SECURE IMPLEMENATION OF ELLIPTIC CURVE CRYPTOGRAPHY

Marc Joye

ECC Workshop
Bochum, Sept. 20–22, 2004

GEMPLUS

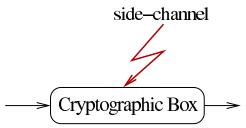
Agenda

- Part I: Reminder
- Part II: SPA-like attacks/countermeasures
- Part III: DPA-like attacks/countermeasures
- Part IV: Fault attacks/countermeasures

PART I:

REMINDER

Side-Channel Attacks



- *Side-channel*
  - timing attack, power attack, electro-magnetic attack, ...

Elliptic Curves

- *Definition.* If $\mathrm{Char}\mathbb{K} \neq 2, 3$ then an elliptic curve over $\mathbb{K}$ is the set of points $(x, y) \in \mathbb{K} \times \mathbb{K}$ satisfying the (non-singular) Weierstraß equation

$$y^2 = x^3 + ax + b \quad \cup \quad \{\mathcal{O}\}$$

- *Fact.* The points of an elliptic curve over a field $\mathbb{K}$ form an Abelian group, where the group operation $(+)$ is given by the "chord-and-tangent" law

Elliptic Curve Cryptography

- Koblitz and Miller (1985)
- *Elliptic Curve Discrete Logarithm Problem* (ECDLP)
  - Given $\boldsymbol{P}$ and $\boldsymbol{Q} = d\boldsymbol{P} = \underbrace{\boldsymbol{P} + \boldsymbol{P} + \cdots + \boldsymbol{P}}_{d \text{ times}}$ find d
  - No sub-exponential algorithm for solving the ECDLP (in the general case)
    - ✓ shorter key lengths (ECC-160 $\simeq$ RSA-1024)
- Applications
  - Key exchange, digital signature, encryption

Elliptic Curve Cryptography

- ECDSA (Elliptic Curve Digital Signature Algorithm)
  - Parameters:
    - ✓ curve E with $\#E = h\,n$ and $\boldsymbol{P} \in E$ of order n
    - ✓ private: d; public: $\{\boldsymbol{P}, \boldsymbol{Q} = d\boldsymbol{P}, n\}$
  - Signature generation (r, s)
    1. $k\boldsymbol{P} = (x_k, y_k)$ for a random k
    2. $r := x_k \bmod n$ and $s := k^{-1}[h(m) + dr] \bmod n$
  - Signature verification
    1. $u_1 = s^{-1}\,h(m) \bmod n$ and $u_2 = s^{-1}\,r \bmod n$
    2. check whether $\mathrm{x}(u_1\boldsymbol{P} + u_2\boldsymbol{Q}) \stackrel{?}{\equiv} r \pmod n$

PART II:

SPA-LIKE ATTACKS

SPA-like Attacks

- Addition formulæ
  - Let $\boldsymbol{P_1} = (x_1, y_1)$ and $\boldsymbol{P_2} = (x_2, y_2)$ then $\boldsymbol{P_1} + \boldsymbol{P_2} = (x_3, y_3)$ where

$$x_3 = \lambda^2 - x_1 - x_2 \quad \text{and} \quad y_3 = \lambda(x_1 - x_3) - y_1$$

$$\text{with } \lambda = \begin{cases} \dfrac{3x_1^2 + a}{2y_1} & \text{if } \boldsymbol{P_1} = \boldsymbol{P_2} \text{ (i.e., doubling)} \\ \dfrac{y_1 - y_2}{x_1 - x_2} & \text{if } \boldsymbol{P_1} \neq \pm\boldsymbol{P_2} \text{ (i.e., true addition)} \end{cases}$$

- Subject to side-channel attacks

SPA-like Attacks

- Protections against simple side-channel attacks
  - Make the addition formulæ *indistinguishable*
    - ✓ universal addition formula (i.e., valid for adding and doubling) [Brier & Joye, PKC 2002] [Olson, CHES 2004]
    - ✓ other parameterizations
      - Hessian curves [Joye & Quisquater, CHES 2001]
      - Jacobian curves [Liardet & Smart, CHES 2001] [Billet & Joye, AAECC-15, 2003]
    - ✓ dummy operations [Trichina & Belleza, CHES 2002] [Chevallier, Ciet & Joye, IEEE TC, 2004]

SPA-like Attacks

- Protections against simple side-channel attacks
  - Make the algorithm *regular*
    - ✓ double-and-add *always* [Coron, CHES '99]
    - ✓ universal exponentiation [Clavier & Joye, CHES 2001]
    - ✓ exponent rewritting [Joye, Electronics Letters, 2002] [Chevallier, Ciet & Joye, IEEE TC, 2004]
    - ✓ Montgomery ladder [Joye & Yen, CHES 2002] [López & Dahab, CHES '99] [Okeya & Sakurai, INDOCRYPT 2000] [Brier & Joye, PKC 2002], [Izu & Takagi, PKC 2002] & [Fischer, Giraud, Knudsen & Seifert, IACR ePrint, 2002]

Unified Formulæ: General Case

- Curve equation

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

- Addition formulæ
  - Let $\boldsymbol{P_1} = (x_1, y_1)$ and $\boldsymbol{P_2} = (x_2, y_2)$ then $\boldsymbol{P_1} + \boldsymbol{P_2} = (x_3, y_3)$ where

$$x_3 = \lambda^2 - x_1 - x_2 \quad \text{and} \quad y_3 = \lambda(x_1 - x_3) - y_1$$

$$\text{with } \lambda = \begin{cases} \dfrac{3x_1^2 + a}{2y_1} & \text{if } \boldsymbol{P_1} = \boldsymbol{P_2} \text{ (i.e., doubling)} \\ \dfrac{y_1 - y_2}{x_1 - x_2} & \text{if } \boldsymbol{P_1} \neq \pm\boldsymbol{P_2} \text{ (i.e., true addition)} \end{cases}$$

Unified Formulæ: General Case

- In all cases, define

$$\boxed{\lambda = \frac{x_1^2 + x_1 x_2 + x_2^2 + a}{y_1 + y_2}}$$

- This technique generalizes to
  - ✓ curves given by the full Weiertraß equation

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

  - ✓ projective coordinates
  - ✓ quartic curves

Unified Formulæ: Hessian Curves

- Weierstraß form:

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

- Hessian form:

$$u^3 + v^3 + 1 = 3Duv$$

- only valid for curves E with $\#E = h\,n$ and $h \propto 3$
  - ✓ $(0,-1)$ and $(-1,0)$ on the Hessian form
  - ✓ ECC standards recommend curves with $h \in \{1,2,3,4\}$

Unified Formulæ: Hessian Curves

- A key observation
  - Addition and doubling
    - ✓ $(U_1 : V_1 : W_1) + (U_2 : V_2 : W_2) =$
      $(V_1^2 U_2 W_2 - V_2^2 U_1 W_1 : U_1^2 V_2 W_2 - U_2^2 V_1 W_1 :$
      $W_1^2 U_2 V_2 - W_2^2 U_1 V_1)$
    - ✓ $2(U_1 : V_1 : W_1) =$
      $(V_1(U_1^3 - W_1^3) : U_1(W_1^3 - V_1^3) : W_1(V_1^3 - U_1^3))$
  - *Observation.*

$$(W_1 : U_1 : V_1) + (V_1 : W_1 : U_1) = 2(U_1 : V_1 : W_1)$$

Unified Formulæ: Hessian Curves

- Let $P_1 = (U_1 : V_1 : W_1)$ and $P_2 = (U_2 : V_2 : W_2)$. Then
  - HessianAdd$(U_1 : V_1 : W_1, U_2 : V_2 : W_2) = P_1 + P_2$
  - HessianAdd$(W_1 : U_1 : V_1, V_1 : W_1 : U_1) = 2P_1$
  - HessianAdd$(U_1 : V_1 : W_1, V_2 : U_2 : W_2) = P_1 - P_2$

Unified Formulæ: Jacobian Curves

- Jacobi form
  - as the intersection of 2 quadrics:

$$\begin{cases} u^2 + v^2 = 1 \\ k^2 u^2 + w^2 = 1 \end{cases}$$

  - ✓ only valid for curves E with $\#E = h\,n$ and $h \propto 4$
  - as a quartic:

$$v^2 = \epsilon u^4 - 2\delta u^2 + 1$$

  - ✓ only valid for curves E with $\#E = h\,n$ and $h \propto 2$ (and $h \propto 4$ when $\epsilon = 1$)

Unified Formulæ: Jacobian Curves

- Projective (extended) Jacobi equation:

$$\boxed{V^2 = \epsilon U^4 - 2\delta U^2 W^2 + W^4}$$

- Addition formulæ
  - Let $P_1 = (U_1 : V_1 : W_1)$ and $P_2 = (U_2 : V_2 : W_2)$
    - ✓ **Negation** $-P_1 = (-U_1 : V_1 : W_1)$
    - ✓ **Addition** $P_1 + P_2 = (U_3 : V_3 : W_3)$ with

$$\begin{cases} U_3 &= U_1 W_1 V_2 + V_1 U_2 W_2 \\ V_3 &= [(W_1 W_2)^2 + \epsilon (U_1 U_2)^2]\,[V_1 V_2 - 2\delta U_1 U_2 W_1 W_2] \\ &\quad + 2\epsilon U_1 U_2 W_1 W_2 (U_1^2 W_2^2 + W_1^2 U_2^2) \\ W_3 &= (W_1 W_2)^2 - \epsilon (U_1 U_2)^2 \end{cases}$$

    - ✓ also valid for doubling

Unified Formulæ: Summary

- Comparison

Model	# Mult.	# Coord.	h
Universal formula	17+1†	3	–
Hesse	12	3	3
Jacobi ($\cap$ 2 quadrics)	16+1	4	4
(ext'd) Jacobi quartic	13+3	3	2
Jacobi quartic	13+1	3	4

† 16+1 on a Weierstraß elliptic curve with $a = -1$

Dummy Operations: D&A *always*

- Goal: given $d = \sum_{i=0}^{m-1} d_i 2^i$, compute $\boldsymbol{Q} = d\boldsymbol{P}$
  - Double-and-add *always* algorithm
    - ✓ add a dummy addition when $d_i = 0$

Input: $\boldsymbol{P}, d = (1, d_{m-2}, \ldots, d_0)_2$
Output: $\boldsymbol{Q} = d\boldsymbol{P}$

$\boldsymbol{R_0} \leftarrow \boldsymbol{P}$
for $i = m-2$ down to 0 do
  $\boldsymbol{R_0} \leftarrow 2\boldsymbol{R_0}$
  $b \leftarrow 1 - d_i$; $\boldsymbol{R_b} \leftarrow \boldsymbol{R_b} + \boldsymbol{P}$
return $\boldsymbol{R_0}$

$d_i = 1 \Rightarrow b = 0$
$d_i = 0 \Rightarrow b = 1$

Dummy Operations: Atomicity

- Make addition and doubling indistinguishable
- Illustration: adding points on $E(GF(2^q))$
  - Addition formulæ
    - ✓ Let $\boldsymbol{P_1} = (x_1, y_1)$ and $\boldsymbol{P_2} = (x_2, y_2)$
    - ✓ **Addition** $\boldsymbol{P_1} + \boldsymbol{P_2}$ is (x_3, y_3) where

$$x_3 = a + \lambda^2 + \lambda + x_1 + x_2, \quad y_3 = (x_1 + x_3)\lambda + x_3 + y_1$$

with $\lambda = (y_1 + y_2)/(x_1 + x_2)$

- ✓ **Doubling** $2\boldsymbol{P_1} = (x_3, y_3)$ where

$$x_3 = a + \lambda^2 + \lambda, \quad y_3 = (x_1 + x_3)\lambda + x_3 + y_1$$

with $\lambda = x_1 + (y_1/x_1)$

Dummy Operations: Atomicity

- Indistinguishable addition/doubling

Addition: $\boldsymbol{P_1} \leftarrow \boldsymbol{P_2} + \boldsymbol{P_1}$		Doubling: $\boldsymbol{P_1} \leftarrow 2\boldsymbol{P_1}$	
$T_1 \leftarrow T_1 + T_3$	$(= x_1 + x_2)$	$T_6 \leftarrow T_1 + T_3$	(fake)
$T_2 \leftarrow T_2 + T_4$	$(= y_1 + y_2)$	$T_6 \leftarrow T_3 + T_6$	$(= x_1)$
$T_5 \leftarrow T_2/T_1$	$(= \lambda)$	$T_5 \leftarrow T_2/T_1$	$(= y_1/x_1)$
$T_1 \leftarrow T_1 + T_5$		$T_5 \leftarrow T_1 + T_5$	$(= \lambda)$
$T_6 \leftarrow T_5^2$	$(= \lambda^2)$	$T_1 \leftarrow T_5^2$	$(= \lambda^2)$
$T_6 \leftarrow T_6 + a$	$(= \lambda^2 + a)$	$T_1 \leftarrow T_1 + a$	$(= \lambda^2 + a)$
$T_1 \leftarrow T_1 + T_6$	$(= x_3)$	$T_1 \leftarrow T_1 + T_5$	$(= x_3)$
$T_2 \leftarrow T_1 + T_4$	$(= x_3 + y_2)$	$T_2 \leftarrow T_1 + T_2$	$(= x_3 + y_1)$
$T_6 \leftarrow T_1 + T_3$	$(= x_2 + x_3)$	$T_6 \leftarrow T_1 + T_6$	$(= x_1 + x_3)$
$T_5 \leftarrow T_5 \cdot T_6$		$T_5 \leftarrow T_5 \cdot T_6$	
$T_2 \leftarrow T_2 + T_5$	$(= y_3)$	$T_2 \leftarrow T_2 + T_5$	$(= y_3)$

return (T_1, T_2)

Dummy Operations: Summary

- Dummy operations allow
  - to make an algorithm regular
    - ✓ e.g., double-and-add *always* algorithm
  - to make addition/doubling indistinguishable
    - ✓ e.g., addition and doubling on $E(GF(2^q))$
- Drawback
  - they penalize the running-time

Regular Algorithms: UEA

- Universal Exponentiation Algorithm (UEA)

$$\Gamma(d) = \{(\gamma(i); \alpha(i), \beta(i))\}_{1 \le i \le r}$$

  - Provide an easy means to evaluate $\boldsymbol{Q} = d\boldsymbol{P}$:
    - ✓ For $i = 1$ to r compute

$$d^{(i)}\boldsymbol{P} = d^{(j(i))}\boldsymbol{P} + d^{(k(i))}\boldsymbol{P}$$

and then set $\boldsymbol{Q} = d^{(r)}\boldsymbol{P}$

$\boldsymbol{R}_{\gamma(i)}$ $\boldsymbol{R}_{\alpha(i)}$ $\boldsymbol{R}_{\beta(i)}$

Regular Algorithms: UEA

- Universal Exponentiation Algorithm (UEA)

Input: $\boldsymbol{P}, \Gamma(d) = \{(\gamma(i); \alpha(i), \beta(i))\}_{1 \le i \le r}$
Output: $\boldsymbol{Q} = d\boldsymbol{P}$

$\boldsymbol{R}_{\alpha(1)} \leftarrow \boldsymbol{P}$
for $i = 1$ to r do
  $\boldsymbol{R}_{\gamma(i)} \leftarrow \boldsymbol{R}_{\alpha(i)} + \boldsymbol{R}_{\beta(i)}$
return $\boldsymbol{R}_{\gamma(r)}$

- Drawback: non-standard representation
  - ✓ mainly applies to fixed d
  - ✓ on-the-fly computation for variable d (e.g., MIST)

Regular Algorithms: Exponent Rew.

- Exponent rewritting
  - Double-and-add algorithm (protected)

$d_i = 1 \Rightarrow$ $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_1}$; $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_0}$; $i \leftarrow i - 1$

$d_i = 0 \Rightarrow$ $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_0}$; $i \leftarrow i - 1$

Input: $\boldsymbol{P}, d = (1, d_{m-2}, \ldots, d_0)_2$
Output: $\boldsymbol{Q} = d\boldsymbol{P}$

$\boldsymbol{R_0} \leftarrow 2\boldsymbol{P}$; $\boldsymbol{R_1} \leftarrow \boldsymbol{P}$; $i \leftarrow m - 2$
while $(i \ge 1)$ do
  $b \leftarrow (1 - d_i)$
  $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R}_{d_i}$
  $d_i \leftarrow 0$; $i \leftarrow i - b$
if $(d_0 = 1)$ then $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_1}$
return $\boldsymbol{R_0}$

Regular Algorithms: Exponent Rew.

- Atomic exponentiation
  - Double-and-add algorithm

$d_i = 1 \Rightarrow$ $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_1}$; $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_0}$; $i \leftarrow i - 1$

$d_i = 0 \Rightarrow$ $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_0}$; $i \leftarrow i - 1$

Input: $\boldsymbol{P}, d = (1, d_{m-2}, \ldots, d_0)_2$
Output: $\boldsymbol{Q} = d\boldsymbol{P}$

$\boldsymbol{R_0} \leftarrow 2\boldsymbol{P}$; $\boldsymbol{R_1} \leftarrow \boldsymbol{P}$; $i \leftarrow m - 2$; $b \leftarrow 0$
while $(i \ge 0)$ do
  $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_b}$
  $b \leftarrow b \oplus d_i$; $i \leftarrow i - \neg b$
return $\boldsymbol{R_0}$

Regular Algorithms: Montgomery L.

- Montgomery powering ladder

Input: $\boldsymbol{P}, d = (1, d_{m-2}, \ldots, d_0)_2$
Output: $\boldsymbol{Q} = d\boldsymbol{P}$

$\boldsymbol{R_0} \leftarrow \boldsymbol{P}$; $\boldsymbol{R_1} \leftarrow 2\boldsymbol{P}$
for $i = m - 2$ down to 0 do
  $b \leftarrow (1 - d_i)$
  $\boldsymbol{R_b} \leftarrow \boldsymbol{R_0} + \boldsymbol{R_1}$; $\boldsymbol{R}_{d_i} \leftarrow 2\boldsymbol{R}_{d_i}$
return $\boldsymbol{R_0}$

- Computations can be carried out with x-coordinates only
  - ✓ faster algorithm + memory savings

Regular Algorithms: Summary

- 2 types of regular algorithms:
  - Algorithms being intrinsically regular
    - ✓ e.g., double-and-add *always*, Montgomery ladder
  - Algorithms repeating the same basic operation
    - ✓ e.g., universal exponentiation (UEA), exponent rewritting
    - ✓ require that addition and doubling are indistinguishable (!)

PART III:

DPA-LIKE ATTACKS

DPA-like Attacks

- Protections against SPA-like attacks are not enough to thwart DPA-like attacks
  - More sophisticated attacks
    - ✓ requiring several measurements
    - ✓ statistical tools

DPA-like Attacks

- Illustration: attacking the double-and-add *always*
  - Find secret d in the computation $Q = dP$

Input: $P, d = (1, d_{m-2}, \ldots, d_0)_2$
Output: $Q = dP$

$R_0 \leftarrow P$
for $i = m-2$ down to 0 do
  $R_0 \leftarrow 2R_0$
  $b \leftarrow 1 - d_i$; $R_b \leftarrow R_b + P$
return R_0

DPA-like Attacks

- Illustration: attacking the double-and-add *always*
  - Find secret d in the computation $Q = dP$
    - ✓ at step j, the attacker
      - already knows bits $d_{m-1}, d_{m-2}, \ldots, d_{j+1}$
      - guesses that next bit $d_j = 1$
      - chooses t points $P_1, \ldots, P_t$ and computes
        $$Q_r = \Big(\sum_{i=j}^{m-1} d_i\, 2^{i-j}\Big) P_r \quad \text{for } 1 \le r \le t$$
      - prepares two sets $\mathcal{S}_0 = \{P_r \mid g(Q_r) = 0\}$ and $\mathcal{S}_1 = \{P_r \mid g(Q_r) = 1\}$
      - if $\langle \mathcal{C}(r) \rangle_{\substack{1 \le r \le t \\ P_r \in \mathcal{S}_1}} - \langle \mathcal{C}(r) \rangle_{\substack{1 \le r \le t \\ P_r \in \mathcal{S}_1}} \not\approx 0$ then $d_j = 1$
    - ✓ iterate the attack to find $d_{j-1}, \ldots$

DPA-like Attacks

- Illustration: attacking the double-and-add *always*
  - Find secret d in the computation $Q = dP$
    - ✓ the previous attack requires that
      - the crypto-device computes $Q = dP$ for a fixed d ($\Rightarrow$ does not apply to ECDSA)
      - the attacker can evaluate
        $$g(Q_r) \quad \text{with } Q_r = \Big(\sum_{i=j}^{m-1} d_i\, 2^{i-j}\Big) P_r$$
  - Countermeasures:
    - ✓ randomize P in the computation of $Q = dP$
    - ✓ randomize d in the computation of $Q = dP$

DPA-like Attacks

- To thwart DPA-like attacks, it is recommended to randomize the inputs of the scalar multiplication
- Randomizing basepoint P
  - blinding [Kocher *et al.*, CRYPTO '99]
  - projective representation [Coron, CHES '99]
  - isomorphism [Joye & Tymen, CHES 2001]
- Randomizing multiplier d
  - blinding [Kocher *et al.*, CRYPTO '99]
  - splitting [Chari *et al.*, CRYPTO '99]
  - Frobenius endomorphism [Joye & Tymen, CHES 2001]

Randomizing the Basepoint

- Goal: compute $Q = dP$
  - Blinding of point P
    - ✓ $S = dR$ for a secret point R
      $$Q = d(R + P) - S = dP$$
    - ✓ (R, S) in EEPROM updated by (kR, kS) for a (small) random k

Randomizing the Basepoint

- Goal: compute $Q = dP$
  - Randomized projective coordinates
    - ✓ Let $P = (x, y)$ $(\neq \mathcal{O})$
      - homogeneous coordinates:
        [$P = (X : Y : Z)$ with $x = X/Z$ and $y = Y/Z$]

$$Q = d(\lambda x : \lambda y : \lambda) \quad \text{for a random } \lambda \neq 0$$

      - projective Jacobian coordinates:
        [$P = (X : Y : Z)$ with $x = X/Z^2$ and $y = Y/Z^3$]

$$Q = d(\theta^2 x : \theta^3 y : \theta) \quad \text{for a random } \theta \neq 0$$

Randomizing the Basepoint

- Goal: compute $Q = dP$
  - Isomorphisms of curves
    - ✓ the elliptic curves

$$E : y^2 = x^3 + ax + b \text{ and } E' : y^2 = x^3 + a'x + b'$$

      are isomorphic $\iff \exists u \neq 0$ s.t. $u^4 a' = a$ and $u^6 b' = b$
    - ✓ the isomorphism is given by

$$\begin{cases} \varphi : & E \to E', \quad P = (x, y) \mapsto P' = (u^{-2}x, u^{-3}y) \\ \varphi^{-1} : & E' \to E, \quad P' = (x', y') \mapsto P = (u^2 x', u^3 y') \end{cases}$$

Randomizing the Basepoint

- Goal: compute $Q = dP$
  - Isomorphisms of curves

$$\begin{array}{ccc} P \in E & \xrightarrow{\text{mult. by } d} & Q = dP \in E \\ \varphi \downarrow & & \uparrow \varphi^{-1} \\ P' \in E' & \xrightarrow{\text{mult. by } d} & Q' = dP' \in E' \end{array}$$

$$\Rightarrow Q = \varphi^{-1}(d \cdot \varphi(P)) = \varphi^{-1}(dP') = \varphi^{-1}(Q') = Q$$

Randomizing the Multiplier

- Goal: compute $Q = dP$
  - Blinding of multiplier d
    - ✓ $\#E = h\,n$ and $\mathrm{ord}_E(P) = n$
    - ✓ $d^* = d + k\,n$

$$Q = d^*P = dP + (k\,n)P = dP$$

Randomizing the Multiplier

- Goal: compute $Q = dP$
  - Splitting of multiplier d
    - ✓ additive

$$Q = rP + (d - r)P$$

    - ✓ multiplicative

$$Q = (dr^{-1})(rP)$$

    - ✓ Euclidean

$$Q = (d \bmod r)P + \lfloor d/r \rfloor (rP)$$

  - Shamir-Strauss double ladder

Randomizing the Multiplier

- Goal: compute $Q = dP$
  - Frobenius on ABC curves
    - ✓ Koblitz curves (ABC)

$$E_{/GF(2^q)} : y^2 + xy = x^3 + ax^2 + 1$$

      with $a \in GF(2)$
    - ✓ Frobenius endormorphism

$$\tau : P = (x, y) \mapsto \tau(P) = (x^2, y^2)$$

Randomizing the Multiplier

- Goal: compute $Q = dP$
  - Frobenius on ABC curves
    - ✓ *Fact 1.* $\mathbb{Z} \subset \mathbb{Z}[\tau] \subset \mathrm{End}(E)$
    - ✓ *Fact 2.* $\tau^q = 1$, i.e., $(\tau^q - 1)P = \mathcal{O}$

$$\Rightarrow Q = dP = tP \quad \text{with } t = d \bmod (\tau^q - 1)$$
$$= \sum t_i \tau^i(P) \quad \text{with } t_i \in \{-1, 0, 1\}$$
$$\text{and } \tau(x, y) = (x^2, y^2)$$

- ✓ Letting $t^* = d \bmod \rho(\tau^q - 1)$, we have

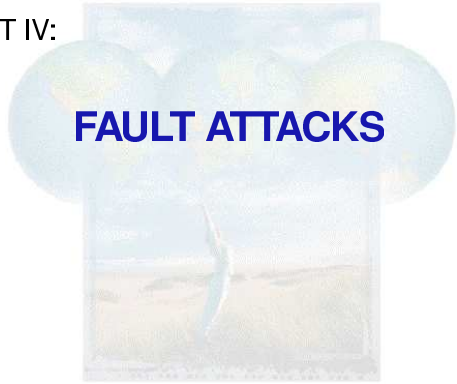
$$Q = t^* P = dP$$

since $t^* \equiv d \pmod{(\tau^q - 1)}$

DPA-like Attacks: Summary

- To thwart DPA-like attacks, it is recommended to randomize
  - basepoint P,
    - ✓ e.g., blinding, projective representation, isomorphism
  - *and* multiplier d
    - ✓ e.g., blinding, splitting, Frobenius endomorphism
- (Refined power analysis) [Goubin, PKC 2003] [Akishita & Takagi, ISC 2003]

PART IV:



FAULT ATTACKS

Fault Attacks

- [EUROCRYPT '97] Dan Boneh, Richard DeMillo, and Richard Lipton, "On the importance of checking cryptographic protocols for faults"
- [CRYPTO '97] Eli Biham and Adi Shamir, "Differential fault analysis of secret key cryptosystems"
- [CRYPTO 2000] Ingrid Biehl, Bernd Meyer, and Volker Müller, "Differential fault attacks on elliptic curve cryptosystems"
- [IACR ePrint 2003] Mathieu Ciet and Marc Joye, "Elliptic curve cryptosystems in the presence of permanent and transient faults"

Fault Attacks

- Attack Model
  - EC parameters are in non-volatile memory
    - ✓ permanent faults
      - in a unknown position
      - in any system parameter
    - ✓ transient faults
      - during parameter transfer
- Adversary's goal
  - recover the value of d in the computation of $Q = dP$

BMM Observation

- Addition formulæ
  - Elliptic curve $E : y^2 = x^3 + ax + b$
  - Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ then $P_1 + P_2 = (x_3, y_3)$ where

$$x_3 = \lambda^2 - x_1 - x_2 \quad \text{and} \quad y_3 = \lambda(x_1 - x_3) - y_1$$

with

$$\lambda = \begin{cases} \dfrac{3x_1^2 + a}{2y_1} & \text{if } P_1 = P_2 \text{ (i.e., doubling)} \\ \dfrac{y_1 - y_2}{x_1 - x_2} & \text{if } P_1 \neq \pm P_2 \text{ (i.e., true addition)} \end{cases}$$

- Parameter b is not involved

BMM Observation

- Elliptic curve $E : y^2 = x^3 + ax + b$ over $\mathbb{F}_p$ and $\boldsymbol{P} \in E$
- If a 'point' $\widetilde{\boldsymbol{P}} = (\tilde{x}, \tilde{y}) \in \mathbb{F}_p \times \mathbb{F}_p$ but $\widetilde{\boldsymbol{P}} \notin E$ then the computation of $d\widetilde{\boldsymbol{P}}$ will take place on the curve

$$\widetilde{E} : y^2 = x^3 + ax + \tilde{b}$$

  - ✓ if (i) $\mathrm{ord}_{\widetilde{E}} \widetilde{P} = r$ is small, and (ii) discrete logarithms are computable in $\langle \widetilde{\boldsymbol{P}} \rangle$, then

$$d \pmod{r}$$

  can be recovered

BMM Observation

- Elliptic curve $E : y^2 = x^3 + ax + b$ over $\mathbb{F}_p$ and $\boldsymbol{P} \in E$
- If a 'point' $\widetilde{\boldsymbol{P}} = (\tilde{x}, \tilde{y}) \in \mathbb{F}_p \times \mathbb{F}_p$ but $\widetilde{\boldsymbol{P}} \notin E$ then the computation of $d\widetilde{\boldsymbol{P}}$ will take place on the curve

$$\widetilde{E} : y^2 = x^3 + ax + \tilde{b}$$

- Iterating the process for several $\boldsymbol{P}_i$ gives
  - $d \pmod{r_i}$ for several r_i
  - d by Chinese remaindering

Fault Analysis

- Problem: recover d in $\boldsymbol{Q} = d\boldsymbol{P}$ on

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

  over $\mathbb{F}_p$
- Fault analysis
  - faults in basepoint $\boldsymbol{P}$
  - faults in definition field $\mathbb{F}_p$
  - faults in curve parameters

Fault Analysis

- Problem: recover d in $\boldsymbol{Q} = d\boldsymbol{P}$ on

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

  over $\mathbb{F}_p$
- Fault analysis
  - faults in basepoint $\boldsymbol{P}$
    - ✓ device computes $\widetilde{\boldsymbol{Q}} = d\widetilde{\boldsymbol{P}}$ with $\widetilde{\boldsymbol{P}} = (\tilde{x}, y)$
    - ✓ $\widetilde{\boldsymbol{Q}} = d(\tilde{x}, y) = (\tilde{x}_d, \tilde{y}_d) \in \widetilde{E}$
      $\Rightarrow \tilde{b} = \tilde{y}_d^2 - \tilde{x}_d^3 - a\tilde{x}_d$
    - ✓ $\tilde{x}$ is a root in $\mathbb{F}_p[X]$ of $X^3 + aX + \tilde{b} - y^2$
  - compute $d \pmod{r}$ from $\widetilde{Q} = d\widetilde{P}$

Fault Analysis

- Problem: recover d in $\boldsymbol{Q} = d\boldsymbol{P}$ on

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

  over $\mathbb{F}_p$
- Fault analysis
  - faults in definition field $\mathbb{F}_p$
    - ✓ device computes $\widetilde{\boldsymbol{Q}} = d\widetilde{\boldsymbol{P}}$ with $\widetilde{\boldsymbol{P}} = (\tilde{x}, \tilde{y})$ with $\tilde{x} \equiv x \pmod{\tilde{p}}$ and $\tilde{y} \equiv y \pmod{\tilde{p}}$
    - ✓ $\widetilde{\boldsymbol{Q}} = d(\tilde{x}, y) = (\tilde{x}_d, \tilde{y}_d) \in \widetilde{E}$
      $\Rightarrow \tilde{b} \equiv \tilde{y}_d^2 - \tilde{x}_d^3 - a\tilde{x}_d \equiv y^2 - x^3 - ax \pmod{\tilde{p}}$
    - ✓ $\tilde{p}$ divides $(\tilde{y}_d^2 - \tilde{x}_d^3 - a\tilde{x}_d) - (y^2 - x^3 - ax)$
  - compute $d \pmod{r}$ from $\widetilde{Q} = d\widetilde{P}$ [e.g., Mersenne p]

Fault Analysis

- Problem: recover d in $\boldsymbol{Q} = d\boldsymbol{P}$ on

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

  over $\mathbb{F}_p$
- Fault analysis
  - faults in curve parameters
    - ✓ $\widetilde{\boldsymbol{Q}} = d(x, y) = (\tilde{x}_d, \tilde{y}_d) \in \widetilde{E} : y^2 = x^3 + \tilde{a}x + \tilde{b}$
    - ✓ $\begin{cases} y^2 = x^3 + \tilde{a}x + \tilde{b} \\ y_d^2 = x_d^3 + \tilde{a}x + \tilde{b} \end{cases}$
      $\Rightarrow \tilde{a} = \ldots, \tilde{b} = \ldots$
  - compute $d \pmod{r}$ from $\widetilde{Q} = d\widetilde{P}$

Fault Analysis

- Problem: recover d in $Q = dP$ on
$$E : y^2 = x^3 + ax + b$$
over $\mathbb{F}_p$
- Fault analysis
  - Check public and secret parameters for faults

CONCLUSION

Conclusion

- Efficient and various countermeasures are known for computing $Q = dP$ against
  - SPA-like attacks
  - DPA-like attacks
- Combine several methods
- Check public parameters for faults