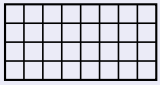
Efficient linear layers for Spook

Gaëtan Leurent

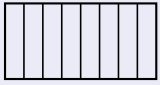
Spook day

LS-Designs

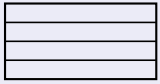
LS-Designs



State as a bit-matrix



S-box layer

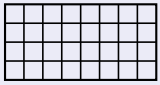


L-box layer

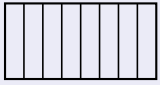
- Clean design for lightweight ciphers
  - Robin & Fantomas [FSE '14]
  - PRIDE [Crypto '14]
  - SCREAM & iSCREAM [CAESAR candidate]
  - ...
- Orthogonal non-linear layer and linear layer
  - S-Box with good differential and linear properties
  - L-Box with branch number $\mathcal{B}$
  - $\mathcal{B}$ active S-Boxes every two rounds
- In practice
  - Bitslice S-Box
    - One line of state mapped to one register
  - Efficient L-Box
    - Table-based (Robin / Fantomas / SCREAM / iSCREAM)
    - Sequence of instructions (PRIDE)

LS-Designs

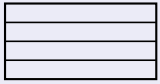
LS-Designs



State as a bit-matrix



S-box layer



L-box layer

In practice: Robin

```
void C13(uint16_t X[4], uint16_t Y[4]) {
  uint16_t a, b, c, d;
  Y[0] ^= a = (X[0] & X[1]) ^ X[2];
  Y[2] ^= c = (X[1] | X[2]) ^ X[3];
  Y[3] ^= d = (  a  & X[3]) ^ X[0];
  Y[1] ^= b = (  c  & X[0]) ^ X[1];
}
#define Sbox(x) C13(x+4, x), C13(x, x+4), C13(x+4, x)
extern uint16_t L1[256], L2[256];

void Encrypt(uint16_t x[8], uint16_t k[8]) {
  for (int j=0; j<8; j++) x[j] ^= k[j];      // Key adition
  for (int i=0; i<16; i++) {
    x[0] ^= L1[i+1];                         // Round constant
    Sbox(x);                                 // S-box
    for (int j=0; j<8; j++) {
      x[j]  = L2[x[j]>>8] ^ L1[x[j]&0xff];   // L-box
      x[j] ^= k[j];                          // Key adition
} } }
```

Maximum branch number

Differential Branch Number

- $\mathcal{B}_d(L) = \min_{x \neq 0}\{w(x) + w(L(x))\}$
- Equivalently, minimum distance of code generated by $[I \mid L]$

Linear Branch Number

- $\mathcal{B}_l(L) = \min_{x \neq 0}\{w(x) + w(L^\top(x))\}$
- Equivalently, minimum distance of code generated by $\left[I \mid L^\top\right]$

- When implemented as tables, arbitrary L-Box can be used
- Bounds on $\mathcal{B}$ from coding theory

n	8	16	32	64	128
upper bound	5	8	16	28	56
best known	5	8	12	22	38

Efficient linear layers

- In order to avoid tables, use L-Box with efficient implementation
- PRIDE considered circulant matrices, to be implementated with rotations and XORs

n	8	16	32	64	128
upper bound	5	8	16	28	56
best known	5	8	12	22	38
best circulant found	4*	8*	12*	20	38?

* exhaustive search

- Implementation cost?
  - Naive: hamming weight
  - Optimal: reuse values

Example: 16 bits

- Optimal branch number: $\mathcal{B} = 8$
- Can be reached with weight 7 (also for inverse)
- Example: circ(0000000101101111) [Pride]
  - Naive implementation: 6 ROT + 6 XOR
  - $x \oplus \text{rot}(x, 1) \oplus \text{rot}(x, 2) \oplus \text{rot}(x, 3) \oplus \text{rot}(x, 5) \oplus \text{rot}(x, 6) \oplus \text{rot}(x, 8)$
- Optimal implementation: 4 ROT + 4 XOR

$L_{16} = \text{circ}(0111100000001011)$

Input: x
Word size: 16
$a = x \oplus \text{rot}(x, 2)$
$b = a \oplus \text{rot}(a, 1)$
$a = x \oplus \text{rot}(a, 1)$
return $a \oplus \text{rot}(b, 11)$

$L_{16}^{-1} = \text{circ}(0011111000001001)$

Input: x
Word size: 16
$a = x \oplus \text{rot}(x, 1)$
$a = a \oplus \text{rot}(x, 4)$
$b = a \oplus \text{rot}(a, 2)$
return $\text{rot}(a, 15) \oplus \text{rot}(b, 9)$

Example: 16 bits

- Optimal branch number: $\mathcal{B} = 8$
- Can be reached with weight 7 (also for inverse)
- Example: circ(0000000101101111) [Pride]
  - Naive implementation: 6 ROT + 6 XOR
  - $x \oplus \text{rot}(x, 1) \oplus \text{rot}(x, 2) \oplus \text{rot}(x, 3) \oplus \text{rot}(x, 5) \oplus \text{rot}(x, 6) \oplus \text{rot}(x, 8)$
- Optimal implementation: 4 ROT + 4 XOR

$L_{16} = \text{circ}(0111100000001011)$

Input: x
Word size: 16
$a = x \oplus \text{rot}(x, 2)$
$b = a \oplus \text{rot}(a, 1)$
$a = x \oplus \text{rot}(a, 1)$
return $a \oplus \text{rot}(b, 11)$

$L_{16}^{-1} = \text{circ}(0011111000001001)$

Input: x
Word size: 16
$a = x \oplus \text{rot}(x, 1)$
$a = a \oplus \text{rot}(x, 4)$
$b = a \oplus \text{rot}(a, 2)$
return $\text{rot}(a, 15) \oplus \text{rot}(b, 9)$

Search strategy

1. Search over sequences of ROT/XOR → list of efficient L
   - Exhaustive for small n
   - Randomized for larger n
   - Match L and L^{-1} (memory-less for large n)

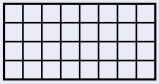
2. Test branch number

for all $x : w(x) < b/2$ **do**
  if $w(x) + w(L(x)) < b$ **or** $w(x) + w(L^{-1}(x)) < b$ **then**
    return 0
return 1

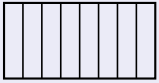
   - Optimizations
     - Use `PCLMUL` instruction
     - Limit x up to rotation equivalence

Interleaved LS-Designs

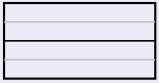
ILS-Designs



State as a bit-matrix



S-box layer



L-box layer

- LS-Design: L-Box independently on each line
  - In terms of SPN: diffusion layer with binary coefficients
- Mixing bits of each S-Box allows larger branch number
  - Code over a larger alphabet
- For efficiency, focus on circulant matrices
  - $(a, b) = L(x, y) = \begin{pmatrix} \text{circ}(????) \cdot x^\top \oplus \text{circ}(????) \cdot y^\top \\ \text{circ}(????) \cdot x^\top \oplus \text{circ}(????) \cdot y^\top \end{pmatrix}$
- Further reduction: interleaved
  - $\text{Interleave}(a, b) = \text{circ}(????) \cdot \text{Interleave}(x^\top, y^\top)$
  - Reuse previous search: 2n bits, modified weight
  - 2n-bit ROT/XOR can be rewritten on n-bit x, y

Results

- Linear layer with low implementation cost and high branch number

n	8	16	32	64	128
upper bound	5	8	16	28	56
best known	5	8	12	22	38
best circulant found	4	8	12	20	38?
cost		4 X+R	5 X+R	6 X+R	
best interleaved found		10	16		
cost		5 X+R	6 X+R		

- Estimated cycle counts on Cortex-M0 and M3 (guesstimate...)

Table (M0)	ROT+XOR (M0)	Table (M3)	ROT+XOR (M3)		
n = 16, circulant	(2 bytes)	9	16	6	8
n = 32, circulant	(4 bytes)	19	10	12	5
n = 32, interleaved	(8 bytes)	46	24	32	12

Results

- Linear layer with low implementation cost and high branch number

n	8	16	32	64	128
upper bound	5	8	16	28	56
best known	5	8	12	22	38
best circulant found	4	8	12	20	38?
cost		4 X+R	5 X+R	6 X+R	
best interleaved found		10	16		
cost		5 X+R	6 X+R		

- Estimated cycle counts on Cortex-M0 and M3 (guesstimate...)

Table (M0)	ROT+XOR (M0)	Table (M3)	ROT+XOR (M3)		
n = 16, circulant	(2 bytes)	9	16	6	8
n = 32, circulant	(4 bytes)	19	10	12	5
n = 32, interleaved	(8 bytes)	46	24	32	12

Spook L-Box

$L_{2\times32}$

Input: (x, y)
Word size: 32

$a = x \oplus \mathrm{rot}(x, 12)$
$b = y \oplus \mathrm{rot}(y, 12)$
$a = a \oplus \mathrm{rot}(a, 3)$
$b = b \oplus \mathrm{rot}(b, 3)$
$a = a \oplus \mathrm{rot}(x, 17)$
$b = b \oplus \mathrm{rot}(y, 17)$
$c = a \oplus \mathrm{rot}(a, 31)$
$d = b \oplus \mathrm{rot}(b, 31)$
$a = a \oplus \mathrm{rot}(d, 26)$
$b = b \oplus \mathrm{rot}(c, 25)$
$a = a \oplus \mathrm{rot}(c, 15)$
$b = b \oplus \mathrm{rot}(d, 15)$
return (a, b)

$L^{-1}_{2\times32}$

Input: (x, y)
Word size: 32

$a = x \oplus \mathrm{rot}(x, 25)$
$b = y \oplus \mathrm{rot}(y, 25)$
$c = x \oplus \mathrm{rot}(a, 31)$
$d = y \oplus \mathrm{rot}(b, 31)$
$c = c \oplus \mathrm{rot}(a, 20)$
$d = d \oplus \mathrm{rot}(b, 20)$
$a = c \oplus \mathrm{rot}(c, 31)$
$b = d \oplus \mathrm{rot}(d, 31)$
$c = c \oplus \mathrm{rot}(b, 26)$
$d = d \oplus \mathrm{rot}(a, 25)$
$a = a \oplus \mathrm{rot}(c, 17)$
$b = b \oplus \mathrm{rot}(d, 17)$
return $(\mathrm{rot}(a, 16), \mathrm{rot}(b, 16))$