Combined Attack on ECC using Points of Low Order

Junfeng Fan, Benedikt Gierlichs, Frederik Vercauteren

COSIC, Katholieke Universiteit Leuven, Belgium

- ECC: Elliptic curve over finite field
  - A set of points P(x,y) and $\mathcal{O}$ at infinity
- $\mathcal{O}$ required to form an abelian group
- But in crypto you should never see $\mathcal{O}$
- $\mathcal{O}$ is not easy to deal with in implementation
  - Point addition: if $P \neq \pm Q$ then . . . else
  - Point doubling: if $ord(P) > 2$ then . . . else

  } $\mathcal{O}$ appears
- But these cases should never occur anyway

So how does your implementation deal with $\mathcal{O}$?

ECC background

- E over $\mathbf{F}_p$: $y^2 = x^3 + ax + b \quad a, b \in \mathbf{F}_p \quad 4a^3 + 27b^2 \neq 0$
- $E(K) := \{(x, y) \mid y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\}$
- $\#E(K) \simeq \#K$ with error $\leq 2\sqrt{\#K}$
- Use in crypto: scalar multiplication k·P
  - EC discrete logarithm problem: given P and k·P, find k
  - Hard because order of P huge on strong curves
  - Implemented as sequence of 'small' operations

$$P \xrightarrow{\text{scalar multiplication}} k \cdot P$$

Group law and implementation

- Addition: $P + Q = (x_3, y_3)$ with $x_3 = (\frac{y_2 - y_1}{x_2 - x_1})^2 - x_1 - x_2$

  $$y_3 = (\frac{y_2 - y_1}{x_2 - x_1})(x_1 - x_3) - y_1$$

- Doubling: $2P = (x_3, y_3)$ with $x_3 = (\frac{3x_1^2 + a}{2y_1}) - x_1 - x_2$

  $$y_3 = (\frac{3x_1^2 + a}{2y_1})(x_1 - x_3) - y_1$$

- Note that b is not used in the formulae

Group law and implementation (2)

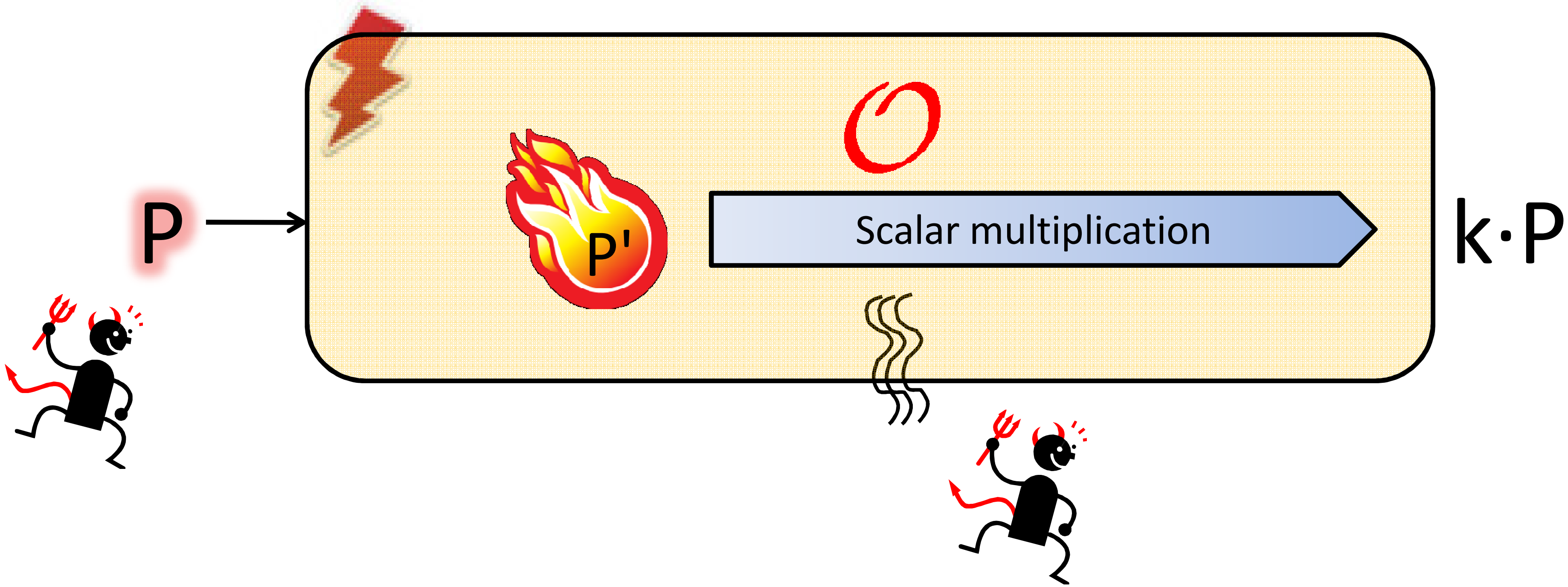
- Implementations:
  - Coordinate system: affine, projective, Jacobian, etc.

- Full domain correct:
  - Implementation computes P+Q and 2·P correctly on the whole domain
  - For Weierstrass curves this typically requires IF statements

- Partial domain correct: not full domain correct
  - For some inputs, implementation
    - Crashes, e.g. division by zero for affine coordinates
    - No crash but gets stuck at fixed point

The attack

- Setting: target computes k·P for any given P, k is secret
- Idea: choose rogue input P s.t. a <u>fault ε</u> turns it into
  - P' is point of very low order



Points with low-order neighbours

- Given curve E: $y^2 = x^3 + ax + b$, integers l and δ
- Construct P (x_p, y_p) on E s.t.
  - $\exists$ curve E': $y^2 = x^3 + ax + b'$
  - With P'$(x_{p'}, y_{p'})$ of order l on E'
  - Hamming dist. of bit representations $x_p || y_p$ and $x_{p'} || y_{p'}$ is δ
- If δ = 1 we call P and P' neighbours

- Input: E, l, δ $\longrightarrow$

- Output: P and P' $\longleftarrow$

Attack against a toy implementation

- Full domain correct
  - $\mathcal{O}$ and all following computations will be handled correctly
- Double and add scalar multiplication

- Input P with neighbour P' of order 4, inject fault and measure
- Doubling: 2·P', $\boxed{2\cdot 2P'}$, 2·3P' or $\boxed{2\cdot\mathcal{O}}$ (borderline cases)
- Addition: generates always odd multiples of P', never $\mathcal{O}$
- $\mathcal{O}$ occurs only after 2 consecutive doublings
- If $\mathcal{O}$ occurs during processing of bit k_i, bit k_{i+1} must be 0
- Uniquely identifies all 0 key bits (possibly except LSB)

Attack against a toy implementation

- Full domain correct
- Double and add scalar multiplication
- Input P with neighbour P' of order 4, inject fault and measure
- $k = 5405 = \cancel{1}010100011101_2$

i	11	10	9	8	7	6	5	4	3	2	1	0
k_i	0	1	0	1	0	0	0	1	1	1	0	1
$\boldsymbol{R}$	$2P'$	$\mathcal{O},P'$	$2P'$	$\mathcal{O},P'$	$2P'$	$\mathcal{O}$	$\mathcal{O}$	$\mathcal{O},P'$	$2P',3P'$	$2P',3P'$	$2P'$	$\mathcal{O},P'$

view	Op	$\mathcal{O}$	Op	Op	$\mathcal{O}$	Op	Op	$\mathcal{O}$	$\mathcal{O}$	$\mathcal{O}$	Op	Op	Op	Op	Op	Op	$\mathcal{O}$	Op
step 1	0			0			0	0	0							0		
step 2	0	1		0	1		0	0	0	1		1		1		0	1	

- Obtain all of k with a single trace!

Attack against a toy implementation

- Affine coordinates, partial domain correct (crash at 1st $\mathcal{O}$)
- Double and add scalar multiplication

- First occurrence of $\mathcal{O}$ leaks, then no more information
- For P' of order l, we obtain index $\mathrm{I}(l)$ s.t. the first $\mathrm{I}(l)$ bits of k form an integer divisible by l
  - Also information if not divisible by l

- Repeat with P' of increasing orders l
- Requires several traces with the same k
- Incremental search algorithm, obtain almost all of k

Feasibility of attack

- Need to be able to choose input P s.t. P' is of low order
  - El Gamal encryption/decryption, static Diffie-Hellman, etc.
- Or: system with fixed base point where P is already rogue
  - Nice back-door: impossible to check all error patterns

- Fault injection: need a specific error ε
  - ε is 1 bit-flip, 256 random byte faults, only ε leads to P' and $\mathcal{O}$
  - ε can be adjusted to any likely error pattern, in all coordinates
  - Precise timing

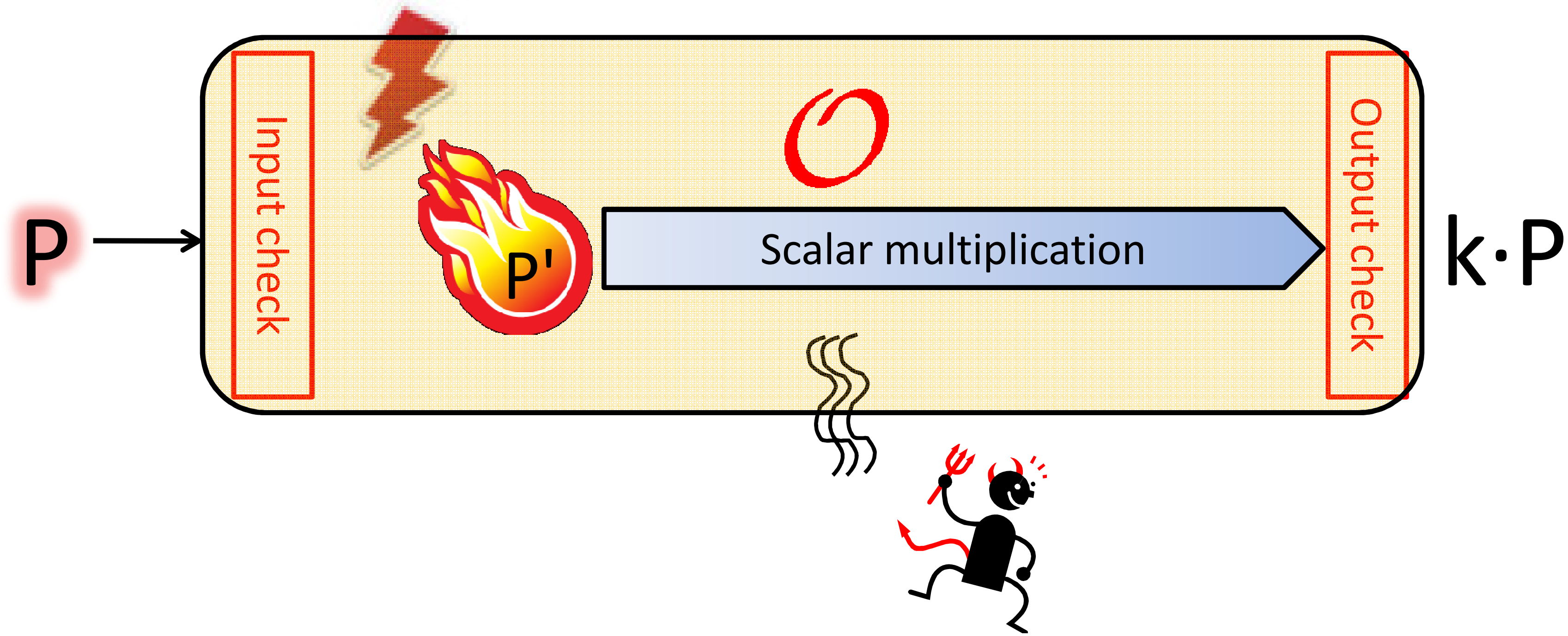
- Side channel: need leakage
  - We assume leakage by IFs, crashes, zero-value coordinates, etc.
- Group law implementation
  - Attack does not apply if all curve coefficients are used in PA/PD formulas

Attacks on scalar multiplication and countermeasures

- SPA
  - Solution: regular algorithm / implementation (atomicity)
- DPA
  - Solution: key, field, curve and point randomization
- Faults
  - Solution: check output point and curve parameter validity
- Low-order attack (weak curve attack)
  - Solution: check input point validity
  - Small co-factor check (all NIST curves have co-factor 1)

Attack against protected implementations

- Input point validity check
  - No problem if we can inject fault after check but before mult
- Output point / curve parameters validity check
  - No problem, we already got the info



Attack against protected implementations

- Regular exponentiation algorithms / implementations
  - Attack is fairly independent of scalar multiplication algorithm
  - Each algorithm computes some multiples of P that depend on k
  - If so, the attack applies

- Example: Montgomery ladder, 2 registers R_0 and $R_1=R_0 + P$
  - Input P with neighbour P' of order 4
  - If 2 consecutive key bits are equal, R_0 or R_1 doubled twice, $\mathcal{O}$ occurs
  - If 2 consecutive key bits are different, ordinary doublings
  - $\mathcal{O}$ can never be the result of an addition
- Obtain almost all of k with a single trace

More in the paper

- Countermeasures we looked at
  - Random scalar splitting: $k = k_1+k_2$, $k \cdot P = k_1 \cdot P + k_2 \cdot P$
  - Scalar blinding: $k' = k + r \cdot \#E$
  - Ephemeral keys
  - Coordinate randomization, e.g. random projective coord.
  - Random EC isomorphisms
  - Base point blinding
- Binary curves
  - Applicability of attack depends on coordinate system
  - Affine and standard projective coord.: attack applies since only a used
  - Jabobian: attack does not apply since a and b used
  - Lopez-Dahab: attack does not apply; only b is used but changing a results in isomorphic curve over its quadratic twist

Conclusion

- Our attack:
  - Input rogue P and inject fault after initial checks
  - P turns into P' of low order
  - $k \cdot P'$ leads to $\mathcal{O}$ which can be detected via side channels
- Requires chosen inputs (or rogue fixed base point)
- Very powerful attack on full domain correct implementations
  - Defeats many countermeasures, requires only a single trace
- Combining countermeasures does not automatically protect against combined attacks
- Countermeasures that prevent our attack:
  - Sensors, concurrent validity checks, base point blinding, etc.

Thank you. Questions?