Linear cryptanalysis

Simone Ragusa
Last updated Sunday, April 07, 2024

S-box relation

Algorithm to compute the relations and their biases

Python + SageMath

```python
sbox = {
    0: 0,  # 0 * 2 = 0
    1: 2,  # 1 * 2 = 2
    2: 4,  # 2 * 2 = 4
    3: 8,  # non-linear
    4: 6,  # non-linear
    5: 10, # 5 * 2 = 10
    6: 1,  # 6 * 2 = 12 = 1 mod 11
    7: 3,  # 7 * 2 = 14 = 3 mod 11
    8: 5,  # 8 * 2 = 16 = 5 mod 11
    9: 7,  # 9 * 2 = 18 = 7 mod 11
    10: 9  # 10 * 2 = 20 = 9 mod 11
}

p = 11
field = GF(p)

for i in range(p):
    for j in range(p):
        count = 0
        for v in sbox:
            s = field(i * v) + field(j * sbox[v])
            if s == 0:
                count += 1
        b = count/p - 1/p
        if b > 0.5:
            print(f'{i} * v + {j} * S-box(v) = 0 ; bias = {b}')
```

High-bias relations

The S-boxes relations with the highest bias, i.e., $\varepsilon = 8/11$, are the following.

$$v_r^i + 5 \cdot \text{S-box}(v_r^i) = 0$$

$$2 \cdot v_r^i + 10 \cdot \text{S-box}(v_r^i) = 0$$

$$3 \cdot v_r^i + 4 \cdot \text{S-box}(v_r^i) = 0$$

$$4 \cdot v_r^i + 9 \cdot \text{S-box}(v_r^i) = 0$$

$$5 \cdot v_r^i + 3 \cdot \text{S-box}(v_r^i) = 0$$

$$6 \cdot v_r^i + 8 \cdot \text{S-box}(v_r^i) = 0$$

$$7 \cdot v_r^i + 2 \cdot \text{S-box}(v_r^i) = 0$$

$$8 \cdot v_r^i + 7 \cdot \text{S-box}(v_r^i) = 0$$

$$9 \cdot v_r^i + 1 \cdot \text{S-box}(v_r^i) = 0$$

$$10 \cdot v_r^i + 6 \cdot \text{S-box}(v_r^i) = 0$$

There is also the "trivial" one, with bias $\varepsilon = 10/11$, which is the following.

$$0 \cdot v_r^i + 0 \cdot \text{S-box}(v_r^i) = 0$$

Active S-boxes and round relations

Round 1

Active S-boxes: S_1^1, S_1^4, S_1^5, S_1^8.

$$\begin{aligned} T_1^1 &= 5v_1^1 + 3y_1^1 = 5(u^1 + k_1^1) + 3y_1^1 \\ T_1^4 &= 5v_1^4 + 3y_1^4 = 5(u^4 + k_1^4) + 3y_1^4 \\ T_1^5 &= 7v_1^5 + 2y_1^5 = 7(u^5 + k_1^5) + 2y_1^5 \\ T_1^8 &= 7v_1^8 + 2y_1^8 = 7(u^8 + k_1^8) + 2y_1^8 \end{aligned}$$

Active round outputs: w_1^1, w_1^4, w_1^5, w_1^8.

$$\begin{aligned} w_1^1 &= 2y_1^1 + 5y_1^8 \\ w_1^4 &= 2y_1^4 + 5y_1^5 \\ w_1^5 &= y_1^1 + 7y_1^8 \\ w_1^8 &= y_1^4 + 7y_1^5 \end{aligned}$$

Round 2

Active S-boxes: S_2^1, S_2^4, S_2^5, S_2^8.

$$\begin{aligned} T_2^1 &= 4v_2^1 + 5y_2^1 = 4(w_1^1 + k_2^1) + 9y_2^1 \\ T_2^4 &= 4v_2^4 + 5y_2^4 = 4(w_1^4 + k_2^4) + 9y_2^4 \\ T_2^5 &= 0v_2^5 + 0y_2^5 = 0(w_1^5 + k_2^5) + 0y_2^5 = 0 \\ T_2^8 &= 0v_2^8 + 0y_2^8 = 0(w_1^8 + k_2^8) + 0y_2^8 = 0 \end{aligned}$$

Active round outputs: w_2^1, w_2^4, w_2^5, w_2^8.

$$\begin{aligned} w_2^1 &= 2y_2^1 + 5y_2^8 \\ w_2^4 &= 2y_2^4 + 5y_2^5 \\ w_2^5 &= y_2^1 + 7y_2^8 \\ w_2^8 &= y_2^4 + 7y_2^5 \end{aligned}$$

Round 3

Active S-boxes: S_3^1, S_3^4, S_3^5, S_3^8.

$$\begin{aligned} T_3^1 &= 6v_3^1 + 8y_3^1 = 6(w_2^1 + k_3^1) + 8y_3^1 \\ T_3^4 &= 6v_3^4 + 8y_3^4 = 6(w_2^4 + k_3^4) + 8y_3^4 \\ T_3^5 &= 2v_3^5 + 10y_3^5 = 2(w_2^5 + k_3^5) + 10y_3^5 \\ T_3^8 &= 2v_3^8 + 10y_3^8 = 2(w_2^8 + k_3^8) + 10y_3^8 \end{aligned}$$

Active round outputs: w_3^1, w_3^4, w_3^5, w_3^8.

$$w_3^1 = 2y_3^1 + 5y_3^8$$
$$w_3^4 = 2y_3^4 + 5y_3^5$$
$$w_3^5 = y_3^1 + 7y_3^8$$
$$w_3^8 = y_3^4 + 7y_3^5$$

Round 4

Active S-boxes: S_4^1, S_4^4, S_4^5, S_4^8.

$$T_4^1 = v_4^1 + 5y_4^1 = w_3^1 + k_4^1 + 5y_4^1$$
$$T_4^4 = v_4^4 + 5y_4^4 = w_3^4 + k_4^4 + 5y_4^4$$
$$T_4^5 = v_4^5 + 5y_4^5 = w_3^5 + k_4^5 + 5y_4^5$$
$$T_4^8 = v_4^8 + 5y_4^8 = w_3^8 + k_4^8 + 5y_4^8$$

Active round outputs: w_4^1, w_4^4, w_4^5, w_4^8.

$$w_4^1 = 2y_4^1 + 5y_4^8 = v_5^1 - k_5^1$$
$$w_4^4 = 2y_4^4 + 5y_4^5 = v_5^4 - k_5^4$$
$$w_4^5 = y_4^1 + 7y_4^8 = v_5^5 - k_5^5$$
$$w_4^8 = y_4^4 + 7y_4^5 = v_5^8 - k_5^8$$

Round 4 S-boxes output y_4^i can be then rewritten as follows.

$$y_4^1 = 2(v_5^1 - k_5^1) - 3(v_5^5 - k_5^5)$$
$$y_4^4 = 2(v_5^4 - k_5^4) - 3(v_5^8 - k_5^8)$$
$$y_4^5 = 10(v_5^8 - k_5^8) - 5(v_5^4 - k_5^4)$$
$$y_4^8 = 10(v_5^5 - k_5^5) - 5(v_5^1 - k_5^1)$$

Up-to-end-of-round-4 full relation

Expanding T_r^i for all r, i.

$$\begin{aligned}
&T_1^1 + T_1^4 + T_1^5 + T_1^8 + T_2^1 + T_2^4 + T_2^5 + T_2^8 + T_3^1 + T_3^4 + T_3^5 + T_3^8 + T_4^1 + T_4^4 + T_4^5 + T_4^8 \\
&= 5(u^1 + k_1^1) + 3y_1^1 + 5(u^4 + k_1^4) + 3y_1^4 + 7(u^5 + k_1^5) + 2y_1^5 + 7(u^8 + k_1^8) + 2y_1^8 \\
&\quad +4(w_1^1 + k_2^1) + 9y_2^1 + 4(w_1^4 + k_2^4) + 9y_2^4 \\
&\quad +6(w_2^1 + k_3^1) + 8y_3^1 + 6(w_2^4 + k_3^4) + 8y_3^4 + 2(w_2^5 + k_3^5) + 10y_3^5 + 2(w_2^8 + k_3^8) + 10y_3^8 \\
&\quad +w_3^1 + k_4^1 + 5y_4^1 + w_3^4 + k_4^4 + 5y_4^4 + w_3^5 + k_4^5 + 5y_4^5 + w_3^8 + k_4^8 + 5y_4^8
\end{aligned}$$

Substituting w_r^i for all r, i.

$$\begin{aligned}
&5(u^1 + k_1^1) + 3y_1^1 + 5(u^4 + k_1^4) + 3y_1^4 + 7(u^5 + k_1^5) + 2y_1^5 + 7(u^8 + k_1^8) + 2y_1^8 \\
&\quad +4(2y_1^1 + 5y_1^8 + k_2^1) + 9y_2^1 + 4(2y_1^4 + 5y_1^5 + k_2^4) + 9y_2^4 \\
&\quad +6(2y_2^1 + 5y_2^8 + k_3^1) + 8y_3^1 + 6(2y_2^4 + 5y_2^5 + k_3^4) + 8y_3^4 + 2(y_2^1 + 7y_2^8 + k_3^5) + 10y_3^5 + 2(y_2^4 + 7y_2^5 + k_3^8) + 10y_3^8 \\
&\quad +2y_3^1 + 5y_3^8 + k_4^1 + 5y_4^1 + 2y_3^4 + 5y_3^5 + k_4^4 + 5y_4^4 + y_3^1 + 7y_3^8 + k_4^5 + 5y_4^5 + y_3^4 + 7y_3^5 + k_4^8 + 5y_4^8
\end{aligned}$$

Doing products modulo 11, and reordering by plaintext p-its, key p-its, and S-boxes output p-its (where $p = 11$).

$$
\begin{aligned}
5u^1 &+ 5u^4 + 7u^5 + 7u^8 \\
&+5k_1^1 + 5k_1^4 + 7k_1^5 + 7k_1^8 + 4k_2^1 + 4k_2^4 + 6k_3^1 + 6k_3^4 + 2k_3^5 + 2k_3^8 + k_4^1 + k_4^4 + k_4^5 + k_4^8 \\
&+3y_1^1 + 3y_1^4 + 2y_1^5 + 2y_1^8 + 8y_1^1 + 9y_1^8 + 8y_1^4 + 9y_1^5 \\
&+9y_2^1 + 9y_2^4 + y_2^1 + 8y_2^8 + y_2^4 + 8y_2^5 + 2y_2^1 + 3y_2^8 + 2y_2^4 + 3y_2^5 \\
&+8y_3^1 + 8y_3^4 + 10y_3^5 + 10y_3^8 + 2y_3^1 + 5y_3^8 + 2y_3^4 + 5y_3^5 + y_3^1 + 7y_3^8 + y_3^4 + 7y_3^5 \\
&+5y_4^1 + 5y_4^4 + 5y_4^5 + 5y_4^8
\end{aligned}
$$

Grouping the S-boxes output p-its (where $p = 11$).

$$
\begin{aligned}
5u^1 &+ 5u^4 + 7u^5 + 7u^8 \\
&+5k_1^1 + 5k_1^4 + 7k_1^5 + 7k_1^8 + 4k_2^1 + 4k_2^4 + 6k_3^1 + 6k_3^4 + 2k_3^5 + 2k_3^8 + k_4^1 + k_4^4 + k_4^5 + k_4^8 \\
&+(3y_1^1 + 8y_1^1) + (3y_1^4 + 8y_1^4) + (2y_1^5 + 9y_1^5) + (2y_1^8 + 9y_1^8) \\
&+(9y_2^1 + y_2^1 + 2y_2^1) + (9y_2^4 + y_2^4 + 2y_2^4) + (8y_2^5 + 3y_2^5) + (8y_2^8 + 3y_2^8) \\
&+(8y_3^1 + 2y_3^1 + y_3^1) + (8y_3^4 + 2y_3^4 + y_3^4) + (10y_3^5 + 5y_3^5 + 7y_3^5) + (10y_3^8 + 5y_3^8 + 7y_3^8) \\
&+5y_4^1 + 5y_4^4 + 5y_4^5 + 5y_4^8
\end{aligned}
$$

Doing the additions modulo 11.

$$
\begin{aligned}
5u^1 &+ 5u^4 + 7u^5 + 7u^8 \\
&+5k_1^1 + 5k_1^4 + 7k_1^5 + 7k_1^8 + 4k_2^1 + 4k_2^4 + 6k_3^1 + 6k_3^4 + 2k_3^5 + 2k_3^8 + k_4^1 + k_4^4 + k_4^5 + k_4^8 \\
&+\cancel{11y_1^1} + \cancel{11y_1^4} + \cancel{11y_1^5} + \cancel{11y_1^8} \\
&+\cancel{11y_2^1} + \cancel{11y_2^4} + \cancel{11y_2^5} + \cancel{11y_2^8} \\
&+\cancel{11y_3^1} + \cancel{11y_3^4} + \cancel{22y_3^5} + \cancel{22y_3^8} \\
&+5y_4^1 + 5y_4^4 + 5y_4^5 + 5y_4^8
\end{aligned}
$$

Replacing the y_4^i by expressions inolving v_5^i and k_5^i.

$$
\begin{aligned}
5u^1 &+ 5u^4 + 7u^5 + 7u^8 \\
&+5k_1^1 + 5k_1^4 + 7k_1^5 + 7k_1^8 + 4k_2^1 + 4k_2^4 + 6k_3^1 + 6k_3^4 + 2k_3^5 + 2k_3^8 + k_4^1 + k_4^4 + k_4^5 + k_4^8 \\
&+5[2(v_5^1 - k_5^1) - 3(v_5^5 - k_5^5)] \\
&+5[2(v_5^4 - k_5^4) - 3(v_5^8 - k_5^8)] \\
&+5[10(v_5^8 - k_5^8) - 5(v_5^4 - k_5^4)] \\
&+5[10(v_5^5 - k_5^5) - 5(v_5^1 - k_5^1)]
\end{aligned}
$$

Doing products modulo 11, and expanding the expressions.

$$
\begin{aligned}
5u^1 &+ 5u^4 + 7u^5 + 7u^8 \\
&+5k_1^1 + 5k_1^4 + 7k_1^5 + 7k_1^8 + 4k_2^1 + 4k_2^4 + 6k_3^1 + 6k_3^4 + 2k_3^5 + 2k_3^8 + k_4^1 + k_4^4 + k_4^5 + k_4^8 \\
&+10v_5^1 - 10k_5^1 - 4v_5^5 + 4k_5^5 \\
&+10v_5^4 - 10k_5^4 - 4v_5^8 + 4k_5^8 \\
&+6v_5^8 - 6k_5^8 - 3v_5^4 + 3k_5^4 \\
&+6v_5^5 - 6k_5^5 - 3v_5^1 + 3k_5^1
\end{aligned}
$$

Doing the additions modulo 11.

$$
\begin{aligned}
&5u^1 + 5u^4 + 7u^5 + 7u^8 \\
&\quad +5k_1^1 + 5k_1^4 + 7k_1^5 + 7k_1^8 + 4k_2^1 + 4k_2^4 + 6k_3^1 + 6k_3^4 + 2k_3^5 + 2k_3^8 + k_4^1 + k_4^4 + k_4^5 + k_4^8 \\
&\quad +7v_5^1 + 4k_5^1 + 2v_5^5 + 9k_5^5 \\
&\quad +7v_5^4 + 4k_5^4 + 2v_5^8 + 9k_5^8
\end{aligned}
$$

Reordering by plaintext p-its, key p-its, and round 5 S-boxes input p-its (where $p = 11$).

$$
\begin{aligned}
&5u^1 + 5u^4 + 7u^5 + 7u^8 \\
&\quad +5k_1^1 + 5k_1^4 + 7k_1^5 + 7k_1^8 \\
&\quad +4k_2^1 + 4k_2^4 \\
&\quad +6k_3^1 + 6k_3^4 + 2k_3^5 + 2k_3^8 \\
&\quad +k_4^1 + k_4^4 + k_4^5 + k_4^8 \\
&\quad +4k_5^1 + 4k_5^4 + 9k_5^5 + 9k_5^8 \\
&\quad +7v_5^1 + 7v_5^4 + 2v_5^5 + 2v_5^8
\end{aligned}
$$

Suppose the key p-its are fixed, we are left with the following relation.

$$5u^1 + 5u^4 + 7u^5 + 7u^8 + 7v_5^1 + 7v_5^4 + 2v_5^5 + 2v_5^8$$

Linear cryptanalysis full relation verification

The following is a Matsui-like algorithm to obtain the most probable 6th round key. It uses a known key and randomly generated plaintexts so that it is possible to check the result.

Python + SageMath

```python
import collections
import itertools

true_key = vector(field, [5, 3, 9, 0, 1, 2, 8, 6])

num_pairs = 5000
plaintexts = [random_vector(field, blocksize) for _ in range(num_pairs)]
ciphertexts = [enc_aeslike_nearly_linear(true_key, plaintext) for plaintext in
plaintexts]

counter = collections.Counter()
for plaintext, ciphertext in zip(plaintexts, ciphertexts):
    for candidate_subkey in itertools.product(range(p), repeat=4):
        candidate_subkey = vector(field, candidate_subkey)
        y51 = candidate_subkey[0] + ciphertext[0]
        y54 = candidate_subkey[1] + ciphertext[3]
        y55 = candidate_subkey[2] + ciphertext[4]
        y58 = candidate_subkey[3] + ciphertext[7]
        v51 = inverse_substitution_table[y51]
        v54 = inverse_substitution_table[y54]
        v55 = inverse_substitution_table[y55]
        v58 = inverse_substitution_table[y58]
        z = (5 * plaintext[0] + 5 * plaintext[3]
             + 7 * plaintext[4] + 7 * plaintext[7]
             + 7 * v51 + 7 * v54 + 2 * v55 + 2 * v58)
        if z == 0:
            candidate_subkey.set_immutable()
            counter.update([candidate_subkey])

```

```python
counts = dict(counter)
max_count = -1
for candidate_subkey in counts:
    current_count = counts[candidate_subkey]
    counts[candidate_subkey] = abs(current_count - num_pairs/2)
    if counts[candidate_subkey] > max_count:
        max_count = counts[candidate_subkey]
        most_probable_subkey = candidate_subkey

print(f'Most probable subkey: {most_probable_subkey}')
```