

Fault Attacks on Public Key Elements: Application to DLP-based Schemes

Chong-Hee Kim, Philippe Bulens,
Christophe Petit and Jean-Jacques Quisquater

Fault Attacks

Fault Attacks

- ▶ Secure Cryptographic protocols are vulnerable to
 - ▶ Side-channel attacks (time, power, e.m. waves)
 - ▶ Fault attacks

Fault Attacks

- ▶ Secure Cryptographic protocols are vulnerable to
 - ▶ Side-channel attacks (time, power, e.m. waves)
 - ▶ Fault attacks
- ▶ Fault attacks: induce faults in order to
 - ▶ Deduce secret values
 - ▶ Make the device accept a wrong signature
 - ▶ ...

Fault Attacks

- ▶ Secure Cryptographic protocols are vulnerable to
 - ▶ Side-channel attacks (time, power, e.m. waves)
 - ▶ Fault attacks
- ▶ Fault attacks: induce faults in order to
 - ▶ Deduce secret values
 - ▶ Make the device accept a wrong signature
 - ▶ ...
- ▶ Fault injections by
 - ▶ Variation of supply voltage, clock frequency, temperature
 - ▶ Use of white light, X-ray, ion beams

Fault Attacks

- ▶ Secure Cryptographic protocols are vulnerable to
 - ▶ Side-channel attacks (time, power, e.m. waves)
 - ▶ Fault attacks
- ▶ Fault attacks: induce faults in order to
 - ▶ Deduce secret values
 - ▶ Make the device accept a wrong signature
 - ▶ ...
- ▶ Fault injections by
 - ▶ Variation of supply voltage, clock frequency, temperature
 - ▶ Use of white light, X-ray, ion beams
- ▶ Targets: CRT-RSA, DES, RSA, ElGamal, Luc and Demytko, ECC, AES, DSA,...

Fault Attacks on Public Key Elements

- ▶ Fault injections target
 - ▶ Secret key elements of symmetric algorithms
 - ▶ Secret key elements of asymmetric algorithms
- ▶ Secret key elements are now protected against FA

Fault Attacks on Public Key Elements

- ▶ Fault injections target
 - ▶ Secret key elements of symmetric algorithms
 - ▶ Secret key elements of asymmetric algorithms
- ▶ Secret key elements are now protected against FA
- ▶ Recently:
 - ▶ Fault injections on the *public* key elements of ECDLP algorithm
 - ▶ Fault injections on the *public* key elements of IFP algorithm

Fault Attacks on Public Key Elements

- ▶ Fault injections target
 - ▶ Secret key elements of symmetric algorithms
 - ▶ Secret key elements of asymmetric algorithms
- ▶ Secret key elements are now protected against FA
- ▶ Recently:
 - ▶ Fault injections on the *public* key elements of ECDLP algorithm
 - ▶ Fault injections on the *public* key elements of IFP algorithm
- ▶ Can this approach be extended to DLP-based algorithms ?

Outline

- ▶ **Introduction**
- ▶ New attacks on DLP-based schemes
 - ▶ A new fault attack on ElGamal
 - ▶ A new fault attack on DSA

Outline

- ▶ **Introduction**
- ▶ New attacks on DLP-based schemes
 - ▶ A new fault attack on ElGamal
 - ▶ A new fault attack on DSA
- ▶ Theoretical analysis (DSA)
- ▶ Simulation results (DSA & ElGamal)

Outline

- ▶ **Introduction**
- ▶ New attacks on DLP-based schemes
 - ▶ A new fault attack on ElGamal
 - ▶ A new fault attack on DSA
- ▶ Theoretical analysis (DSA)
- ▶ Simulation results (DSA & ElGamal)
- ▶ Further work
- ▶ Conclusion

Outline

- ▶ Introduction
- ▶ **New attacks on DLP-based schemes**
 - ▶ A new fault attack on ElGamal
 - ▶ A new fault attack on DSA
- ▶ Theoretical analysis (DSA)
- ▶ Simulation results (DSA & ElGamal)
- ▶ Further work
- ▶ Conclusion

Outline

- ▶ Introduction
- ▶ New attacks on DLP-based schemes
 - ▶ A new fault attack on ElGamal
 - ▶ A new fault attack on DSA
- ▶ Theoretical analysis (DSA)
- ▶ Simulation results (DSA & ElGamal)
- ▶ Further work
- ▶ Conclusion

ElGamal Signature Algorithm

▶ KeyGen:

- ▶ Pick random prime p ; pick random $x, g < p$; $y = g^x \bmod p$
- ▶ Private key is x ; Public key is (y, g, p)

ElGamal Signature Algorithm

▶ KeyGen:

- ▶ Pick random prime p ; pick random $x, g < p$; $y = g^x \bmod p$
- ▶ Private key is x ; Public key is (y, g, p)

▶ Signature

- ▶ Pick random k s.t. $\gcd(k, p - 1) = 1$
- ▶ Compute $u \equiv g^k \bmod p$ and $v \equiv \frac{m - xu}{k} \bmod (p - 1)$
- ▶ Signature is (u, v)

ElGamal Signature Algorithm

▶ KeyGen:

- ▶ Pick random prime p ; pick random $x, g < p$; $y = g^x \bmod p$
- ▶ Private key is x ; Public key is (y, g, p)

▶ Signature

- ▶ Pick random k s.t. $\gcd(k, p - 1) = 1$
- ▶ Compute $u \equiv g^k \bmod p$ and $v \equiv \frac{m-xu}{k} \bmod (p - 1)$
- ▶ Signature is (u, v)

▶ Verification

- ▶ Check that $y^u u^v \equiv g^m \bmod p$

A new Fault Attack on ElGamal

- ▶ Assumption:
 - ▶ Attacker generates transient random faults on p
 - ▶ He knows (or can guess) the resulting p^*

A new Fault Attack on ElGamal

▶ Assumption:

- ▶ Attacker generates transient random faults on p
- ▶ He knows (or can guess) the resulting p^*

▶ If $\gcd(k, p^* - 1) = 1$:

$$u^* \equiv g^k \pmod{p^*} \quad \text{and} \quad v^* \equiv \frac{m-xu^*}{k} \pmod{p^* - 1}$$

A new Fault Attack on ElGamal

- ▶ Assumption:
 - ▶ Attacker generates transient random faults on p
 - ▶ He knows (or can guess) the resulting p^*
- ▶ If $\gcd(k, p^* - 1) = 1$:
 $u^* \equiv g^k \pmod{p^*}$ and $v^* \equiv \frac{m-xu^*}{k} \pmod{p^* - 1}$
- ▶ **Let t s.t. $t|p^*$ and $\varphi(t)|(p^* - 1)$**

$$(u^*)^{v^*} \equiv g^{k \frac{m-xu^*}{k}} \equiv g^{(m-xu^*)} \pmod{t}$$

A new Fault Attack on ElGamal

- ▶ Assumption:
 - ▶ Attacker generates transient random faults on p
 - ▶ He knows (or can guess) the resulting p^*
- ▶ If $\gcd(k, p^* - 1) = 1$:
 $u^* \equiv g^k \bmod p^*$ and $v^* \equiv \frac{m-xu^*}{k} \bmod (p^* - 1)$
- ▶ **Let t s.t. $t|p^*$ and $\varphi(t)|(p^* - 1)$**

$$\begin{aligned}(u^*)^{v^*} &\equiv g^{k \frac{m-xu^*}{k}} \equiv g^{(m-xu^*)} \bmod t \\ \frac{(u^*)^{v^*}}{g^m} &\equiv (g^{-u^*})^x \bmod t\end{aligned}$$

A new Fault Attack on ElGamal

- ▶ So for each t s.t. $t|p^*$ and $\varphi(t)|(p^* - 1)$ we get

$$\beta \equiv \alpha^x \pmod{t}$$

- ▶ The only unknown is x

$$\text{DL}(\alpha, \beta, t) \equiv x \pmod{r}$$

(r : multiplicative order of α modulo t)

A new Fault Attack on ElGamal

- ▶ So for each t s.t. $t|p^*$ and $\varphi(t)|(p^* - 1)$ we get

$$\beta \equiv \alpha^x \pmod{t}$$

- ▶ The only unknown is x

$$\text{DL}(\alpha, \beta, t) \equiv x \pmod{r}$$

(r : multiplicative order of α modulo t)

- ▶ Repeat and use CRT:

$$x = \text{CRT}(x \pmod{r_1}, x \pmod{r_2}, \dots)$$

Outline

- ▶ Introduction
- ▶ **New attacks on DLP-based schemes**
 - ▶ A new fault attack on ElGamal
 - ▶ **A new fault attack on DSA**
- ▶ Theoretical analysis (DSA)
- ▶ Simulation results (DSA & ElGamal)
- ▶ Further work
- ▶ Conclusion

Digital Signature Algorithm (DSA)

▶ KeyGen:

- ▶ Pick $\{p, q, h, g\}$ s.t. p prime, $q|(p-1)$, $g \in \mathbb{Z}_p^*$, $\text{ord}(g) = q$
- ▶ Private key is $x \in \mathbb{Z}_p^*$; Public key is $y = g^x \bmod p$

Digital Signature Algorithm (DSA)

▶ KeyGen:

- ▶ Pick $\{p, q, h, g\}$ s.t. p prime, $q|(p-1)$, $g \in \mathbb{Z}_p^*$, $\text{ord}(g) = q$
- ▶ Private key is $x \in \mathbb{Z}_p^*$; Public key is $y = g^x \bmod p$

▶ Signature

- ▶ Pick random $k < q$
- ▶ Compute $u \equiv (g^k \bmod p) \bmod q$ and $v \equiv \frac{h(m)+xu}{k} \bmod q$
- ▶ Signature is (u, v)

Digital Signature Algorithm (DSA)

▶ KeyGen:

- ▶ Pick $\{p, q, h, g\}$ s.t. p prime, $q|(p-1)$, $g \in \mathbb{Z}_p^*$, $\text{ord}(g) = q$
- ▶ Private key is $x \in \mathbb{Z}_p^*$; Public key is $y = g^x \bmod p$

▶ Signature

- ▶ Pick random $k < q$
- ▶ Compute $u \equiv (g^k \bmod p) \bmod q$ and $v \equiv \frac{h(m)+xu}{k} \bmod q$
- ▶ Signature is (u, v)

▶ Verification

- ▶ Check that $u = (g^{wh(m)} y^{wu} \bmod p) \bmod q$ where $w = v^{-1} \bmod q$

A new Fault Attack on DSA

- ▶ Assumption:
 - ▶ Attacker generates transient random faults on p and q
 - ▶ He knows (or can guess) the resulting p^* and q^*

A new Fault Attack on DSA

- ▶ Assumption:
 - ▶ Attacker generates transient random faults on p and q
 - ▶ He knows (or can guess) the resulting p^* and q^*
- ▶ If $\gcd(k, q^*) = 1$:

$$u^* \equiv (g^k \bmod p^*) \bmod q^*$$

$$v^* \equiv \frac{h(m) - xu^*}{k} \bmod q^*$$

A new Fault Attack on DSA

- ▶ Assumption:
 - ▶ Attacker generates transient random faults on p and q
 - ▶ He knows (or can guess) the resulting p^* and q^*
- ▶ If $\gcd(k, q^*) = 1$:

$$u^* \equiv (g^k \bmod p^*) \bmod q^*$$

$$v^* \equiv \frac{h(m) - xu^*}{k} \bmod q^*$$

- ▶ **Let t s.t. $t|p^*$ and $\varphi(t)|(p^* - 1)$ and $t|q^*$**

$$(u^*)^{v^*} \equiv g^{k \frac{h(m) - xu^*}{k}} \equiv g^{(h(m) - xu^*)} \bmod t$$

A new Fault Attack on DSA

- ▶ Assumption:
 - ▶ Attacker generates transient random faults on p and q
 - ▶ He knows (or can guess) the resulting p^* and q^*
- ▶ If $\gcd(k, q^*) = 1$:

$$u^* \equiv (g^k \bmod p^*) \bmod q^*$$

$$v^* \equiv \frac{h(m) - xu^*}{k} \bmod q^*$$

- ▶ **Let t s.t. $t|p^*$ and $\varphi(t)|(p^* - 1)$ and $t|q^*$**

$$(u^*)^{v^*} \equiv g^{k \frac{h(m) - xu^*}{k}} \equiv g^{(h(m) - xu^*)} \bmod t$$

$$\frac{(u^*)^{v^*}}{g^{h(m)}} \equiv (g^{-u^*})^x \bmod t$$

A new Fault Attack on DSA

- ▶ So for each t s.t. $t|p^*$ and $\varphi(t)|(p^* - 1)$ and $t|q^*$

$$\beta \equiv \alpha^x \pmod{t}$$

- ▶ The only unknown is x

$$\text{DL}(\alpha, \beta, t) \equiv x \pmod{r}$$

(r : multiplicative order of α modulo t)

A new Fault Attack on DSA

- ▶ So for each t s.t. $t|p^*$ and $\varphi(t)|(p^* - 1)$ and $t|q^*$

$$\beta \equiv \alpha^x \pmod{t}$$

- ▶ The only unknown is x

$$\text{DL}(\alpha, \beta, t) \equiv x \pmod{r}$$

(r : multiplicative order of α modulo t)

- ▶ Repeat and use CRT:

$$x = \text{CRT}(x \pmod{r_1}, x \pmod{r_2}, \dots)$$

Outline

- ▶ Introduction
- ▶ New attacks on DLP-based schemes
 - ▶ A new fault attack on ElGamal
 - ▶ A new fault attack on DSA
- ▶ **Theoretical analysis (DSA)**
- ▶ Simulation results (DSA & ElGamal)
- ▶ Further work
- ▶ Conclusion

Theoretical Analysis (DSA)

Theoretical Analysis (DSA)

- ▶ When do we get information on x ?

Theoretical Analysis (DSA)

- ▶ When do we get information on x ?
 - ▶ Only if $\exists t$ such that $t|p^*$ and $\varphi(t)|(p^* - 1)$ and $t|q^*$

Theoretical Analysis (DSA)

- ▶ When do we get information on x ?
 - ▶ Only if $\exists t$ such that $t|p^*$ and $\varphi(t)|(p^* - 1)$ and $t|q^*$
 - ▶ Not when $\gcd(k, q^*) \neq 1$:
the device may stop while computing $v^* \equiv \frac{h(m) - xu^*}{k} \pmod{q^*}$

Theoretical Analysis (DSA)

- ▶ When do we get information on x ?
 - ▶ Only if $\exists t$ such that $t|p^*$ and $\varphi(t)|(p^* - 1)$ and $t|q^*$
 - ▶ **Not** when $\gcd(k, q^*) \neq 1$:
the device may stop while computing $v^* \equiv \frac{h(m) - xu^*}{k} \pmod{q^*}$
- ▶ How much information do we get from $\beta \equiv \alpha^x \pmod{t}$?

Theoretical Analysis (DSA)

- ▶ When do we get information on x ?
 - ▶ Only if $\exists t$ such that $t|p^*$ and $\varphi(t)|(p^* - 1)$ and $t|q^*$
 - ▶ Not when $\gcd(k, q^*) \neq 1$:
the device may stop while computing $v^* \equiv \frac{h(m) - xu^*}{k} \pmod{q^*}$
- ▶ How much information do we get from $\beta \equiv \alpha^x \pmod{t}$?
 - ▶ We recover x modulo the order of α modulo t

Theoretical Analysis (DSA)

- ▶ When do we get information on x ?
 - ▶ Only if $\exists t$ such that $t|p^*$ and $\varphi(t)|(p^* - 1)$ and $t|q^*$
 - ▶ **Not** when $\gcd(k, q^*) \neq 1$:
the device may stop while computing $v^* \equiv \frac{h(m) - xu^*}{k} \pmod{q^*}$
- ▶ How much information do we get from $\beta \equiv \alpha^x \pmod{t}$?
 - ▶ We recover x modulo the **order of α modulo t**
 - ▶ $\alpha \equiv g^{-u^*} \equiv g^{-(g^k \pmod{p^*})} \pmod{q^*} \equiv g^{-g^k \pmod{t}} \pmod{t}$

Theoretical Analysis (DSA)

► Four steps :

1. $\forall t$: $P_t =$ probability that

$$C_t := (t \mid q^*) \wedge (\varphi(t) \mid q^*) \wedge (t \mid p^*) \wedge (\gcd(k, q^*) = 1)$$

Theoretical Analysis (DSA)

► Four steps :

1. $\forall t: P_t =$ probability that

$$C_t := (t \mid q^*) \wedge (\varphi(t) \mid q^*) \wedge (t \mid p^*) \wedge (\gcd(k, q^*) = 1)$$

2. $\forall r, t: P_{r|t}$ probability that x is recovered modulo r if C_t holds

Theoretical Analysis (DSA)

► Four steps :

1. $\forall t: P_t =$ probability that

$$C_t := (t \mid q^*) \wedge (\varphi(t) \mid q^*) \wedge (t \mid p^*) \wedge (\gcd(k, q^*) = 1)$$

2. $\forall r, t: P_{r|t}$ probability that x is recovered modulo r if C_t holds

3. $\forall r: P[r]$ probability to get x modulo r after one fault

Theoretical Analysis (DSA)

► Four steps :

1. $\forall t: P_t =$ probability that

$$C_t := (t \mid q^*) \wedge (\varphi(t) \mid q^*) \wedge (t \mid p^*) \wedge (\gcd(k, q^*) = 1)$$

2. $\forall r, t: P_{r|t}$ probability that x is recovered modulo r if C_t holds

3. $\forall r: P[r]$ probability to get x modulo r after one fault

4. Number of faults required

First step : P_t

$$P_t = \Pr[(t \mid q^*) \wedge (\varphi(t) \mid q^*) \wedge (t \mid p^*) \wedge (\gcd(k, q^*) = 1)]$$

First step : P_t

$$P_t = \Pr[(t \mid q^*) \wedge (\varphi(t) \mid q^*) \wedge (t \mid p^*) \wedge (\gcd(k, q^*) = 1)]$$

► Let $l_t = \text{lcm}(t, \varphi(t))$ and write $q^* = q_1^* l_t + q_2^*$

$$\begin{aligned} P_t &= \Pr[l_t \mid q^*] \Pr[t \mid p^*] \Pr[\gcd(k, q^*) = 1 \text{ when } l_t \mid q^*] \\ &= \Pr[q_2^* = 0] \Pr[t \mid p^*] \Pr[\gcd(k, q_1^* l_t) = 1] \\ &= \Pr[q_2^* = 0] \Pr[t \mid p^*] \Pr[(\gcd(k, q_1^*) = 1) \wedge (\gcd(k, l_t) = 1)] \\ &\approx \frac{1}{l_t} \frac{1}{t} \Pr[\gcd(k, q_1^*) = 1] \Pr[\gcd(k, l_t) = 1] \\ &\approx \frac{1}{l_t} \frac{1}{t} 0.6 \frac{\varphi(l_t)}{l_t} = 0.6 \frac{\varphi(l_t)}{t l_t^2} \end{aligned}$$

First step : P_t

$$P_t = \Pr[(t \mid q^*) \wedge (\varphi(t) \mid q^*) \wedge (t \mid p^*) \wedge (\gcd(k, q^*) = 1)]$$

- ▶ Let $l_t = \text{lcm}(t, \varphi(t))$ and write $q^* = q_1^* l_t + q_2^*$

$$\begin{aligned} P_t &= \Pr[l_t \mid q^*] \Pr[t \mid p^*] \Pr[\gcd(k, q^*) = 1 \text{ when } l_t \mid q^*] \\ &= \Pr[q_2^* = 0] \Pr[t \mid p^*] \Pr[\gcd(k, q_1^* l_t) = 1] \\ &= \Pr[q_2^* = 0] \Pr[t \mid p^*] \Pr[(\gcd(k, q_1^*) = 1) \wedge (\gcd(k, l_t) = 1)] \\ &\approx \frac{1}{l_t} \frac{1}{t} \Pr[\gcd(k, q_1^*) = 1] \Pr[\gcd(k, l_t) = 1] \\ &\approx \frac{1}{l_t} \frac{1}{t} 0.6 \frac{\varphi(l_t)}{l_t} = 0.6 \frac{\varphi(l_t)}{t l_t^2} \end{aligned}$$

- ▶ $P_t \approx O(t^{-e})$ with $2 \leq e \leq 3$

Second step : $P_{r|t}$

$P_{r|t}$ probability that x is recovered modulo r if C_t holds

Second step : $P_{r|t}$

$P_{r|t}$ probability that x is recovered modulo r if C_t holds

- ▶ We recover x modulo $r_{\alpha,t}$ from $\beta \equiv \alpha^x \pmod t$
($r_{\alpha,t}$ is the multiplicative order of α modulo t)

Second step : $P_{r|t}$

$P_{r|t}$ probability that x is recovered modulo r if C_t holds

- ▶ We recover x modulo $r_{\alpha,t}$ from $\beta \equiv \alpha^x \pmod t$
($r_{\alpha,t}$ is the multiplicative order of α modulo t)
- ▶ $\alpha \equiv g^{-g^k \pmod t} \pmod t \Rightarrow r_{\alpha,t}$ is u.d. in $[0 \dots \varphi(t) - 1]$

Second step : $P_{r|t}$

$P_{r|t}$ probability that x is recovered modulo r if C_t holds

- ▶ We recover x modulo $r_{\alpha,t}$ from $\beta \equiv \alpha^x \pmod t$
($r_{\alpha,t}$ is the multiplicative order of α modulo t)
- ▶ $\alpha \equiv g^{-g^k \pmod t} \pmod t \Rightarrow r_{\alpha,t}$ is u.d. in $[0 \dots \varphi(t) - 1]$
- ▶ For $r = p_r^{e_r}$ we prove $1/2 \leq P_{r|t} \leq 1$

Second step : $P_{r|t}$

$P_{r|t}$ probability that x is recovered modulo r if C_t holds

- ▶ We recover x modulo $r_{\alpha,t}$ from $\beta \equiv \alpha^x \pmod t$
($r_{\alpha,t}$ is the multiplicative order of α modulo t)
- ▶ $\alpha \equiv g^{-g^k \pmod t} \pmod t \Rightarrow r_{\alpha,t}$ is u.d. in $[0 \dots \varphi(t) - 1]$
- ▶ For $r = p_r^{e_r}$ we prove $1/2 \leq P_{r|t} \leq 1$
- ▶ We approximate $P_{r|t} \approx 1$ for any $r = p_r^{e_r} | \varphi(t)$
(x is recovered modulo r iff $r | \varphi(t)$ and C_t holds)

Third step : $P[r]$

$P[r_j]$ probability to get x modulo r after one fault

- ▶ $x \bmod r$ is recovered for any t s.t. $r|\varphi(t)$ and C_t holds

Third step : $P[r]$

$P[r_j]$ probability to get x modulo r after one fault

- ▶ $x \bmod r$ is recovered for any t s.t. $r|\varphi(t)$ and C_t holds
- ▶ How to combine these probabilities?

Using

- ▶ $P_t \approx O(t^{-e})$ with $2 \leq e \leq 3$ (first step)
- ▶ $P_{r|t} \approx 1$ (second step)

Third step : $P[r]$

$P[r_j]$ probability to get x modulo r after one fault

- ▶ $x \bmod r$ is recovered for any t s.t. $r|\varphi(t)$ and C_t holds
- ▶ How to combine these probabilities?

Using

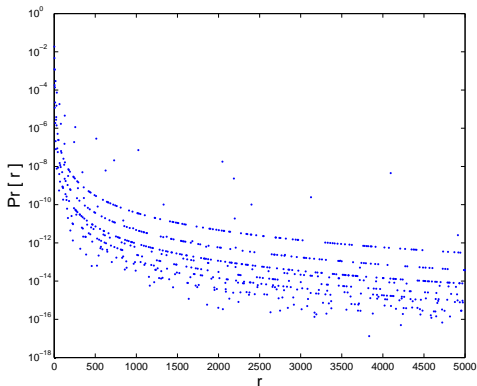
- ▶ $P_t \approx O(t^{-e})$ with $2 \leq e \leq 3$ (first step)
- ▶ $P_{r|t} \approx 1$ (second step)

We approximate

$$P[r] \approx P_{t_r} = 0.6 \frac{\varphi(t_r)}{t_r l_{t_r}^2} \quad \text{where } t_r = \arg \max_{t \text{ s.t. } r|\varphi(t)} P_t$$

Third step : $P[r]$

- ▶ $P[r_j]$ probability to get x modulo r after one fault



Fourth step : Number of faults required

- ▶ CRT : we need $\{x \bmod r_j\}_j$ such that $\text{lcm}(\{r_j\}_j) > x$

Fourth step : Number of faults required

- ▶ CRT : we need $\{x \bmod r_j\}_j$ such that $\text{lcm}(\{r_j\}_j) > x$
- ▶ If $P[r_j] > N^{-1}$, then likely to get $x \bmod r_j$ after N faults

Fourth step : Number of faults required

- ▶ CRT : we need $\{x \bmod r_j\}_j$ such that $\text{lcm}(\{r_j\}_j) > x$
- ▶ If $P[r_j] > N^{-1}$, then likely to get $x \bmod r_j$ after N faults
- ▶ **Number of bits of x recovered with N faults** is about

$$\log_2(\text{lcm}(\{r_j | P[r_j] \geq N^{-1}\}))$$

Fourth step : Number of faults required

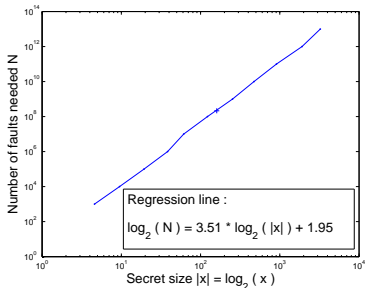
- ▶ CRT : we need $\{x \bmod r_j\}_j$ such that $\text{lcm}(\{r_j\}_j) > x$
- ▶ If $P[r_j] > N^{-1}$, then likely to get $x \bmod r_j$ after N faults
- ▶ **Number of bits of x recovered with N faults** is about

$$\log_2(\text{lcm}(\{r_j | P[r_j] \geq N^{-1}\}))$$

- ▶ This concludes the theoretical analysis

Theoretical Analysis: Results (DSA)

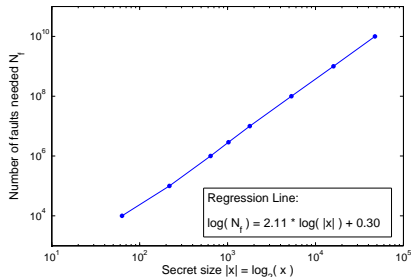
$N_s = 10^5$	$ x \approx \log_2(\text{lcm}(\{r_j\}))$
10^4	9
10^5	19
10^6	38
10^7	61
10^8	122
10^9	253
10^{10}	472
$2.14 \cdot 10^8$	160



- ▶ If secret size \leftarrow secret size $*2$,
then number of faults needed \leftarrow number of faults needed $*10$

Theoretical Analysis: Results (ElGamal)

$N_f \approx b_s^{-1}$	$ x \approx \log_2(\text{lcm}(\{r_j\}))$
10^4	63
10^5	218
10^6	641
10^7	1788
10^8	5319
10^9	16019
10^{10}	47468
$2.89 \cdot 10^6$	1024



- Attack a bit more than quadratic in the secret size

Outline

- ▶ Introduction
- ▶ New attacks on DLP-based schemes
 - ▶ A new fault attack on ElGamal
 - ▶ A new fault attack on DSA
- ▶ Theoretical analysis (DSA)
- ▶ Simulation results (DSA & ElGamal)
- ▶ Further work
- ▶ Conclusion

Simulation Results: DSA

- ▶ Simulations with p a 1024-bit prime, q a 160-bit prime and $|g| = |p|$

Exp. n°	$m = 32$	$m = 64$	$m = 96$	$m = 128$	$m = 160$
1	463986	5676441	8549083	26476140	38621903
2	811215	8898520	14945495	22174790	34861119
3	284706	3336328	5577267	11579152	20960118
4	798230	6438425	21496166	31049856	61711260
5	282811	8363054	14303797	26594960	39066576
6	721742	8762969	20601681	38398403	73933924
7	453623	3174393	10220088	17145832	22623221

Simulation Results: DSA

- ▶ Simulations with p a 1024-bit prime, q a 160-bit prime and $|g| = |p|$

Exp. n°	$m = 32$	$m = 64$	$m = 96$	$m = 128$	$m = 160$
1	463986	5676441	8549083	26476140	38621903
2	811215	8898520	14945495	22174790	34861119
3	284706	3336328	5577267	11579152	20960118
4	798230	6438425	21496166	31049856	61711260
5	282811	8363054	14303797	26594960	39066576
6	721742	8762969	20601681	38398403	73933924
7	453623	3174393	10220088	17145832	22623221

- ▶ Mean of $4.16 \cdot 10^7$ faults

Simulation Results: DSA

- ▶ Simulations with p a 1024-bit prime, q a 160-bit prime and $|g| = |p|$

Exp. n°	$m = 32$	$m = 64$	$m = 96$	$m = 128$	$m = 160$
1	463986	5676441	8549083	26476140	38621903
2	811215	8898520	14945495	22174790	34861119
3	284706	3336328	5577267	11579152	20960118
4	798230	6438425	21496166	31049856	61711260
5	282811	8363054	14303797	26594960	39066576
6	721742	8762969	20601681	38398403	73933924
7	453623	3174393	10220088	17145832	22623221

- ▶ Mean of $4.16 \cdot 10^7$ faults
- ▶ 3 to 10 times smaller than the $2.14 \cdot 10^8$ theoretical prediction

Simulation Results: ElGamal

- ▶ Simulations with p a 1024-bit prime

Exp n°	$m = 64$	$m = 128$	$m = 256$	$m = 512$	$m = 1024$
1	5794	39790	104258	786038	3366136
2	4902	24054	126230	611494	2893428
3	4902	24054	126230	571170	2688882
4	8810	21742	102974	572170	2931512
5	7420	23290	142810	696046	3011938
6	3078	43274	135194	636864	2808468

Simulation Results: ElGamal

- ▶ Simulations with p a 1024-bit prime

Exp n°	$m = 64$	$m = 128$	$m = 256$	$m = 512$	$m = 1024$
1	5794	39790	104258	786038	3366136
2	4902	24054	126230	611494	2893428
3	4902	24054	126230	571170	2688882
4	8810	21742	102974	572170	2931512
5	7420	23290	142810	696046	3011938
6	3078	43274	135194	636864	2808468

- ▶ Mean of $2.95 \cdot 10^6$ faults

Simulation Results: ElGamal

- ▶ Simulations with p a 1024-bit prime

Exp n°	$m = 64$	$m = 128$	$m = 256$	$m = 512$	$m = 1024$
1	5794	39790	104258	786038	3366136
2	4902	24054	126230	611494	2893428
3	4902	24054	126230	571170	2688882
4	8810	21742	102974	572170	2931512
5	7420	23290	142810	696046	3011938
6	3078	43274	135194	636864	2808468

- ▶ Mean of $2.95 \cdot 10^6$ faults
- ▶ Confirm theoretical predictions

Outline

- ▶ Introduction
- ▶ New attacks on DLP-based schemes
 - ▶ A new fault attack on ElGamal
 - ▶ A new fault attack on DSA
- ▶ Theoretical analysis (DSA)
- ▶ Simulation results (DSA & ElGamal)
- ▶ Further work
- ▶ Conclusion

Further Work

- ▶ Reduce the number of faults
 - ▶ Include an exhaustive search part

Further Work

- ▶ Reduce the number of faults
 - ▶ Include an exhaustive search part
- ▶ Guess the faulty values
 - ▶ Statistical methods

Further Work

- ▶ Reduce the number of faults
 - ▶ Include an exhaustive search part
- ▶ Guess the faulty values
 - ▶ Statistical methods
- ▶ Other attacks

Outline

- ▶ Introduction
- ▶ New attacks on DLP-based schemes
 - ▶ A new fault attack on ElGamal
 - ▶ A new fault attack on DSA
- ▶ Theoretical analysis (DSA)
- ▶ Simulation results (DSA & ElGamal)
- ▶ Further work
- ▶ **Conclusion**

Conclusion

- ▶ First fault attacks with fault injections *on the public key elements* of two *DLP-based schemes* (DSA and ElGamal signatures)

Conclusion

- ▶ First fault attacks with fault injections *on the public key elements* of two *DLP-based schemes* (DSA and ElGamal signatures)
- ▶ Theoretical & simulation-based analysis

Conclusion

- ▶ First fault attacks with fault injections *on the public key elements* of two *DLP-based schemes* (DSA and ElGamal signatures)
- ▶ Theoretical & simulation-based analysis
- ▶ Our attacks recover the whole keys

Conclusion

- ▶ First fault attacks with fault injections *on the public key elements* of two *DLP-based schemes* (DSA and ElGamal signatures)
- ▶ Theoretical & simulation-based analysis
- ▶ Our attacks recover the whole keys
- ▶ Compared to similar attacks against IFP and ECDL schemes, high number of faults needed

Questions?

Questions?

Questions?

Questions?

First step : P_t

- ▶ $\Pr[(\gcd(k, q_1^*) = 1) \wedge (\gcd(k, l_t) = 1)] \approx \Pr[\gcd(k, q_1^*) = 1] \Pr[\gcd(k, l_t) = 1]$ for most t because

$$\begin{aligned} \frac{\Pr[(\gcd(k, q_1^*) = 1) \wedge (\gcd(k, l_t) = 1)]}{\Pr[\gcd(k, q_1^*) = 1] \Pr[\gcd(k, l_t) = 1]} &\leq \frac{\Pr[\gcd(k, q_1^*) = 1]}{\Pr[\gcd(k, q_1^*) = 1] \Pr[\gcd(k, l_t) = 1]} \\ &= \frac{1}{\Pr[\gcd(k, l_t) = 1]} = \frac{l_t}{\varphi(l_t)} \end{aligned}$$

and the last is not too large for most t .

Third step : $P[r]$

- ▶ $P[r_j]$ probability to get x modulo r after one fault
- ▶ $x \bmod r$ is recovered *for any* t s.t. $r|\varphi(t)$ and C_t holds

Third step : $P[r]$

- ▶ $P[r_j]$ probability to get x modulo r after one fault
- ▶ $x \bmod r$ is recovered for any t s.t. $r|\varphi(t)$ and C_t holds
- ▶ How to combine these probabilities?
 - ▶ $P_T := \Pr[C_t = 1 \Leftrightarrow t \in T]$

$$P[r] \approx \sum_{\substack{T=\{t_i\} \\ \text{s.t. } \exists t \in T \text{ s.t. } r|\varphi(t)}} P_T$$

Third step : $P[r]$

- ▶ $P[r_j]$ probability to get x modulo r after one fault
- ▶ $x \bmod r$ is recovered for any t s.t. $r|\varphi(t)$ and C_t holds
- ▶ How to combine these probabilities?
 - ▶ $P_T := \Pr[C_t = 1 \Leftrightarrow t \in T]$

$$P[r] \approx \sum_{\substack{T=\{t_i\} \\ \text{s.t. } \exists t \in T \text{ s.t. } r|\varphi(t)}} P_T$$

- ▶ We approximate

$$P[r] \approx \sum_{\substack{T=\{t_i\} \\ \text{s.t. } t_r \in T}} P_T$$

Third step : $P[r]$

- ▶ $P[r_j]$ probability to get x modulo r after one fault
- ▶ $x \bmod r$ is recovered for any t s.t. $r|\varphi(t)$ and C_t holds
- ▶ How to combine these probabilities?
 - ▶ $P_T := \Pr[C_t = 1 \Leftrightarrow t \in T]$

$$P[r] \approx \sum_{\substack{T=\{t_i\} \\ \text{s.t. } \exists t \in T \text{ s.t. } r|\varphi(t)}} P_T$$

- ▶ We approximate

$$P[r] \approx \sum_{\substack{T=\{t_i\} \\ \text{s.t. } t_r \in T}} P_T = P_{t_r} \quad \text{where } t_r = \arg \max_{t \text{ s.t. } r|\varphi(t)} P_t$$