Secure Implementation of ECDSA Signatures in Bitcoin

Di Wang

University College London

Supervised by: Dr. Nicolas T. Courtois

September 11, 2014

Overview

Motivation

- Bitcoin
- ECDSA: A main building block in Bitcoin
- ECDSA curve used: *secp256k1*

What Has Been Achieved

- ECDSA prototype
- Bitcoin transaction
- Performance improvement
- Side channel countermeasures

ECDSA Prototype

- Key generation: *secp256k1*
- Signature generation/verification
- Point multiplication: Double-and-add method (Affine Coordinates)

Algorithm 1 Double-and-add Algorithm

Require: d, P
Ensure: $Q = dP$
if $d_1 = 1$ **then**
  $Q := P$
end if
for $i = 1$ to n **do**
  $Q := 2Q$
  if $d_i = 1$ **then**
    $Q := Q + P$
  end if
end for
return Q

Performance Improvement

- Projective Coordinates (Avoids inversion)
- Window Method

Algorithm 2 Window Method Algorithm

Require: d, P
Ensure: $Q = dP$
if $d_1 > 0$ **then**
    $Q := d_1 P$
end if
for $i = 1$ to n **do**
    $Q := 2^w Q$ (Repeated doubling)
    if $d_i > 0$ **then**
        $Q := Q + d_i P$(Precomputed value $d_i P$)
    end if
end for
return Q

Side Channel Countermeasures

- Simple Power Analysis (SPA)
  1. Double-and-add-always method
  2. Montgomery ladder method
  3. Window-add-always method
- Differential Power Analysis (DPA)
  1. Scalar blinding
  2. Random scalar splitting

Side Channel Countermeasures

Simple Power Analysis

- Double-and-add-always Method

Algorithm 3 Double-and-add-always Algorithm

Require: d, P
Ensure: $Q = dP$
$Q[0] = \mathcal{O}$
$Q[1] = \mathcal{O}$
if $d_1 = 1$ **then**
  $Q[0] := P$
end if
for $i = 1$ to n **do**
  $Q[0] := 2Q[0]$
  $Q[1] := Q[0] + P$
  $Q[0] := 2Q[d_i]$
end for
return $Q[0]$

Side Channel Countermeasures

Simple Power Analysis

- Montgomery Ladder Method

Algorithm 4 Montgomery Ladder Algorithm

Require: d, P
Ensure: $Q = dP$
$Q[0] = \mathcal{O}$
$Q[1] = \mathcal{O}$
if $d_1 = 1$ **then**
    $Q[0] := P$
    $Q[1] := 2P$
end if
for $i = 1$ to n **do**
    $Q[1 - d_i] := Q[0] + Q[1]$
    $Q[d_i] := 2Q[d_i]$
end for
return $Q[0]$

Side Channel Countermeasures

Simple Power Analysis

- Window-add-always Method

Algorithm 5 Window-add-always Algorithm

Require: d, P
Ensure: $Q = dP$

$Q[0] = \mathcal{O}$
$Q[1] = \mathcal{O}$
if $d_1 > 0$ **then**
  $Q[0] := d_1 P$
end if
for $i = 1$ to n **do**
  $Q[0] := 2^w Q[0]$ (Repeated doubling)
  $Q[1] := Q[0] + d_i P$(Precomputed value $d_i P$)
  if $d_i > 0$ **then**
    $Q[0] = Q[1]$
  else
    $Q[0] = Q[0]$
  end if
end for
return $Q[0]$

Side Channel Countermeasures

Differential Power Analysis

To compute $Q = dP$:

- Scalar Blinding
  Choose a 20-bit random number k,
  Then $d' = d + k * \#N$.
  $Q = d'P = (d + k * \#N)P = dP$ since $\#NP = \mathcal{O}$.
- Random Scalar Splitting
  Choose a 200-bit random number r,
  Then $Q = dP = (d - r)P + rP$.

Four Versions

- Version 1: Double-and-add-always + Scalar Blinding
- Version 2: Window-add-always + Scalar Blinding
- Version 3: Montgomery Ladder + Random Scalar Splitting
- Version 4: Montgomery Ladder + Scalar Blinding

Speed Performance Results

	Prototype	Improved	Version 1	Version 2	Version 3	Version 4
#ADD	127	59	256	64	456	256
1ADD	28 us	17 us	17 us	19 us	17 us	20 us
#DBL	256	256	256	256	455	256
1DBL	32 us	15 us	15 us	16 us	13 us	15 us
Cal PM	11.75 ms	4.81 ms	8.20 ms	5.31 ms	13.67 ms	8.98 ms
1PM	12.64 ms	5.42 ms	9.36 ms	5.84 ms	15.12 ms	9.74 ms
SIGN	13.78 ms	6.53 ms	10.23 ms	7.13 ms	16.26 ms	11.39ms
1M	1.0 us	1.15 us	1.15 us	1.1 us	1.05 us	1.2 us
1S	——	1.15 us	1.05 us	1.15 us	1.1 us	1.05 us
1I	18.25 us	——	——	——	——	——

Affine Coordinates:
Prototype-Double-and-add
Projective Coordinates:
Improved-Window method
Version 1-Double-and-add-always + Scalar Blinding
Version 2-Window-add-always + Scalar Blinding
Version 3-Montgomery Ladder + Random Scalar Splitting
Version 4-Montgomery Ladder + Scalar Blinding

Security Evaluation

	Version 1	Version 2	Version 3	Version 4
SPA	Good	Poor	Good	Good
DPA	Fair	Fair	Good	Fair
FA	Poor	Poor	Fair	Fair
AUTHOR	Coron, J.S. [2]	Di Wang / Coron J.S.[2]	Montgomery, P. L. [3]/Ciet, M.& Joye, M. [1]	Montgomery, P. L.[3] / Coron, J.S.[2]
YEAR	1999	2014/1999	1987/2003	1987/1999
Version 1-Double-and-add-always + Scalar Blinding Version 2-Window-add-always + Scalar Blinding Version 3-Montgomery Ladder + Random Scalar Splitting Version 4-Montgomery Ladder + Scalar Blinding				

Further Work

- Coherence check [4]
- Security evaluation of overlap side channel countermeasures

References

Cite, M. and Joye, M. (2003)
(virtually) free randomization techniques for elliptic curve cryptography.
Information and Communications Security, pp. 348-359.

Coron, J.S. (1999)
Resistance against differential power analysis for elliptic curve cryptosystems.
Cryptographic Hardware and Embedded Systems, pp. 292-302.

Montgomery, P.L. (1987)
Speeding the pollard and elliptic curve methods of factorization.
Mathematics of computation, pp. 243-264.

Dominguez Oviedo, A. (2008)
On fault-based attacks and countermeasures for elliptic curve cryptosystems.

Thanks!