Fast Substitution-Box Evaluation Algorithm and Its Efficient Higher-Order Masking Scheme for Block Ciphers

Hai Huang\textsuperscript{1,2}, Leibo Liu\textsuperscript{1}\textsuperscript{*}, Min Zhu\textsuperscript{1}, Shouyi Yin\textsuperscript{1} & Shaojun Wei\textsuperscript{1}

\textsuperscript{1}Institute of Microelectronics, Tsinghua University, Beijing 100084, China; \textsuperscript{2}Harbin University of Science and Technology, Harbin 150080, China

Appendix A  Proof of Theorem 1

If \( a \) is the primitive root, then all elements over \( GF(2^n) \) can be generated from (A1).

\[ x_i = a^i (\mod p), \ i \in [0, 2^n - 1] \tag{A1} \]

Since,

\[ x_i = a^i (\mod p) \]

Here, \( a^k \) is one solution of \( x^m \equiv 1 (\mod p) \), hence all solutions satisfy,

\[ x_j = a^{j} (\mod p), \ j \in \mathbb{N} \tag{A3} \]

Since the equation only has \( m \) solutions, hence this theorem is proven.

Appendix B  Proof of Theorem 2

The \( k \) of \( m \)-degree congruence equations can be constructed as:

\[
\begin{cases}
    x^m \equiv b_1 \equiv 1 (\mod p) \\
    x^m \equiv b_2 (\mod p) \\
    \cdots \\
    x^m \equiv b_{k-1} (\mod p) \\
    x^m \equiv b_k (\mod p)
\end{cases} \tag{B1}
\]

Since \( b_j, j \in [2, k] \) are solutions of \( x^k \equiv 1 (\mod p) \), then

\[ b_j = a^{(j-1)m} (\mod p), \ j \in [1, m] \tag{B2} \]

Hence, multiplying both sides of (B1) by \( b_j \) from (B2) proves this theorem.

* Corresponding author (email: liulb@tsinghua.edu.cn)
## Appendix C  Table C1

<table>
<thead>
<tr>
<th>n</th>
<th>Factors</th>
<th>$p$</th>
<th>m</th>
<th>$x$ satisfies $x^m \equiv 1 \pmod{p}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>(3, 5)</td>
<td>$x^4 + x + 1$</td>
<td>3</td>
<td>1, 6, 7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>1, 8, 10, 12, 15</td>
</tr>
<tr>
<td>6</td>
<td>(3, 21)</td>
<td>$x^6 + x + 1$</td>
<td>21</td>
<td>1, 3, 5, 8, 13, 14, 15, 17, 18, 22, 23, 24, 25, 40, 43, 51, 54, 57, 58, 59, 62</td>
</tr>
<tr>
<td></td>
<td>(7, 9)</td>
<td></td>
<td>7</td>
<td>1, 14, 15, 22, 23, 24, 25</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td>1, 6, 11, 20, 26, 28, 31, 58, 59</td>
</tr>
<tr>
<td>8</td>
<td>(3, 85)</td>
<td>$x^8 + x^4 + x^3 + x^2 + 1$</td>
<td>17</td>
<td>1, 12, 13, 80, 176, 237</td>
</tr>
<tr>
<td></td>
<td>(5, 51)</td>
<td>$x^8 + x^4 + x^3 + x^2 + 1$</td>
<td>15</td>
<td>1, 12, 13, 80, 176, 237</td>
</tr>
<tr>
<td></td>
<td>(15, 17)</td>
<td></td>
<td>17</td>
<td>1, 8, 29, 47, 53, 54, 57, 64, 74, 99, 102, 171, 179, 194, 211, 232, 239</td>
</tr>
<tr>
<td>9</td>
<td>(7, 73)</td>
<td>$x^9 + x^4 + 1$</td>
<td>7</td>
<td>1, 28, 29, 332, 333, 336, 337</td>
</tr>
</tbody>
</table>

## Algorithm D1

**Algorithm D1** Higher-order masking scheme for GLUT

**Input:** Boolean shares $x_{a_0}, x_{a_1}, x_{a_2}, \ldots, x_{a_d}$ of $x$

**Output:** Boolean shares $y = GLUT(x) \oplus x_{b_1} \oplus x_{b_2} \cdots \oplus x_{b_d}, x_{b_1}, x_{b_2}, \ldots, x_{b_d}$

**BM to MM operation**

1: $(x_{m_0}, x_{m_1}, \cdots, x_{m_d}) \leftarrow BMtoMM(x_{a_0}, x_{a_1}, \cdots, x_{a_d}); \ \\text{\textbackslash Alg. D2}$

**Table re-computation**

2: for $n = 1$ to $d$

3: Random generates $x_{b_n} \in GF(2^n)$; \ \Masks refreshing

4: $GLUT'(u) \leftarrow GLUT(u \otimes x_{m_n}) \oplus x_{b_n};$ \ \Re-computation

5: $y \leftarrow GLUT'(x_{m_0});$

6: end

7: return $y, x_{b_1}, x_{b_2}, \ldots, x_{b_d}.$
Algorithm D2

**Algorithm D2** BM to MM

**Input:** Boolean shares $xa_0, xa_1, xa_2, \ldots, xa_d$

**Output:** Multiplicative shares $xm_0, xm_1, xm_2, \ldots, xm_d$

1. $xm_0 = xa_0$
2. for $n = 1$ to $d$
3. Random generates $i \in [1, k], j \in [1, m]$;
4. $xm_n = M_{ij}$;
5. $xm_0 = xm_0 \otimes xm_n$;
6. for $k = 1$ to $d - n$
7. Random generates $r \in [0, 255]$; // Masks refreshing
8. $xa_k = xm_n \otimes xa_k$;
9. $xa_k = r \otimes xa_k$;
10. $xm_0 = r \oplus xa_k$;
11. $xa_k = r$;
12. end
13. $xa_{d-n+1} = xm_n \otimes xa_{d-n+1}$;
14. $xm_0 = xm_0 \oplus xa_{d-n+1}$;
15. end
16. return $xm_0, xm_1, xm_2, \ldots, xm_d$.

---

**Appendix E** Table E1

<table>
<thead>
<tr>
<th></th>
<th>XOR</th>
<th>Multiplication</th>
<th>LUT access</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_2$-linear</td>
<td>0</td>
<td>0</td>
<td>$d + 1$</td>
</tr>
<tr>
<td>Multiplication over $GF(2^n)$</td>
<td>$2(d^2 + d)$</td>
<td>$(d + 1)^2$</td>
<td>0</td>
</tr>
<tr>
<td>$x \times g(x)$</td>
<td>$5(d^2 + d)$</td>
<td>0</td>
<td>$2d^2 + 3d + 1$</td>
</tr>
<tr>
<td>GLUT</td>
<td>$d^2 + d$</td>
<td>$(d(d + 3))/2$</td>
<td>$d$</td>
</tr>
</tbody>
</table>

**Table E2** Computing-complexity of different AEs S-boxes.

<table>
<thead>
<tr>
<th></th>
<th>XOR</th>
<th>Multiplication</th>
<th>LUT access</th>
</tr>
</thead>
<tbody>
<tr>
<td>[6]</td>
<td>$8(d^2 + d)$</td>
<td>$4(d + 1)^2$</td>
<td>$7(d + 1)$</td>
</tr>
<tr>
<td>[7]</td>
<td>$8(d^2 + d)$</td>
<td>$4(d + 1)^2$</td>
<td>$11(d + 1)$</td>
</tr>
<tr>
<td>[8]</td>
<td>$8(d^2 + d)$</td>
<td>$4(d + 1)^2$</td>
<td>$6(d + 1)$</td>
</tr>
<tr>
<td>[9]</td>
<td>$17(d^2 + d)$</td>
<td>$(d + 1)^2$</td>
<td>$6d^2 + 15d + 10$</td>
</tr>
<tr>
<td>This paper</td>
<td>$3(d^2 + d)$</td>
<td>$(3d^2 + 7d + 2)/2$</td>
<td>$5d + 4$</td>
</tr>
</tbody>
</table>