Some Security Aspects of the MIST Randomized Exponentiation Algorithm

Colin D. Walter

www.comodo.net (Bradford, UK)
colin.walter@comodo.net
Power Analysis Attacks

• With no counter-measures and the binary $\text{exp}^n \text{alg}^m$, averaging power traces at the same instants during several $\text{exp}^n$s enables one to differentiate squares and multiplies and hence deduce the exponent bits (Kocher).

• Averaging power traces over individual digit-by-digit products in a single $\text{exp}^n$ enables one to differentiate multiplicands in $m$-ary $\text{exp}^n$ and hence deduce the exponent (CHES 2001).

• Smartcards have limited scope for including expensive, tamper-resistant, hardware measures.

• Good software counter-measures are required: new algorithms as well as modifying arguments e.g. $D$ to $D+r\phi(N)$. 
$m$-ary Exp$^n$ (Reversed)

\{ To compute: $P = C^D$ \}

$Q \leftarrow C$ ;
$P \leftarrow 1$ ;
While $D > 0$ do
Begin

\[ d \leftarrow D \mod m ; \]
If $d \neq 0$ then
\[ P \leftarrow Q^d \times P ; \]
$Q \leftarrow Q^m$ ;
$D \leftarrow D \div m ;$
\{ Invariant: $C^{D_{Init}} = Q^D \times P$ \}
End
The **MIST** Exp\(n\) Algorithm

\[
\{ \text{To compute: } P = C^D \}
\]

\(Q \leftarrow C\);
\(P \leftarrow 1\);
\textbf{While} \(D > 0\) \textbf{do}

\textbf{Begin}

\(d \leftarrow D \mod m\);
\textbf{If} \(d \neq 0\) \textbf{then}

\(P \leftarrow Q^d \times P\);
\(Q \leftarrow Q^m\);
\(D \leftarrow D \div m\);

\{ Invariant: \(C^D.\text{Init} = Q^D \times P\) \}

\textbf{End}
Randomary Exponentiation

The main computational part of the loop is:

\[
\text{If } d \neq 0 \text{ then} \\
P \leftarrow Q^d \times P ; \\
Q \leftarrow Q^m
\]

- To provide the required efficiency, a set of possible values for \( m \) are chosen so that an efficient addition chain for \( m \) contains \( d \), e.g.

\[
1+1=2, \quad 2+1=3, \quad 2+3=5 \quad \text{is an addition chain for base } m=5 \text{ suitable for digits } d = 0, 1, 2 \text{ or } 3.
\]

- Comparable to the 4-ary method regarding time complexity.
Running Example

Fix the base set = \{2, 3, 5\}. Consider \( D = 235 \)

<table>
<thead>
<tr>
<th>( D )</th>
<th>( m, d )</th>
<th>( Q ) (before)</th>
<th>( Q^d )</th>
<th>( Q^m )</th>
<th>( P ) (after)</th>
</tr>
</thead>
<tbody>
<tr>
<td>235</td>
<td>3, 1</td>
<td>( C^1 )</td>
<td>( C^1 )</td>
<td>( C^3 )</td>
<td>( C^1 )</td>
</tr>
<tr>
<td>78</td>
<td>2, 0</td>
<td>( C^3 )</td>
<td>1</td>
<td>( C^6 )</td>
<td>( C^1 )</td>
</tr>
<tr>
<td>39</td>
<td>5, 4</td>
<td>( C^6 )</td>
<td>( C^{24} )</td>
<td>( C^{30} )</td>
<td>( C^{1}\times C^{24} = C^{25} )</td>
</tr>
<tr>
<td>7</td>
<td>2, 1</td>
<td>( C^{30} )</td>
<td>( C^{30} )</td>
<td>( C^{60} )</td>
<td>( C^{25}\times C^{30} = C^{55} )</td>
</tr>
<tr>
<td>3</td>
<td>3, 0</td>
<td>( C^{60} )</td>
<td>1</td>
<td>( C^{180} )</td>
<td>( C^{55} )</td>
</tr>
<tr>
<td>1</td>
<td>2, 1</td>
<td>( C^{180} )</td>
<td>( C^{180} )</td>
<td>( C^{360} )</td>
<td>( C^{55}\times C^{180} = C^{235} )</td>
</tr>
</tbody>
</table>
Choice of Base Set

- **Security**: Bases must be chosen so that sequences of squares & multiplies or \( op^d \) sharing do not reveal \( m \).

- **Efficiency**:
  - Bases \( m \) must be chosen so that raising to the power \( m \) is (time) efficient enough.
  - Space is required to store addition chains.
  - As few registers as possible should be used for the exponentiation.

- **One Solution**: Take the set of bases \( \{2,3,5\} \).
Choice of Base

**Example algorithm (see CT-RSA 2002 paper):**

\[
m \leftarrow 0 ;
\]

**If** Random(8) < 7 **then**

  **If** \( D \mod 2 \) = 0 **then** \( m \leftarrow 2 \) **else**
  **If** \( D \mod 5 \) = 0 **then** \( m \leftarrow 5 \) **else**
  **If** \( D \mod 3 \) = 0 **then** \( m \leftarrow 3 \);

**If** \( m = 0 \) **then**

**Begin**

  \( p \leftarrow \text{Random}(8) ; \)
  **If** \( p < 6 \) **then** \( m \leftarrow 2 \) **else**
  **If** \( p < 7 \) **then** \( m \leftarrow 5 \) **else**
  \( m \leftarrow 3 \)

**End**
Probability of \((m,d)\)

- Define probabilities:

  \[ p_i = \text{prob}(D \equiv i \mod 30) \]
  \[ p_{m|i} = \text{prob}(\text{choosing } m \text{ given } D \equiv i \mod 30) \]

- Then:

  \[ p_m = \sum_{i \mod 30} p_i p_{m|i} \quad \text{is prob of base } m \]
  \[ p_{m,d} = \sum_{i \equiv d \mod 30} p_i p_{m|i} \quad \text{is prob of pair } (m,d) \]

- For the base selection process above:

  \[ p_2 = 0.629 \quad p_3 = 0.228 \quad p_5 = 0.142 \]
Addition Sub-Chains

- Let \((ijk)\) mean: multiply contents at addresses \(i\) and \(j\) and write result to address \(k\).
- Use 1 for location of \(Q\), 2 for temporary register, 3 for \(P\):

\[
\begin{align*}
(111) & \quad \text{for } (m,d) = (2,0) \\
(112, 133) & \quad \text{for } (m,d) = (2,1) \\
(112, 121) & \quad \text{for } (m,d) = (3,0) \\
(112, 133, 121) & \quad \text{for } (m,d) = (3,1) \\
(112, 233, 121) & \quad \text{for } (m,d) = (3,2) \\
(112, 121, 121) & \quad \text{for } (m,d) = (5,0) \\
(112, 133, 121, 121) & \quad \text{for } (m,d) = (5,1) \\
(112, 233, 121, 121) & \quad \text{for } (m,d) = (5,2) \\
(112, 121, 133, 121) & \quad \text{for } (m,d) = (5,3) \\
(112, 222, 233, 121) & \quad \text{for } (m,d) = (5,4)
\end{align*}
\]
S&M Sequences

• Assume an attacker can distinguish Squares and Multiplies from a single exponentiation (e.g. from Hamming weights of arguments deduced from power variation on bus.)

• A division chain is the list of pairs \((m,d)\) used in an \(\exp^n\) scheme. It determines the addition chain to be used, and hence the sequence of squares and multiplies which occur:

\[
\begin{align*}
(2,0) & \quad S & & (2,1), (3,0) & \quad SM \\
(3,1), (3,2), (5,0) & \quad SMM & & (5,1), (5,2), (5,3) & \quad SMMM \\
(5,4) & \quad SSMM
\end{align*}
\]

• Base sub-chain boundaries are deduced from occurrences of \(S\) except for ambiguity between \((5,4)\) and \((2,0)(3,x)\) or \((2,0)(5,0)\).
## Running Example

<table>
<thead>
<tr>
<th>$D$</th>
<th>$(m,d)$</th>
<th>$S&amp;M$ subchain</th>
<th>Interpretations</th>
</tr>
</thead>
<tbody>
<tr>
<td>235</td>
<td>(3,1)</td>
<td>$S(M)M$</td>
<td>(3,1), (3,2), (5,0)</td>
</tr>
<tr>
<td>78</td>
<td>(2,0)</td>
<td>$S$</td>
<td>(2,0)</td>
</tr>
<tr>
<td>39</td>
<td>(5,4)</td>
<td>$SSMM$</td>
<td>(5,4), (2,0)(3,1), (2,0)(3,2), (2,0)(5,0)</td>
</tr>
<tr>
<td>7</td>
<td>(2,1)</td>
<td>$SM$</td>
<td>(2,1), (3,0)</td>
</tr>
<tr>
<td>3</td>
<td>(3,0)</td>
<td>$SM$</td>
<td>(2,1), (3,0)</td>
</tr>
<tr>
<td>1</td>
<td>(2,1)</td>
<td>$(S)M$</td>
<td>(2,1)</td>
</tr>
</tbody>
</table>

**Result:** $SM.S(SSMM.SM.SM.M)$ with $112341 = 48$ choices.

(Modifications for end conditions: e.g. the initial $M$ and final $S$ are superfluous.)
Exponent Choices

• There is/are:
  1 way to interpret $S$
  2 ways to interpret $SM$
  3 ways to interpret $SMM$ with preceding $M$
  4 ways to interpret $SMM$ with preceding $S$
  4 ways to interpret $SMMM$

• The probabilities of the sub-chains can be calculated:
  $p_S = \text{prob}(S) = p_{2,0}$; $p_{SM} = p_{2,1} + p_{3,0}$; $p_{SMM} =$ etc.

• So average number of choices to interpret a sub-chain is
  $1p's \ 2p'sm \ 3p'_{msmm} \ 4p'_{ssmm} \ 4p'_{smmm} \approx 1.7079$
  where ' is the modification due to parsing $SSMM$ into $S.SMM$ always.
S&M Theorem

- There are on average $0.766 \log_2 D$ occurrences of $S$ per addition chain, so $1.7079^{0.766 \log_2 D} = D^{0.5916}$ exponents which can generate the same S&M sequence.

- **Theorem:** The search space for exponents with the same S&M sequence as $D$ has size approx $D^{3/5}$.

- For 4-ary $exp^n$, it is much easier to average traces, easier to be certain of the S&M sequence, and the search space is only $D^{7/18}$ – which is smaller.

- Both are computationally infeasible searches.
Operand Re-Use

- From its location, address, power use in mult^n or Hamming weight, it may be possible to identify re-use of operands. Assume we know when operands are equal, but nothing more.
  - since only squares have equal operands, this means the S&M sequence can be recovered.
  - for classical *m-ary & sliding windows* exp^n, there is a fixed pre-computed multiplicand for each exp^t digit value, so the secret exponent can be reconstructed uniquely.

- **MIST** operand sharing leaves ambiguities:
  - (2,1) and (3,0) have the same operand sharing pattern and both are common: \( p_{SM} = 0.458 \).
## Running Example

<table>
<thead>
<tr>
<th>$D$</th>
<th>$(m,d)$</th>
<th>Op Sharing Interpretations</th>
</tr>
</thead>
<tbody>
<tr>
<td>235</td>
<td>(3,1)</td>
<td>(3,1)</td>
</tr>
<tr>
<td>78</td>
<td>(2,0)</td>
<td>(2,0)</td>
</tr>
<tr>
<td>39</td>
<td>(5,4)</td>
<td>(5,4)</td>
</tr>
<tr>
<td>7</td>
<td>(2,1)</td>
<td>(2,1), (3,0)</td>
</tr>
<tr>
<td>3</td>
<td>(3,0)</td>
<td>(2,1), (3,0)</td>
</tr>
<tr>
<td>1</td>
<td>(2,1)</td>
<td>(2,1)</td>
</tr>
</tbody>
</table>

**Result:** $2^2 = 4$ choices.

( Modifications for end conditions: e.g. the most significant digit $d$ is non-zero.)
Operand Re-Use Theorem

- With similar working to the S&M case:

**Theorem**: For MIST, the search space for exponents with the same operand sharing sequence as $D$ has size approx $D^{1/3}$.

- The search space for $m$-ary exp$^n$ has size $D^0$.

- There are several necessary boring technicalities to ensure mathematical rigour – skip sections 4 and 5 in the paper!
Difficulties?

- The above requires correct identification of \( \text{op}^d \) sharing first (operands are never used more than 3 times)
- Mistakes are not self-correcting in an obvious way; only a few errors can vastly increase the search space.
- There is no known way to combine results from other \( \text{exp}^{\text{ns}} \), especially if exponent blinding is applied.
- *Always selecting zero digits vastly decreases the search.*
- Small public exponent, no exponent blinding and known RSA modulus provide half the bits, reducing the search space to \( D^{1/6} \).
Conclusion

- “Random-ary exponentiation” – a novel exp<n> alg<m> suitable for RSA on smartcard (no inverses need to be computed).
- Time & Space are comparable to 4-ary exp<n>.
- Random choices & little operand re-use make the usual averaging for DPA much more restricted.
- MIST is much stronger against power analysis than standard exp<n> algorithms.