbol{z}}$$

1. Disturbance of Miller function computation
2. Skipping of the final exponentiation

$$\begin{aligned}\alpha &= f_{\boldsymbol{n'},-P}(\psi(Q)) \\ &= l_{[2]P',-P}(\psi(Q)) \cdot g_{P'}(\psi(Q)) \cdot g_{[2^{-1}]P'}(\psi(Q))^2.\end{aligned}$$

Sketch of the first practical fault attack

$$\eta_n(P, Q) = f_{\boldsymbol{n},-P}(\psi(Q))^{\boldsymbol{z}}$$

1. Disturbance of Miller function computation
2. Skipping of the final exponentiation

$$\begin{aligned}\alpha &= f_{\boldsymbol{n'},-P}(\psi(Q)) \\ &= l_{[2]P',-P}(\psi(Q)) \cdot g_{P'}(\psi(Q)) \cdot g_{[2^{-1}]P'}(\psi(Q))^2.\end{aligned}$$

Disturbance of Miller function computation

```
call fb4_mul_dxs
.LVL43:
subi r16, 1
sbc r17, __zero_reg__
.loc 1 247 0 discriminator 2
breq .+2
rjmp .L2
.LBE2:
.loc 1 486 0
subi r28, 36
sbci r29, -2
out __SP_L__, r28
out __SP_H__, r29
pop r29
```

Disturbance of Miller function computation

```
call fb4_mul_dxs
.LVL43:
subi r16, 1
sbc r17, __zero_reg__
.loc 1 247 0 discriminator 2
breq .+2
rjmp .L2
.LBE2:
.loc 1 486 0
subi r28, 36
sbci r29, -2
out __SP_L__, r28
out __SP_H__, r29
pop r29
```

Disturbance of Miller function computation

```
call fb4_mul_dxs
.LVL43:
subi r16, 1
sbc r17, __zero_reg__
.loc 1 247 0 discriminator 2
breq .+2

.LBE2:
.loc 1 486 0
subi r28, 36
sbci r29, -2
out __SP_L__, r28
out __SP_H__, r29
pop r29
```

Effect: Instruction Skips

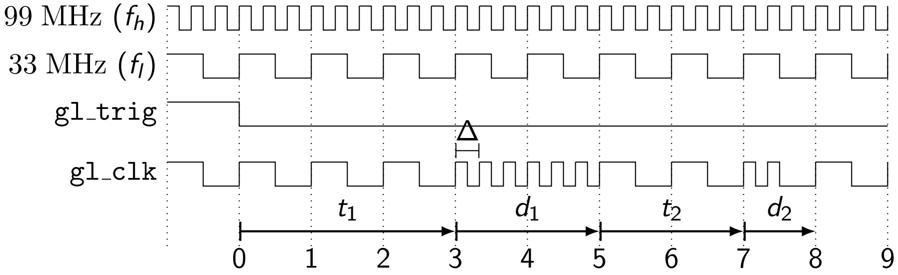
Mechanism: CPU Clock Glitching

building on the results of Balasch, Gierlichs, Verbauwhede, 2011 [BGV11]

Effect: Instruction Skips
Mechanism: CPU Clock Glitching

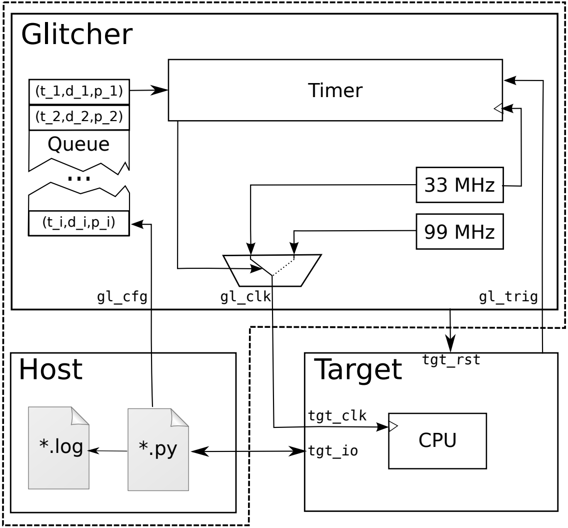
building on the results of Balasch, Gierlichs, Verbauwhede, 2011 [BGV11]

Clock Glitching

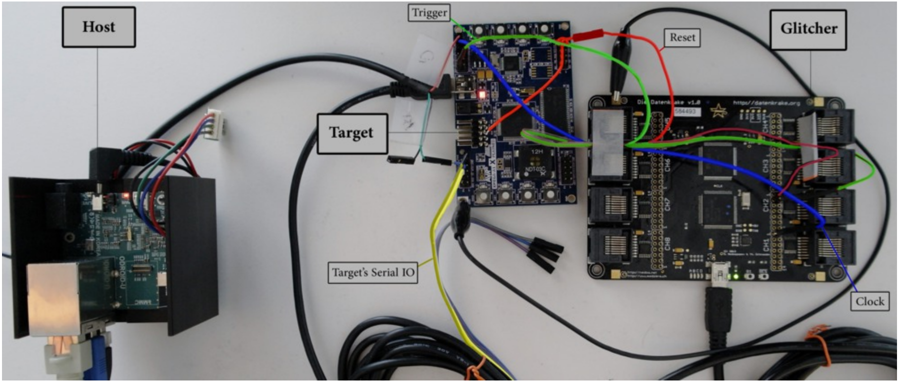


t_i = timing of i-th glitch
d_i = duration of i-th glitch

Block Diagram of the Setup



The Setup



Target Device: ATXMega128A1 (Atmel AVR family)

Approach

Two phases:

- Profiling Phase
  - Determination of good parameters for the glitches
- Target Phase
  - Fault attack
  - Automatic detection if both faults have been successful
  - Computation of secret input point

Profiling Phase for First Fault

t_1 in instruction cycles	occurrence	in %
422,780	1	< 0.01
424,515	1	< 0.01
424,941	1	< 0.01
427,731	1	< 0.01
431,069	1	< 0.01
581,804	3	0.01
581,903	28	0.08
582,001	7	0.02
582,002	590	1.66
582,100	30	0.08
582,101	1,763	4.95
582,111	1	< 0.01
582,199	297	0.83
582,200	32,890	92.35

Target Phase

for each value of t_1:

- $d_1 \in \{3, 5\}$
- $d_2 \leq 5$
- $t_2 \in \{26, \ldots, 30\}$
- 2 values for each p_1 and p_2
- 10 repetitions

$2000 = 2 \cdot 5 \cdot (30 - 25) \cdot 2 \cdot 2 \cdot 10$ tests for each value of t_1

$< 7,5$ seconds per test $\implies$ > 10.000 tests per day

Mathematical Analysis

- Algebraic model of the secret point Q

$$\begin{aligned} f_P(x_Q, y_Q) &:= f_{n',-P}(\psi(x_Q, y_Q)) - \alpha \\ &= l_{[2]P',-P}(\psi(Q)) \cdot g_{P'}(\psi(Q)) \cdot g_{[2^{-1}]P'}(\psi(Q))^2 - \alpha. \end{aligned}$$

- Computation of candidates Q' for Q

$$V_Q = V\left(f_P^{(1)}, \ldots, f_P^{(4)}\right) \cap E$$

- Checking the candidates $Q' \in V_Q$

$$\eta_n(P, Q') \stackrel{?}{=} \eta_n(P, Q)$$

Results

- First practical fault attack on pairings
- System for several independent instruction skips
- Applicable to a wide range of pairings
- Many instructions are potential targets for a similar attack

- Same system recently used to transfer the DLP from a cryptographically strong elliptic curve to a weak singular curve (Günther, Blömer, FDTC 2015 [GB15])
- Practically performed against BLS short signature scheme

Results

- First practical fault attack on pairings
- System for several independent instruction skips
- Applicable to a wide range of pairings
- Many instructions are potential targets for a similar attack
- Same system recently used to transfer the DLP from a cryptographically strong elliptic curve to a weak singular curve (Günther, Blömer, FDTC 2015 [GB15])
- Practically performed against BLS short signature scheme

Results

- First practical fault attack on pairings
- System for several independent instruction skips
- Applicable to a wide range of pairings
- Many instructions are potential targets for a similar attack
- Same system recently used to transfer the DLP from a cryptographically strong elliptic curve to a weak singular curve (Günther, Blömer, FDTC 2015 [GB15])
- Practically performed against BLS short signature scheme

Countermeasures (1/3)

Hardware Countermeasures

- sensors which detect attempts of glitching
- power down crypto devices when operated outside the specified clock

Countermeasures (2/3)

Software Countermeasures

generic:

- checksums
- redundant computations
  (e.g., compute Miller loop twice and compare the results)

final exponentiation:

- code optimization (to prevent that there is a function call)

cryptographic protocols:

- hash results of pairing computation

Countermeasures (2/3)

Software Countermeasures

generic:

- checksums
- redundant computations
  (e.g., compute Miller loop twice and compare the results)

final exponentiation:

- code optimization (to prevent that there is a function call)

cryptographic protocols:

- hash results of pairing computation

Countermeasures (2/3)

Software Countermeasures

generic:

- checksums
- redundant computations
  (e.g., compute Miller loop twice and compare the results)

final exponentiation:

- code optimization (to prevent that there is a function call)

cryptographic protocols:

- hash results of pairing computation

Countermeasures (3/3)

Software Countermeasures

Miller algorithm:

- ensure that the whole loop is actually computed
- ensure that the result of the computation of the whole loop is actually further used
- blinding the points based on the bilinearity (no randomization based on redundant representation!)
- random delays and dummy operations impede the determination of the timings

Future Work

- other physical faults, e.g., laser
- skipping instructions within the final exponentiation
- consider complete protocols
  - repeated fault attacks with the same input might not be possible
  - result of pairing is not released

Thank you for your attention.

Juliane Krämer: jkraemer@cdc.tu-darmstadt.de

Future Work

- other physical faults, e.g., laser
- skipping instructions within the final exponentiation
- consider complete protocols
  - repeated fault attacks with the same input might not be possible
  - result of pairing is not released

Thank you for your attention.

Juliane Krämer: jkraemer@cdc.tu-darmstadt.de

Future Work

- other physical faults, e.g., laser
- skipping instructions within the final exponentiation
- consider complete protocols
  - repeated fault attacks with the same input might not be possible
  - result of pairing is not released

Thank you for your attention.

Juliane Krämer: jkraemer@cdc.tu-darmstadt.de

Future Work

- other physical faults, e.g., laser
- skipping instructions within the final exponentiation
- consider complete protocols
  - repeated fault attacks with the same input might not be possible
  - result of pairing is not released

Thank you for your attention.

Juliane Krämer: jkraemer@cdc.tu-darmstadt.de

Literatur I

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, volume 2139 of Lecture Notes in Computer Science, pages 213–229. Springer-Verlag, 2001.

[BGOS07] Paulo S. L. M. Barreto, Steven D. Galbraith, Colm O'Eigeartaigh, and Michael Scott. Efficient pairing computation on supersingular Abelian varieties. Des. Codes Cryptogr., 42(3):239–271, 2007.

[BGV11] Josep Balasch, Benedikt Gierlichs, and Ingrid Verbauwhede. An In-depth and Black-box Characterization of the Effects of Clock Glitches on 8-bit MCUs. In Luca Breveglieri, Sylvain Guilley, Israel Koren, David Naccache, and Junko Takahashi, editors, 2011 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC), pages 105–114, 2011.

[BMM00] Ingrid Biehl, Bernd Meyer, and Volker Müller. Differential fault attacks on elliptic curve cryptosystems. In Mihir Bellare, editor, Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, volume 1880 of Lecture Notes in Computer Science, pages 131–146. Springer, 2000.

Literatur II

[GB15] Peter Günther and Johannes Blömer. Singular curve point decompression attack. In Proceedings of Fault Tolerance and Diagnosis in Cryptography (FDTC), 2015. To appear.

[LFG13] Ronan Lashermes, Jacques Fournier, and Louis Goubin. Inverting the Final Exponentiation of Tate Pairings on Ordinary Elliptic Curves Using Faults. In Cryptographic Hardware and Embedded Systems — CHES 2013, volume 8086 of Lecture Notes in Computer Science, pages 365–382. Springer Berlin Heidelberg, 2013.

[LPE+14] Ronan Lashermes, Marie Paindavoine, Nadia El Mrabet, Jacques Fournier, and Louis Goubin. Practical validation of several fault attacks against the miller algorithm. In 2014 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC), 2014.

[PV04] Daniel Page and Frederik Vercauteren. Fault and Side-Channel Attacks on Pairing Based Cryptography. IACR Cryptology ePrint Archive, Report 2004/283, 2004.

[Sco05] Michael Scott. Computing the Tate Pairing. In Topics in Cryptology - CT-RSA 2005, pages 293–304. Springer Berlin Heidelberg, 2005.

Literatur III

[WS07] Claire Whelan and Michael Scott. The Importance of the Final Exponentiation in Pairings When Considering Fault Attacks. In Pairing, volume 4575 of Lecture Notes in Computer Science, pages 225–246. Springer Berlin Heidelberg, 2007.