Threshold Implementations
with some applications

Begül Bilgin

UNIVERSITEIT TWENTE.
Threshold Implementations

A provably secure *countermeasure* against first order *power analysis* that is based on *multi party computation* and *secret sharing*
Outline

- Power analysis
- Countermeasures
- Threshold Implementations
- Applications
Power Analysis

1998 - Paul Kocher, Joshua Jaffe and Benjamin Jun
Power Analysis
Simple Power Analysis

DES implementation

[courtesy: P. Kocher]
Simple Power Analysis

RSA Exponentiation

Crypto coprocessor optimized for squaring

[courtesy: C.Clavier]
Simple Power Analysis

Crypto coprocessor optimized for squaring

RSA Exponentiation

Key dependent implementations

Crypto coprocessor optimized for squaring

[courtesy: C.Clavier]
Differential Power Analysis

Leakage of \textit{Crypt}(pt, key) is like a fingerprint
Countermeasures

- Hardware countermeasures
  - Balancing power consumption [Tiri et al., CHES’03]
- Masking
  - Intermediate values [Chari et al., CRYPTO’99; Goubin et al., CHES’99]
  - Threshold Implementations [Nikova et al., ICISC’08]
  - Shamir’s Secret Sharing [Goubin et al., Prouff et al., CHES’11]
- Leakage-Resilient Crypto
Threshold Implementations

\[(x, y, z, \ldots) \xrightarrow{S()} (a, b, c, \ldots)\]
Threshold Implementations

Shares

$(x_1, y_1, z_1, \ldots)$

$(x_2, y_2, z_2, \ldots)$

$(x_s, y_s, z_s, \ldots)$

$S_1$

$(a_1, b_1, c_1, \ldots)$

$(a_2, b_2, c_2, \ldots)$

$(a_s, b_s, c_s, \ldots)$

$S_2$

$S_s$
Threshold Implementations

\[ (x_1, y_1, z_1, \ldots) \oplus \]

\[ (x_2, y_2, z_2, \ldots) \oplus \]

\[ \vdots \]

\[ (x_s, y_s, z_s, \ldots) = \]

\[ (x, y, z, \ldots) \]

\[ (a_1, b_1, c_1, \ldots) \oplus \]

\[ (a_2, b_2, c_2, \ldots) \oplus \]

\[ \vdots \]

\[ (a_s, b_s, c_s, \ldots) = \]

\[ (a, b, c, \ldots) \]

Correct
Threshold Implementations

\[(x_1, y_1, z_1, \ldots) \rightarrow (a_1, b_1, c_1, \ldots)\]

\[(x_2, y_2, z_2, \ldots) \rightarrow (a_2, b_2, c_2, \ldots)\]

\[\vdots\]

\[(x_s, y_s, z_s, \ldots) \rightarrow (a_s, b_s, c_s, \ldots)\]

\[=\]

\[(x, y, z, \ldots) \rightarrow (a, b, c, \ldots)\]

Correct, Non-complete
Threshold Implementations

Non-completeness

\[ S(x, y, z) = x + yz \]

\[ S_1 = x_2 + y_2z_2 + y_2z_3 + y_3z_2 \]

\[ S_2 = x_3 + y_3z_3 + y_3z_1 + y_1z_3 \]

\[ S_3 = x_1 + y_1z_1 + y_1z_2 + y_2z_1 \]

To protect a function with degree \( d \), at least \( d+1 \) shares are required
Threshold Implementations

\[(x_1, y_1, z_1, \ldots) \oplus \ldots \oplus (x_s, y_s, z_s, \ldots) = (x, y, z, \ldots)\]

\[(a_1, b_1, c_1, \ldots) \oplus \ldots \oplus (a_s, b_s, c_s, \ldots) = (a, b, c, \ldots)\]

Correct, Non-complete, Uniform
Threshold Implementations

Uniformity

A masking $X$ is **uniform** if and only if there exists a constant $p$ such that for all $x$ we have:

- if $X \in S_h(x)$ then $\Pr(X|x) = p$, else $\Pr(X|x) = 0$.

\[
\begin{array}{c|c|c}
 a & b & f = a \text{ AND } b \\
 0 & 0 & 0 \\
 0 & 1 & 0 \\
 1 & 0 & 0 \\
 1 & 1 & 1 \\
\end{array}
\]

\[
\begin{array}{c|cccccccc}
 f & (0,0,0) & (0,0,1) & (0,1,0) & (0,1,1) & (1,0,0) & (1,0,1) & (1,1,0) & (1,1,1) \\
 0 & 4 & 0 & 0 & 4 & 0 & 4 & 4 & 0 \\
 0 & 4 & 0 & 0 & 4 & 0 & 4 & 4 & 0 \\
 0 & 4 & 0 & 0 & 4 & 0 & 4 & 4 & 0 \\
 1 & 0 & 4 & 4 & 0 & 4 & 0 & 0 & 4 \\
 0 & 12 & 0 & 0 & 12 & 0 & 12 & 12 & 0 \\
 1 & 0 & 4 & 4 & 0 & 4 & 0 & 0 & 4 \\
\end{array}
\]
Threshold Implementations

Uniformity

A masking $X$ is *uniform* if and only if there exists a constant $p$ such that for all $x$ we have:

if $X \in \mathcal{S}_h(x)$ then $\Pr(X|x) = p$, else $\Pr(X|x) = 0$.

If unshared function is a permutation, the shared function should also be a permutation.
Threshold Implementations

If the masking of $x$ is uniform, then the stochastic functions $S_i$ and $x$ are independent (for any choice of $i$).

If the masking of $x$ is uniform and the circuit $S$ is non-complete, then any single component function of $S$ does not leak information on $x$. 
Threshold Implementations

Observations

✓ Linear functions are easy to protect

● As the nonlinearity increases
  ✗ DPA becomes easier
  ✗ Sharing becomes costly
  ✓ S-boxes become mathematically stronger

Decomposing nonlinear functions
Threshold Implementations

Decomposing nonlinear functions

Most of the block ciphers use 4x4 permutations

4x4 permutations have at most degree 3
Threshold Implementations

Decomposing nonlinear functions

\[ S = G \circ F \]

All \( n \times n \) affine bijections are in alternating group \( A_{2^n} \)

All 4x4 quadratic S-boxes belong to \( A_{16} \)

A 4x4 bijection can be decomposed using quadratic bijections IFF it belongs to \( A_{16} \)
Threshold Implementations

Decomposing nonlinear functions

$$S' = AoSoB$$

302 affine equivalent classes of 4x4 S-boxes

$$S = G \circ F$$

S-boxes in half of them belong to $A_{16}$ 3 shares
## Threshold Implementations

### Decomposing nonlinear functions

<table>
<thead>
<tr>
<th>remark</th>
<th>unshared</th>
<th>3 shares</th>
<th>4 shares</th>
<th>5 shares</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1 2 3 4</td>
<td>1 2 3</td>
<td>1</td>
</tr>
<tr>
<td>affine</td>
<td>1</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>quadratic</td>
<td>6 5 1</td>
<td>6</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>cubic in $A_{16}$</td>
<td>30 28 2</td>
<td>30</td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>cubic in $A_{16}$</td>
<td>114 113 1</td>
<td>114</td>
<td></td>
<td>114</td>
</tr>
<tr>
<td>cubic in $S_{16} \setminus A_{16}$</td>
<td>151</td>
<td></td>
<td>4 22 125</td>
<td>151</td>
</tr>
</tbody>
</table>
## Threshold Implementations

### Decomposing nonlinear functions

<table>
<thead>
<tr>
<th>remark</th>
<th>unshared</th>
<th>3 shares</th>
<th>4 shares</th>
<th>5 shares</th>
</tr>
</thead>
<tbody>
<tr>
<td>affine</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>quadratic</td>
<td>6</td>
<td>5</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>cubic in $A_{16}$</td>
<td>30</td>
<td>28</td>
<td>2</td>
<td>30</td>
</tr>
<tr>
<td>cubic in $A_{16}$</td>
<td>114</td>
<td>113</td>
<td>1</td>
<td>114</td>
</tr>
<tr>
<td>cubic in $S_{16} \setminus A_{16}$</td>
<td>151</td>
<td>113</td>
<td>1</td>
<td>114</td>
</tr>
</tbody>
</table>

**Uniformity problem**

25
Threshold Implementations

Decomposing nonlinear functions

<table>
<thead>
<tr>
<th>remark</th>
<th>unshared</th>
<th>3 shares</th>
<th>4 shares</th>
<th>5 shares</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>affine</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>quadratic</td>
<td>6</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>cubic in $A_{16}$</td>
<td>30</td>
<td></td>
<td>28</td>
<td>2</td>
</tr>
<tr>
<td>cubic in $A_{16}$</td>
<td>114</td>
<td></td>
<td>113</td>
<td>1</td>
</tr>
<tr>
<td>cubic in $S_{16} \setminus A_{16}$</td>
<td>151</td>
<td></td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

Many S-boxes with good cryptographic properties
Threshold Implementations
Decomposing nonlinear functions

<table>
<thead>
<tr>
<th>remark</th>
<th>unshared</th>
<th>3 shares</th>
<th>4 shares</th>
<th>5 shares</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>affine</td>
<td>1</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>quadratic</td>
<td>6</td>
<td>5</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>cubic in $A_{16}$</td>
<td>30</td>
<td>28</td>
<td>2</td>
<td>30</td>
</tr>
<tr>
<td>cubic in $A_{16}$</td>
<td>114</td>
<td>113</td>
<td>1</td>
<td>114</td>
</tr>
<tr>
<td>cubic in $S_{16} \setminus A_{16}$</td>
<td>151</td>
<td></td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

Applications

• PRESENT: uses 4x4 S-box with degree 3
  • Implemented with 3 shares
  • 3,3 kGE (1,1 kGE unprotected)

• KECCAK: uses 5x5 S-box with degree 2
  • Implemented with 3 shares
  • 32,6 kGE (10,6 kGE unprotected)

• AES: uses 8x8 S-box with degree 7
  • Implemented with $n$ shares
  • 8,6 kGE (2,4 kGE unprotected)
Applications

AES S-box

Based on multiplicative inverse in GF(256) and an affine transformation

Possible to implement using tower field approach
Applications

AES S-box

2 shares
Applications

AES S-box

4 input 3 output shares
Applications

AES S-box

lin. map

GF(2⁴) square scaler

GF(2⁴) multiplier

GF(2⁴) inverter

GF(2⁴) multiplier

GF(2⁴) multiplier

inv. lin. map

5 shares
Applications

AES S-box

registers after every nonlinear function
Applications

AES S-box

TI of S-box is 3 kGE
(233 GE unprotected S-box
2,4 kGE unprotected AES)
Applications

Design of the crypto algorithm

Secure implementation crypto algorithm

Small on HW

“As light as a feather, and as hard as dragon-scales”

Bilbo Baggins in “The Lord of the Rings: The Fellowship of the Ring”
Applications

FIDES-80

- Authentication Encryption
- Monkey duplex sponge construction
- Key, nonce and tag size is 80 bits
- 10 rounds
Applications

FIDES-80
Applications

FIDES-80
Applications

FIDES-80

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_{i,0}$</td>
<td>$a_{0,0}$</td>
<td>$a_{0,1}$</td>
<td>$a_{0,2}$</td>
<td>$a_{0,3}$</td>
</tr>
<tr>
<td>$a_{i,1}$</td>
<td>$a_{1,0}$</td>
<td>$a_{1,1}$</td>
<td>$a_{1,2}$</td>
<td>$a_{1,3}$</td>
</tr>
<tr>
<td>$a_{i,2}$</td>
<td>$a_{2,0}$</td>
<td>$a_{2,1}$</td>
<td>$a_{2,2}$</td>
<td>$a_{2,3}$</td>
</tr>
<tr>
<td>$a_{i,3}$</td>
<td>$a_{3,0}$</td>
<td>$a_{3,1}$</td>
<td>$a_{3,2}$</td>
<td>$a_{3,3}$</td>
</tr>
</tbody>
</table>

- State
- SubBytes
- ShiftRows
Applications

FIDES-80

Almost MDS
branch number is 4
Applications

FIDES-80

1R

State

→

SubBytes

→

ShiftRows

→

MixColumns

→

ConstantAddition
Applications

FIDES-80

• S-box: 5-bit Almost Bent (AB)
  - optimal resistance against differential & linear cryptanalysis
  - degree 2 (two), degree 3 (one), degree 4 (one)
Applications

FIDES-80

- S-box: 5-bit Almost Bent (AB)
  - optimal resistance against differential & linear cryptanalysis
  - degree 2 (two), degree 3 (one), degree 4 (one)

TI with 4 shares
Applications

FIDES-80

Affine Equivalent to AB permutation

Unshared S-box

Shared S-box

# of S-boxes

# of GE (UMC 180nm)
Applications

FIDES-80

Affine Equivalent to AB permutation

Unshared S-box

Shared S-box

4,2 kGE (1,1kGE unprotected)
Conclusion

- TI is provably secure against first order DPA
- TI can be efficient
- Room for improvement:
  - Solutions to uniformity problems
  - Security against higher order DPA
- Consider countermeasures during design process
Thank You!