Generalized Polynomial Decomposition for S-boxes with Application to Side-Channel Countermeasures

Dahmnum Goudarzi, Matthieu Rivain, Srinivas Vivek, and Damien Vergnaud
Background
Secure Software S-box Implementations

- Higher-Order Masking

\[ x = x_1 + x_2 + \cdots + x_d \]

- Main Challenge: S-box evaluations
  - Linear operations: \( O(d) \)
  - Non-linear operations: \( O(d^2) \)

- Goal: Find S-box representation with less non-linear operations
Polynomial Methods

- S-box seen as a polynomial over $\mathbb{F}_{2^n}$

$$S(x) = \sum_{i=0}^{n} a_i x^i$$

---

Generic Methods

$$S(x) = \sum_{i} (p_i \star q_i)(x)$$

- CRV decomposition, $\star = \times$
- Algebraic decomposition, $\star = \circ$

Specific Methods: example on AES

$$S_{AES}(x) = \text{Aff}(x^{254})$$

- RP 4-mult chain on $\mathbb{F}_{2^8}$
- KHL 5-mult chain on $\mathbb{F}_{2^4}$
Bitslice Methods

- S-box seen as a Boolean circuit

\[ S: x \rightarrow (f_1(x), f_2(x), \ldots, f_m(x)) \]

<table>
<thead>
<tr>
<th>Generic Methods</th>
<th>Specific Methods: example on AES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Based on Boolean functions</td>
<td>Based on a Boolean circuit (BMP13)</td>
</tr>
</tbody>
</table>
Polynomial vs Bitslice

- Generic 8-bit S-box evaluation

![Graph showing comparison between Polynomial and Bitslice for Generic 8-bit S-box evaluation](image.png)
Full Field
CRV decomposition

Boolean Field
Boolean decomposition

Intermediate Field?
This work

\[ S(x) \rightarrow (S_1(y, z), S_2(y, z)) \]

1 8-bit function
4-bit functions
Motivation

- Working on smaller fields
  - Degree of parallelisation increased (32-bit architecture)

<table>
<thead>
<tr>
<th></th>
<th>Boolean Case</th>
<th>4-bit field</th>
<th>8-bit field</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mult. in //</td>
<td>32</td>
<td>8</td>
<td>4</td>
</tr>
</tbody>
</table>

- Example: 16 AES S-box with polynomial method
Our results

- Generalized decomposition method for any S-boxes w.r.t 3 parameters
  - $n$: number of inputs
  - $m$: number of output elements
  - $\lambda$: bit-size of the elements

- Study of the median case: example on 8-bit S-boxes:

  $$S(x, y) = (f_1(x, y), f_2(x, y)) \text{ with } x, y \in \mathbb{F}_{2^4}$$

- Implementation in ARM assembly to compare with state of the art
Generalized Decomposition Method
S-box Characterization

- S-box seen as a $n\lambda$-bit to $m\lambda$-bit polynomial over $\mathbb{F}_{2^\lambda}$:

$$S(x) = (f_1(x), f_2(x), \ldots, f_m(x))$$

where $f_1, f_2, \ldots, f_m \in \mathcal{F}_{n,\lambda}$ (set of functions from $\mathbb{F}_{2^\lambda}^n$ to $\mathbb{F}_{2^\lambda}$)
Coordinate Function Decomposition

\[ f(x) = \sum_{i=0}^{t} g_i(x) \cdot h_i(x) \]

- \( g_i \): random linear combinations from a basis \( \langle \tilde{B} \rangle \) with

\[ \langle \tilde{B}_i \rangle = \left\{ g, g = \sum_{i=0}^{\lambda-1} \sum_{\phi \in B} c_{\phi,i} \times \phi^{2^i} \right\} \]

- find \( c_{i,j} \) s.t \( h_i = \sum_j c_{i,j} \phi_j \) by solving:

\[ f(x) = \sum_i \left( \sum_j a_{i,j} \phi_j(x) \right) \left( \sum_j c_{i,j} \phi_j(x) \right), \ \forall x \]
Solving a Linear System

\[ f(x) = \sum_i \left( \sum_j a_{i,j} \phi_j(x) \right) \left( \sum_j c_{i,j} \phi_j(x) \right), \forall x \]

- \( \{e_i\}_{i=1}^{2n^λ} = \mathbb{F}_{2^λ}^n \)

\[ A_1 c_1 + A_2 c_2 + \cdots + A_t c_t = (f(e_1), f(e_2), \ldots, f(e_{2^n})) \]

\[
A_i = \begin{pmatrix}
\phi_1(e_1) \cdot g_i(e_1) & \phi_2(e_1) \cdot g_i(e_1) & \cdots & \phi_{|B|}(e_1) \cdot g_i(e_1) \\
\phi_1(e_2) \cdot g_i(e_2) & \phi_2(e_2) \cdot g_i(e_2) & \cdots & \phi_{|B|}(e_2) \cdot g_i(e_2) \\
\vdots & \vdots & \ddots & \vdots \\
\phi_1(e_{2^n}) \cdot g_i(e_{2^n}) & \phi_2(e_{2^n}) \cdot g_i(e_{2^n}) & \cdots & \phi_{|B|}(e_{2^n}) \cdot g_i(e_{2^n})
\end{pmatrix}
\]
Conditions

- \((t + 1)|\mathcal{B}|\) unknowns, \(2^{n\lambda}\) equations:

\[
(t + 1)|\mathcal{B}| \geq 2^{n\lambda}
\]

- Condition on the sum: \(t \geq \left\lceil \frac{2^{n\lambda}}{|\mathcal{B}|} \right\rceil - 1\)

- Condition on the basis: \(\langle \mathcal{B} \times \mathcal{B} \rangle\) has to span the entire space \(\mathcal{F}_{n,\lambda}\)
Spanning Property

\[ \langle \bar{B} \times \bar{B} \rangle = \mathcal{F}_{n,\lambda} \iff \text{rank}(\text{Mat}(\bar{B} \times \bar{B})) = 2^{n\lambda} \]

with

\[
\text{Mat}(S) = \begin{pmatrix}
\varphi_1(e_1) & \varphi_2(e_1) & \ldots & \varphi_{|S|}(e_1) \\
\varphi_1(e_2) & \varphi_2(e_2) & \ldots & \varphi_{|S|}(e_2) \\
\vdots & \vdots & \ddots & \vdots \\
\varphi_1(e_{2^{n\lambda}}) & \varphi_2(e_{2^{n\lambda}}) & \ldots & \varphi_{|S|}(e_{2^{n\lambda}})
\end{pmatrix}
\]

where \( \{\varphi_1, \varphi_2, \ldots, \varphi_{|S|}\} = S \) and \( S = \langle \bar{B} \times \bar{B} \rangle \)
Basis Construction

- Start with $\mathcal{B} = \{1, x_1, x_2 \ldots, x_n\}$

- Pick $\phi, \psi$ in $\langle \mathcal{B} \rangle$ at random, where

$$\langle \mathcal{B} \rangle = \left\{ g : g = \sum_{i=0}^{\lambda-1} \sum_{\phi \in \mathcal{B}} c_{\phi,i} \times \phi^{2^i} \right\}$$

- Compute $\text{rank}(\text{Mat}(S \times S))$ with $S = \mathcal{B} \cup \phi \cdot \psi$

- Redo $N$ times and choose $(\phi, \psi)$ that increase the rank most

- Repeat until rank is at least $2^{n\lambda}$
### Random Basis

<table>
<thead>
<tr>
<th>$(\lambda, n)$</th>
<th>4-bit s-boxes</th>
<th>8-bit s-boxes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$(1,4)$</td>
<td>$(1,8)$</td>
</tr>
<tr>
<td></td>
<td>$(2,2)$</td>
<td>$(2,4)$</td>
</tr>
<tr>
<td></td>
<td>$(4,1)$</td>
<td>$(4,2)$</td>
</tr>
<tr>
<td></td>
<td>$(1,8)$</td>
<td>$(8,1)$</td>
</tr>
<tr>
<td>$</td>
<td>B_1</td>
<td>$</td>
</tr>
<tr>
<td>$r$</td>
<td>2</td>
<td>17</td>
</tr>
</tbody>
</table>

- Improvements w.r.t previous methods:
  - Boolean case : initial basis from 25 to 17
Decomposition of the S-box

- **Sbox**: \( S : x \rightarrow (f_1(x), f_2(x), \ldots, f_m(x)) \)

- Apply \( m \) coordinate decompositions on the \( f_i \)'s

- **Basis update:**
  - Start with a basis \( B_i \)
  - At each step: \( B_{i+1} \leftarrow B_i \cup \{g_i \cdot h_i\}_{i=0}^{t_i} \)
**Decomposition example**

\[ S : \mathbb{F}_{2^8} \rightarrow \mathbb{F}_{2^8} \]

\[ n = 2, m = 2, \lambda = 4 \rightarrow S(x, y) = (f_1(x, y), f_2(x, y)) \]

\[ B_1 \quad | \quad B_1 \times B_1 \text{ spans } \mathcal{F}_{n, \lambda} \]

\[ f_1(x) = \sum_{i=0}^{t_1} g_{1,i}(x) \cdot h_{1,i}(x) \]

\[ B_2 = B_1 \cup \{ g_{1,i} \cdot h_{1,i} \}_{i=0}^{t_1} \]

\[ f_2(x) = \sum_{i=0}^{t_2} g_{2,i}(x) \cdot h_{2,i}(x) \]
Experimental Results and Implementations
Optimal Parameters

- Cost of the decomposition:

\[ r + \sum_{i=1}^{m} t_i \]

with \( t_i \geq \left\lfloor \frac{2n^\lambda - 1}{\lambda |B_i| + 1} \right\rfloor - 1 \)

- Optimal parameters:

\[ t_i = \left\lfloor \frac{2n^\lambda - 1}{\lambda |B_i| + 1} \right\rfloor - 1 \]
Achievable Results for Median Cases

| Optimal/Achievable | $(\lambda, n)$ | $|B_1|$ | $r$ | $t_1, t_2, \ldots, t_n$ | $C^*$ |
|--------------------|----------------|--------|-----|---------------------|-------|
| **4-bit s-boxes**  |                |        |     |                     |       |
| Optimal            | (2,2)          | 5      | 2   | 1,1                 | 4     |
| Achievable         | (2,2)          | 5      | 2   | 1,1                 | 4     |
| **8-bit s-boxes**  |                |        |     |                     |       |
| Optimal            | (2,4)          | 16     | 11  | 8,5,4,3             | 31    |
| Achievable         | (2,4)          | 16     | 11  | 9,6,5,3             | 34    |
| Optimal            | (4,2)          | 10     | 7   | 6,4                 | 17    |
| Achievable         | (4,2)          | 10     | 7   | 7,4                 | 18    |
Implementation Results

<table>
<thead>
<tr>
<th></th>
<th>Code Size</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRV</td>
<td>27.5 KB</td>
<td>80d B</td>
</tr>
<tr>
<td>Boolean Dec</td>
<td>4.6 KB</td>
<td>644d B</td>
</tr>
<tr>
<td>Our impl.</td>
<td>8.7 KB</td>
<td>92d B</td>
</tr>
</tbody>
</table>

![Graph showing clock cycles vs. d for different implementations]
Conclusion

- Generalized decomposition method well suited for any s-boxes or target architectures against side-channel attacks

![Diagram]

Number of coordinate functions

Decomposition field size

[GR16]

This work

Extension based on [PV16]

[CRV14]
Conclusion

- Case study on 32-bit ARM
  - Median case 8-bit S-box => 2 4-bit functions
  - Implementation comparison with state of the art
  - Memory trade-off

- Can suit low end device with smaller architecture
  - Parallelisation level decreased => poor bitslice performances
  - Few memory requirements