The Montgomery Powering Ladder

Marc Joye

Gemplus Card International
Gémenos, France
marc.joye@gemplus.com
http://www.geocities.com/MarcJoye/

Sung-Ming Yen[*]

LCIS, National Central University
Chung-Li, Taiwan
yensm@csie.ncu.edu.tw
http://www.csie.ncu.edu.tw/∼yensm/

CHES 2002, San Francisco Bay, August 13–15, 2002

http://www.gemplus.com/smart/

GEMPLUS

Agenda

- Montgomery Powering Ladder
- Efficiency Analysis
- Security Analysis
- Conclusion

Agenda

- Montgomery Powering Ladder
- Efficiency Analysis
- Security Analysis
- Conclusion

Montgomery Powering Ladder

- Goal: given $k = \sum_{i=0}^{t-1} k_i\, 2^i$, compute $y = g^k$ in $\mathbb{G}$

Montgomery Powering Ladder

- Goal: given $k = \sum_{i=0}^{t-1} k_i\, 2^i$, compute $y = g^k$ in $\mathbb{G}$
- Key observation
  - define $L_j = \sum_{i=j}^{t-1} k_i\, 2^{i-j}$ and $H_j = L_j + 1$
  - $L_j = 2L_{j+1} + k_j = L_{j+1} + H_{j+1} + k_j - 1$

Montgomery Powering Ladder

- Goal: given $k = \sum_{i=0}^{t-1} k_i\, 2^i$, compute $y = g^k$ in $\mathbb{G}$
- Key observation
  - define $L_j = \sum_{i=j}^{t-1} k_i\, 2^{i-j}$ and $H_j = L_j + 1$
  - $L_j = 2L_{j+1} + k_j = L_{j+1} + H_{j+1} + k_j - 1$
- Montgomery powering ladder: $k = L_0$

Montgomery Powering Ladder

- Goal: given $k = \sum_{i=0}^{t-1} k_i\, 2^i$, compute $y = g^k$ in $\mathbb{G}$
- Key observation
  - define $L_j = \sum_{i=j}^{t-1} k_i\, 2^{i-j}$ and $H_j = L_j + 1$
  - $L_j = 2L_{j+1} + k_j = L_{j+1} + H_{j+1} + k_j - 1$
- Montgomery powering ladder: $k = L_0$
  - $(L_j, H_j) = \begin{cases} (2L_{j+1}, L_{j+1} + H_{j+1}) & \text{if } k_j = 0 \\ (L_{j+1} + H_{j+1}, 2H_{j+1}) & \text{if } k_j = 1 \end{cases}$

Montgomery Powering Ladder

- Goal: given $k = \sum_{i=0}^{t-1} k_i\, 2^i$, compute $y = g^k$ in $\mathbb{G}$
- Key observation
  - define $L_j = \sum_{i=j}^{t-1} k_i\, 2^{i-j}$ and $H_j = L_j + 1$
  - $L_j = 2L_{j+1} + k_j = L_{j+1} + H_{j+1} + k_j - 1$
- Montgomery powering ladder: $k = L_0$
  - $$(L_j, H_j) = \begin{cases} (2L_{j+1}, L_{j+1} + H_{j+1}) & \text{if } k_j = 0 \\ (L_{j+1} + H_{j+1}, 2H_{j+1}) & \text{if } k_j = 1 \end{cases}$$

Montgomery Powering Ladder

- Goal: given $k = \sum_{i=0}^{t-1} k_i \, 2^i$, compute $y = g^k$ in $\mathbb{G}$
- Key observation
  - define $L_j = \sum_{i=j}^{t-1} k_i \, 2^{i-j}$ and $H_j = L_j + 1$
  - $L_j = 2L_{j+1} + k_j = L_{j+1} + H_{j+1} + k_j - 1$
- Montgomery powering ladder: $k = L_0$
  - $(L_j, H_j) = \begin{cases} (2L_{j+1}, L_{j+1} + H_{j+1}) & \text{if } k_j = 0 \\ (L_{j+1} + H_{j+1}, 2H_{j+1}) & \text{if } k_j = 1 \end{cases}$

Montgomery Powering Ladder

- Goal: given $k = \sum_{i=0}^{t-1} k_i\, 2^i$, compute $y = g^k$ in $\mathbb{G}$
- Key observation
  - define $L_j = \sum_{i=j}^{t-1} k_i\, 2^{i-j}$ and $H_j = L_j + 1$
  - $L_j = 2L_{j+1} + k_j = L_{j+1} + H_{j+1} + k_j - 1$
- Montgomery powering ladder: $k = L_0$
  - $(L_j, H_j) = \begin{cases} (2L_{j+1}, L_{j+1} + H_{j+1}) & \text{if } k_j = 0 \\ (L_{j+1} + H_{j+1}, 2H_{j+1}) & \text{if } k_j = 1 \end{cases}$
  - $(g^{L_j}, g^{H_j}) = \begin{cases} ((g^{L_{j+1}})^2, g^{L_{j+1}} \cdot g^{H_{j+1}}) & \text{if } k_j = 0 \\ (g^{L_{j+1}} \cdot g^{H_{j+1}}, (g^{H_{j+1}})^2) & \text{if } k_j = 1 \end{cases}$

Montgomery Powering Ladder

- The algorithm

`Input:` $g, k = (k_{t-1}, \ldots, k_0)_2$

`Output:` $y = g^k$

$R_0 \leftarrow 1$; $R_1 \leftarrow g$

`for` $j = t-1$ `downto` 0 `do`

  `if` $(k_j = 0)$ `then`

    $R_1 \leftarrow R_0\, R_1$; $R_0 \leftarrow (R_0)^2$

  `else [if` $(k_j = 1)$`]`

    $\boxed{R_0 \leftarrow R_0\, R_1;\ R_1 \leftarrow (R_1)^2}$

`return` R_0

$g^{L_j} = g^{L_{j+1}} \cdot g^{H_{j+1}}$

$[R_0 \leftarrow R_0\, R_1]$

$g^{H_j} = (g^{H_{j+1}})^2$

$[R_1 \leftarrow (R_1)^2]$

Montgomery Powering Ladder

- The algorithm

Input: $g, k = (k_{t-1}, \ldots, k_0)_2$

Output: $y = g^k$

$R_0 \leftarrow 1$; $R_1 \leftarrow g$

for $j = t-1$ downto 0 do

  if $(k_j = 0)$ then

    $\boxed{R_1 \leftarrow R_0\, R_1;\ \ R_0 \leftarrow (R_0)^2}$

  else [if $(k_j = 1)$]

    $R_0 \leftarrow R_0\, R_1$; $R_1 \leftarrow (R_1)^2$

return R_0

$$g^{L_j} = (g^{L_{j+1}})^2$$

$$[R_0 \leftarrow (R_0)^2]$$

$$g^{H_j} = g^{L_{j+1}} \cdot g^{H_{j+1}}$$

$$[R_1 \leftarrow R_0\, R_1]$$

Agenda

- Montgomery Powering Ladder
- Efficiency Analysis
- Security Analysis
- Conclusion

Efficiency Analysis

- Lucas chains structure

Efficiency Analysis

- Lucas chains structure
  - R_1/R_0 is invariant $[R_1/R_0 = g]$

Efficiency Analysis

- Lucas chains structure
  - R_1/R_0 is invariant $[R_1/R_0 = g]$

```
R0 ← 1;  R1 ← g
for j = t − 1 downto 0 do
  if (kj = 0) then
    R1 ← R0 R1;  R0 ← (R0)^2
  else [if (kj = 1)]
    R0 ← R0 R1;  R1 ← (R1)^2
return R0
```

$$\boxed{R_1 \leftarrow R_0\,R_1;\ \ R_0 \leftarrow (R_0)^2}$$

$$\frac{R_1}{R_0} \leftarrow \frac{R_0\,R_1}{(R_0)^2} = \frac{R_1}{R_0}$$

Efficiency Analysis

- Lucas chains structure
  - R_1/R_0 is invariant $[R_1/R_0 = g]$

```
R0 ← 1;  R1 ← g
for j = t − 1 downto 0 do
  if (k_j = 0) then
    R1 ← R0 R1;  R0 ← (R0)^2
  else [if (k_j = 1)]
    R0 ← R0 R1;  R1 ← (R1)^2
return R0
```

$$\frac{R_1}{R_0} \leftarrow \frac{(R_1)^2}{R_0\, R_1} = \frac{R_1}{R_0}$$

Efficiency Analysis

- Lucas chains structure
  - R_1/R_0 is invariant $[R_1/R_0 = g]$
  - $\mathbb{G} =$ elliptic curve over a field $\mathbb{K}$

Efficiency Analysis

- Lucas chains structure
  - R_1/R_0 is invariant $[R_1/R_0 = g]$
  - $\mathbb{G} =$ elliptic curve over a field $\mathbb{K}$
    - ✓ computations can be carried out with x-coordinate only

Efficiency Analysis

- Lucas chains structure
  - R_1/R_0 is invariant $[R_1/R_0 = g]$
  - $\mathbb{G} =$ elliptic curve over a field $\mathbb{K}$
    - ✓ computations can be carried out with x-coordinate only
    - ✓ the y-coordinates need not to be handled
      - a lot of multiplications (in $\mathbb{K}$) are saved
      - fewer memory is required

Efficiency Analysis

- Lucas chains structure
  - R_1/R_0 is invariant $[R_1/R_0 = g]$
  - $\mathbb{G} =$ elliptic curve over a field $\mathbb{K}$
    - ✓ computations can be carried out with x-coordinate only
    - ✓ the y-coordinates need not to be handled
      - a lot of multiplications (in $\mathbb{K}$) are saved
      - fewer memory is required
  - similarly for "full" Lucas sequences
    - ✓ computations can be carried out with the V-sequence only

Efficiency Analysis

- Parallel computing

Efficiency Analysis

- Parallel computing
  - simplified presentation

$$R_0 \leftarrow 1\texttt{;}\quad R_1 \leftarrow g$$

$\texttt{for}\ \ j = t-1\ \ \texttt{downto}\ \ 0\ \ \texttt{do}$

$R_{\neg k_j} \leftarrow R_0\, R_1$

$\quad\texttt{if}\ \ (k_j = 0)\ \ \texttt{then}$

$\quad\quad \boxed{R_1 \leftarrow R_0\, R_1}\texttt{;}\quad R_0 \leftarrow (R_0)^2$

$\quad\texttt{else [if}\ \ (k_j = 1)\texttt{]}$

$\quad\quad \boxed{R_0 \leftarrow R_0\, R_1}\texttt{;}\quad R_1 \leftarrow (R_1)^2$

$\texttt{return}\ \ R_0$

Efficiency Analysis

- Parallel computing
  - simplified presentation

$$R_0 \leftarrow 1;\ \ R_1 \leftarrow g$$

$\texttt{for}\ j = t-1\ \texttt{downto}\ 0\ \texttt{do}$

$\quad\texttt{if}\ (k_j = 0)\ \texttt{then}$ $\qquad R_{k_j} \leftarrow (R_{k_j})^2$

$\qquad R_1 \leftarrow R_0\, R_1;$ $\boxed{R_0 \leftarrow (R_0)^2}$

$\quad\texttt{else [if}\ (k_j = 1)\texttt{]}$

$\qquad R_0 \leftarrow R_0\, R_1;$ $\boxed{R_1 \leftarrow (R_1)^2}$

$\texttt{return}\ R_0$

Efficiency Analysis

- Parallel computing
  - simplified presentation

$R_0 \leftarrow 1$; $R_1 \leftarrow g$

for $j = t - 1$ downto 0 do

$\quad R_{\neg k_j} \leftarrow R_0 \, R_1$

$\quad R_{k_j} \leftarrow (R_{k_j})^2$

return R_0

Efficiency Analysis

- Parallel computing
  - simplified presentation

$$R_0 \leftarrow 1;\ \ R_1 \leftarrow g$$

$\texttt{for}\ j = t-1\ \texttt{downto}\ 0\ \texttt{do}$

$$R_{\neg k_j} \leftarrow R_0\, R_1$$

$$R_{k_j} \leftarrow (R_{k_j})^2$$

$\texttt{return}\ R_0$

  - the 2 multiplications are independent

Efficiency Analysis

- Parallel computing
  - parallel Montgomery ladder

$R_0 \leftarrow 1$; $R_1 \leftarrow g$

for $j = t - 1$ downto 0 do

$\quad R_{\neg k_j} \leftarrow R_0\,R_1$ /* Processor 1 */

$\quad R_{k_j} \leftarrow (R_{k_j})^2$ /* Processor 2 */

return R_0

Efficiency Analysis

- Parallel computing
  - parallel Montgomery ladder

```
R_0 ← 1;  R_1 ← g
for j = t−1 downto 0 do
   R_{¬k_j} ← R_0 R_1      /* Processor 1 */
   R_{k_j} ← (R_{k_j})^2   /* Processor 2 */
return R_0
```

$R_0 \leftarrow 1$; $R_1 \leftarrow g$

for $j = t-1$ downto 0 do

$\quad R_{\neg k_j} \leftarrow R_0\, R_1$ /* Processor 1 */

$\quad R_{k_j} \leftarrow (R_{k_j})^2$ /* Processor 2 */

return R_0

  - twice faster with 2 processors

Efficiency Analysis

- Common-multiplicand property

Efficiency Analysis

- Common-multiplicand property
  - R_0 (resp. R_1) is common to the 2 multiplications

$R_0 \leftarrow 1$; $R_1 \leftarrow g$

for $j = t-1$ downto 0 do

  if $(k_j = 0)$ then

    $\boxed{R_1 \leftarrow R_0\, R_1;\ \ R_0 \leftarrow R_0\, R_0}$

  else [if $(k_j = 1)$]

    $R_0 \leftarrow R_0\, R_1$; $R_1 \leftarrow R_1\, R_1$

return R_0

Efficiency Analysis

- Common-multiplicand property
  - R_0 (resp. R_1) is common to the 2 multiplications

```
R0 ← 1;  R1 ← g
for j = t-1 downto 0 do
  if (k_j = 0) then
    R1 ← R0 R1;  R0 ← R0 R0
  else [if (k_j = 1)]
    R0 ← R0 R1;  R1 ← R1 R1
return R0
```

Efficiency Analysis

- Common-multiplicand property
  - R_0 (resp. R_1) is common to the 2 multiplications
  - $\mathbb{G} = \mathbb{Z}_N$
    - ✓ the CM-multiplication by Yen is applicable

Efficiency Analysis

- Common-multiplicand property
  - R_0 (resp. R_1) is common to the 2 multiplications
  - $\mathbb{G} = \mathbb{Z}_N$
    - ✓ the CM-multiplication by Yen is applicable
  - similar savings for more “complicated” groups
    - ✓ e.g., when $\mathbb{G} =$ elliptic curve over $\mathbb{K}$, several multiplications (in $\mathbb{K}$) are identical

Agenda

- Montgomery Powering Ladder
- Efficiency Analysis
- Security Analysis
- Conclusion

Security Analysis

- Side-channel attacks

Security Analysis

- Side-channel attacks
  - SPA-like attacks
    - ✓ Montgomery ladder behaves regularly whatever the scanned bit, k_j

Security Analysis

- Side-channel attacks
  - SPA-like attacks
    - ✓ Montgomery ladder behaves regularly whatever the scanned bit, k_j

$$
\begin{array}{l}
R_0 \leftarrow 1;\ \ R_1 \leftarrow g \\
\texttt{for } j = t-1 \texttt{ downto } 0 \texttt{ do} \\
\quad R_{\neg k_j} \leftarrow R_0\, R_1;\ \ R_{k_j} \leftarrow (R_{k_j})^2 \\
\texttt{return } R_0
\end{array}
$$

Security Analysis

- Side-channel attacks
  - SPA-like attacks
    - ✓ Montgomery ladder behaves regularly whatever the scanned bit, k_j
    - ✓ SPA-resistant, provided that
      - writing in R_0 is indistinguishable from writing in R_1
      - squaring of R_0 is indistinguishable from squaring of R_1

Security Analysis

- Side-channel attacks
  - SPA-like attacks
    - ✓ Montgomery ladder behaves regularly whatever the scanned bit, k_j
    - ✓ SPA-resistant, provided that
      - writing in R_0 is indistinguishable from writing in R_1
      - squaring of R_0 is indistinguishable from squaring of R_1
  - DPA-like attacks
    - ✓ prevented using standard blinding techniques

Security Analysis

- Fault attacks
  - C safe-error attacks

Security Analysis

- Fault attacks
  - C safe-error attacks
    - principle: timely induce a computational fault into the ALU for determining whether an operation is
      - dummy (when the final result is correct), or
      - effective (when the final result is incorrect)

Security Analysis

- Fault attacks
  - C safe-error attacks
    - principle: timely induce a computational fault into the ALU for determining whether an operation is
      - dummy (when the final result is correct), or
      - effective (when the final result is incorrect)
    - this reveals bit-by-bit the value of exponent k in the classical protected binary ladders:
      - the square-and-multiply *always* algorithm, and
      - its right-to-left counterpart

Security Analysis

- Fault attacks
  - C safe-error attacks
    - ✓ there are no dummy operations (mult.) in the Montgomery ladder

$$R_0 \leftarrow 1\texttt{;}\quad R_1 \leftarrow g$$

$$\texttt{for}\ \ j = t-1\ \ \texttt{downto}\ \ 0\ \ \texttt{do}$$

$$\quad R_{\neg k_j} \leftarrow R_0\, R_1\texttt{;}\quad R_{k_j} \leftarrow (R_{k_j})^2$$

$$\texttt{return}\ \ R_0$$

Security Analysis

- Fault attacks
  - C safe-error attacks
    - ✓ there are no dummy operations (mult.) in the Montgomery ladder
    - ✓ the C safe-error model does not apply

Security Analysis

- Fault attacks
  - M safe-error attacks

Security Analysis

- Fault attacks
  - M safe-error attacks
    - principle: timely induce a memory fault inside register R_1 during the evaluation of

$$R_b \leftarrow R_0\, R_1$$

for determining whether the result is written in
- R_1 (when the final result is correct), or
- R_0 (when the final result is incorrect)

Security Analysis

- Fault attacks
  - M safe-error attacks
    - ✓ principle: timely induce a memory fault inside register R_1 during the evaluation of

$$R_b \leftarrow R_0 \, R_1$$

      for determining whether the result is written in
      - R_1 (when the final result is correct), or
      - R_0 (when the final result is incorrect)
    - ✓ this attack readily applies to Montgomery ladder

Security Analysis

- Fault attacks
  - M safe-error attacks
    - principle: timely induce a memory fault inside register R_1 during the evaluation of

$$R_b \leftarrow R_0\, R_1$$

      for determining whether the result is written in
      - R_1 (when the final result is correct), or
      - R_0 (when the final result is incorrect)
    - this attack readily applies to Montgomery ladder
    - BUT a slight modification makes the attack inapplicable

Security Analysis

- Fault attacks
  - M safe-error attacks
    - ✓ original Montgomery ladder

$$
\begin{array}{l}
R_0 \leftarrow 1\texttt{;}\;\; R_1 \leftarrow g \\
\texttt{for}\;\; j = t-1 \;\;\texttt{downto}\;\; 0 \;\;\texttt{do} \\
\quad R_{\neg k_j} \leftarrow R_0\, R_1\texttt{;}\;\; R_{k_j} \leftarrow (R_{k_j})^2 \\
\texttt{return}\;\; R_0
\end{array}
$$

Security Analysis

- Fault attacks
  - M safe-error attacks
    - ✓ modified Montgomery ladder

$R_0 \leftarrow 1$; $R_1 \leftarrow g$

for $j = t - 1$ downto 0 do

$\quad R_{\neg k_j} \leftarrow R_{\neg k_j} R_{k_j}$; $R_{k_j} \leftarrow (R_{k_j})^2$

return R_0

Security Analysis

- Fault attacks
  - M safe-error attacks
    - ✓ modified Montgomery ladder

$$R_0 \leftarrow 1;\ \ R_1 \leftarrow g$$

$$\texttt{for}\ \ j = t-1\ \ \texttt{downto}\ \ 0\ \ \texttt{do}$$

$$\quad R_{\neg k_j} \leftarrow R_{\neg k_j}\, R_{k_j}\texttt{;}\ \ R_{k_j} \leftarrow (R_{k_j})^2$$

$$\texttt{return}\ \ R_0$$

    - ✓ the M safe-error model does no longer apply

Agenda

- Montgomery Powering Ladder
- Efficiency Analysis
- Security Analysis
- Conclusion

Conclusion

- Efficiency
  - Lucas chains structure
  - parallel computing
  - common-multiplicand property

Conclusion

- Efficiency
  - Lucas chains structure
  - parallel computing
  - common-multiplicand property
- Security
  - against SPA-like attacks
  - against C safe-error attacks
  - against M safe-error attacks
    - ✓ after modification

Conclusion

- Efficiency
  - Lucas chains structure
  - parallel computing
  - common-multiplicand property
- Security
  - against SPA-like attacks
  - against C safe-error attacks
  - against M safe-error attacks
    - ✓ after modification
- Montgomery ladder is well suited for efficient and secure exponentiation (in $\mathbb{G}$) in constrained devices