Redundant Modular Reduction Algorithms

Vincent DUPAQUIS
Inside Secure

Alexandre VENELLI
Inside Secure

CARDIS 2011
15/09/2011

DRIVING TRUST

Outline

1. Introduction
    1. Modular reduction
    2. Differential side-channel analysis
    3. Redundant modular arithmetic
2. Dynamic redundant Montgomery reduction algorithm
    1. Classical Montgomery
    2. Our proposed modification
3. Dynamic redundant Barrett reduction algorithm
    1. Classical Barrett
    2. Our proposed modification
4. Efficiency and security considerations
5. Conclusion

Outline

1. Introduction
   1. Modular reduction
   2. Differential side-channel analysis
   3. Redundant modular arithmetic
2. Dynamic redundant Montgomery reduction algorithm
3. Dynamic redundant Barrett reduction algorithm
4. Efficiency and security considerations
5. Conclusion

Modular reduction

- Modular reduction is used in Public Key Cryptography
  - RSA, Diffie-Hellman, ElGamal in $GF(p)$
  - Elliptic Curve Cryptography in $GF(p)$ and $GF(2^n)$
- Montgomery and Barrett are the most well-known
  - Pre-computational step
  - Trade costly multi-precision division for faster multi-precision multiplications
- Focus on RSA and modular exponentiation in particular

Differential Side-Channel Analysis

- Principle of DSCA
  - Find relationships between observed data and some key-related variable using statistical tests

- Classic DSCA countermeasures
  - Message blinding, exponent blinding, exponent splitting

- Example : Message blinding in RSA
  - Instead of computing $S = x^e \ mod \ m$
  - Let r a random, pre-compute $r' = (r^{-1})^e \ mod \ m$
  - Let $x' = rx \ mod \ m$
  - Compute $S' = x'^e mod \ m$
  - Correct result : $S = S'r' mod \ m$

Redundant modular arithmetic

- DSCA countermeasure
- Principle : Instead of working with integers modulo m, integers are kept modulo m plus some multiples of m
- Some propositions based on the idea
  - Time-constant Montgomery reduction (Walter 2002)
  - DSCA countermeasure for AES (Golic and Tymen 2002)
  - DSCA countermeasure in ECC (Smart et al. 2008)
- We extend this work by proposing modular reduction algorithms based on the classic Montgomery and Barrett reductions

Outline

1. Introduction
2. Dynamic redundant Montgomery reduction algorithm
   1. Classical Montgomery
   2. Our proposed modification
3. Dynamic redundant Barrett reduction algorithm
4. Efficiency and security considerations
5. Conclusion

Montgomery reduction algorithm (1)

- Pre-computed value :
  - $R > m$ coprime to m, e.g. $R = b^n$, and $\beta = -m^{-1} \bmod R$
- Integers are transformed into Montgomery form :
  - $u \rightarrow uR \bmod m$
  - $v \rightarrow vR \bmod m$
- Consider the multiplication $x = uvR^2$
- We want to reduce x modulo m

Montgomery reduction algorithm (2)

Algorithm 1 Montgomery reduction algorithm

Input: positive integers $x = (x_{2n-1}, \ldots, x_0)_b, m = (m_{n-1}, \ldots, m_0)_b$ and $\beta = -m^{-1} \bmod R$ where $R = b^n$, $\gcd(b, m) = 1$ and $x < mR$

Output: $xR^{-1} \bmod m$

1: $s_1 \leftarrow x \bmod R,\ s_2 \leftarrow \beta s_1 \bmod R,\ s_3 \leftarrow m s_2$
2: $t \leftarrow (x + s_3)/R$
3: **if** $(t \geq m)$ **then**
4: $\quad t \leftarrow t - m$
5: **end if**
6: **return** t

Dynamic redundant Montgomery reduction (1)

- Property of classic Montgomery reduction :
$$\frac{x+m(x\beta \ mod \ R)}{R} = (xR^{-1} \ mod \ m) + \epsilon m$$ with $\epsilon \in \{0,1\}$

- Now consider the following steps :
  1. $s_1 \leftarrow x \ mod \ R$
  2. $s_2 \leftarrow \beta s_1 \ mod \ R$
  3. $s_2 \leftarrow s_2 + kR$, with k some random integer
  4. $s_3 \leftarrow m s_2$
  5. $t \leftarrow (x + s_3)/R$

- Hence at the end of the reduction
$$(xR^{-1} \ mod \ m) + km \leq t \leq (xR^{-1} \ mod \ m) + (k+1)m$$

Dynamic redundant Montgomery reduction (2)

- Added modulus → output of the reduction bigger in size → problem to further reduce it

- Solution : modify the pre-computed values of Montgomery to process bigger integers

- Instead of the classical $R = b^n$, we use $R' = b^{n+2i}$ and consider integers $x < mR' < b^{2n+2i}$

- Hence the output of the reduction can be integers $t < b^{n+i}$

- Hence the added random k should be $k < b^i - 1$

Outline

1. Introduction
2. Dynamic redundant Montgomery reduction algorithm
3. Dynamic redundant Barrett reduction algorithm
    1. Classical Barrett
    2. Our proposed modification
4. Efficiency and security considerations
5. Conclusion

Barrett reduction algorithm (1)

- Pre-computed value :
  - $\mu = \left\lfloor \frac{b^n}{m} \right\rfloor$

- Integers u and v are not transformed

- Consider the multiplication $x = uv$

- We want to reduce x modulo m

Barrett reduction algorithm (2)

Algorithm 2 Barrett reduction algorithm

Input: positive integers $x = (x_{2n-1}, \ldots, x_0)_b, m = (m_{n-1}, \ldots, m_0)_b$ and $\mu = \lfloor b^{2n}/m \rfloor$

Output: $x \bmod m$

$q_1 \leftarrow \lfloor x/b^{n-1} \rfloor, \ q_2 \leftarrow \mu q_1, \ q_3 \leftarrow \lfloor q_2/b^{n+1} \rfloor$

$r_1 \leftarrow x \bmod b^{n+1}, \ r_2 \leftarrow m q_3 \bmod b^{n+1}, \ r \leftarrow r_1 - r_2$

if $(r \leq 0)$ **then**

  $r \leftarrow r + b^{n+1}$

end if

while $(r \geq m)$ **do**

  $r \leftarrow r - m$

end while

return r

Dynamic redundant Barrett reduction (1)

- Property of classic Barrett reduction : $(x \; mod \; m) + \epsilon m$ with $\epsilon \in \{0, 2\}$
- Estimated quotient : $\hat{q} = \left\lfloor \frac{\frac{x}{b^{n+\beta}} \mu_\alpha}{b^{\alpha-\beta}} \right\rfloor$ with $\mu_\alpha = \left\lfloor \frac{b^{n+\alpha}}{m} \right\rfloor$ for α, β integers
- Bounds on the error from Dhem's work not applicable as maximal error is rarely reached
- We can undervalue the estimated quotient to add multiples of the modulus

Dynamic redundant Barrett reduction (2)

- Consider the following steps
  1. $q_1 \leftarrow \lfloor \frac{x}{b^{n+\beta}} \rfloor$
  2. $q_2 \leftarrow \mu_\alpha q_1$
  3. $q_3 \leftarrow \lfloor \frac{q_2}{b^{\alpha-\beta}} \rfloor$
  4. $q_3 \leftarrow q_3 - k$, with k some random integer
  5. $r_1 \leftarrow x \ mod \ b^\alpha$
  6. $r_2 \leftarrow mq_3 \ mod \ b^\alpha$
  7. $r \leftarrow r_1 - r_2$

Dynamic redundant Barrett reduction (3)

- We choose $\alpha = n + 2i$ and $\beta = -1$ → $\hat{q}$ undervalued by 2

- Hence at the end of the reduction

$$(x \; mod \; m) + km \leq r \leq (x \; mod \; m) + (k+2)m$$

- Larger pre-computed constant to process bigger integers

$$\mu' = \mu_{n+2i} = \left\lfloor \frac{b^{2n+2i}}{m} \right\rfloor$$

- The added random k is bounded by $k < b^i - 2$

Outline

1. Introduction
2. Dynamic redundant Montgomery reduction algorithm
3. Dynamic redundant Barrett reduction algorithm
4. Efficiency and security considerations
5. Conclusion

Efficiency

Algorithm	Time (in ms)
Standard Montgomery	6.1 or 6.3
Dynamic redundant Montgomery with $i = 1$	8.7
Dynamic redundant Montgomery with $i = 2$	9.3
Standard Barrett	6.4 or 6.6
Dynamic redundant Barrett with $i = 1$	6.3
Dynamic redundant Barrett with $i = 2$	6.6

Example of application in a modular exponentiation

Algorithm 6 Multiply always exponentiation using dynamic redundant Montgomery arithmetic

Input: positive integers $e = (e_{l-1}, \ldots, e_0)_2, x, m, \beta'$ and R'. Let $\mathsf{rand}()$ be a function that generates a random integer in $[0, b^i - 1[$ for some integer i.

Output: $x^e \bmod m$

$X \leftarrow x + \mathsf{rand}()m$
$R_0 \leftarrow \mathsf{DRMontRed}(\mathsf{rand}()m, m, R', \beta')$
$R_1 \leftarrow \mathsf{DRMontRed}(XR', m, R', \beta')$
$i \leftarrow l - 1,\ t \leftarrow 0$
while $i \geq 0$ **do**
    $R_0 \leftarrow \mathsf{DRMontRed}(R_0(R_t + \mathsf{rand}()m), m, R', \beta')$
    $t \leftarrow t \oplus e_i,\ i \leftarrow i - 1 + t$
end while
$R_0 \leftarrow \mathsf{DRMontRed}(R_0 R'^{-1}, m, R', \beta')$
$R_0 \leftarrow \mathsf{Normalize}(R_0, m)$
return R_0

Resistance to side-channel attacks

- Resistance to classical DSCA
- Classical *multiply-always* vulnerable to Amiel et al. 2008 attack
- Left-to-right atomic algorithms seem particularly vulnerable to combined attacks (passive + active) by Amiel et al. 2007

Note on elliptic curve cryptography

- ☹ NIST curves using GM primes
- ☺ Brainpool curves or others randomly generated elliptic curves
- Dynamic redundant arithmetic can hide the infinity point from SPA
- → Protection against Goubin’s attack and even the recent combined attack on ECC of Fan et al. 2011

Outline

1. Introduction
2. Dynamic redundant Montgomery reduction algorithm
3. Dynamic redundant Barrett reduction algorithm
4. Efficiency and security considerations
5. Conclusion

Conclusion

- Our modular reduction propositions are
  - parametrized,
  - time constant,
  - efficient
- Dynamic randomization for a small overhead
- Protection against DSCA and more refined attacks like Amiel et al. 2008 or recent combined attacks

Thank you for your attention !