Meet in the Middle - STP

March 20, 2019

Last week's exercise

Solution on whiteboard.

Recap of MitM attack

Whiteboard

Searching for attacks

- By hand - Last week(s)
- Using the computer - This week

Searching for attacks

- By hand - Last week(s)
- Using the computer - This week
  - Excel
  - Tailored program
  - STP - Simple Theorem Prover
  - MILP - Mixed Integer Linear Programming

STP

- Can be used to prove certain properties of a system.
- Constraint Solver.
- Quantifier free.
- Bitvectors.
- Many input languages, we will use CVC (Least annoying).

STP (2)

We can give a set of constraints to STP and ask if the set of constraints is satisfiable.

$$x = 5$$
$$y = 6$$
$$x > y$$

STP (2)

We can give a set of constraints to STP and ask if the set of constraints is satisfiable.

$$x = 5$$
$$y = 6$$
$$x > y$$

Is unsatisfiable.

$$x = 0x5$$
$$y \in \{0, 1\}^4$$
$$z = x \oplus y$$
$$z = 0xF$$

STP (2)

We can give a set of constraints to STP and ask if the set of constraints is satisfiable.

$$x = 5$$
$$y = 6$$
$$x > y$$

Is unsatisfiable.

$$x = 0x5$$
$$y \in \{0, 1\}^4$$
$$z = x \oplus y$$
$$z = 0xF$$

Is satisfiable.

CVC

```
% INPUT
x, y, z: BITVECTOR(4);
ASSERT(
x = 0hex5 AND
z = BVXOR(x, y) AND
z = 0hexF
);

QUERY(FALSE);
COUNTEREXAMPLE;
```

CVC

```
% INPUT
x, y, z: BITVECTOR(4);
ASSERT(
x = 0hex5 AND
z = BVXOR(x, y) AND
z = 0hexF
);

QUERY(FALSE);
COUNTEREXAMPLE;
```

```
% OUTPUT
ASSERT( y = 0xA );
ASSERT( z = 0xF );
ASSERT( x = 0x5 );
Invalid.
```

CVC (2)

```
% INPUT
x, y, z: BITVECTOR(4);
ASSERT(
    % x is non zero
    NOT ( x = 0hex0 ) AND
    % y is zero
    y = 0hex0 AND
    % set a constraint on z
    z = x & ((y << 2)[3:0]) AND
    % assert that z is nonzero
    NOT (z = 0hex0)
);

QUERY(FALSE);
COUNTEREXAMPLE;
```

CVC (2)

```
% INPUT
x, y, z: BITVECTOR(4);
ASSERT(
    % x is non zero
    NOT ( x = 0hex0 ) AND
    % y is zero
    y = 0hex0 AND
    % set a constraint on z
    z = x & ((y << 2)[3:0]) AND
    % assert that z is nonzero
    NOT (z = 0hex0)
);

QUERY(FALSE);
COUNTEREXAMPLE;
```

```
% OUTPUT
Valid.
```

CVC (2.5)

```
% INPUT
x, y, z: BITVECTOR(4);
ASSERT(
    % x is non zero
    NOT ( x = 0hex0 ) AND
    % y is zero
    NOT(y = 0hex0) AND
    % set a constraint on z
    z = x & ((y << 2)[3:0]) AND
    % assert that z is nonzero
    NOT (z = 0hex0)
);

QUERY(FALSE);
COUNTEREXAMPLE;
```

CVC (2.5)

```
% INPUT
x, y, z: BITVECTOR(4);
ASSERT(
    % x is non zero
    NOT ( x = 0hex0 ) AND
    % y is zero
    NOT(y = 0hex0) AND
    % set a constraint on z
    z = x & ((y << 2)[3:0]) AND
    % assert that z is nonzero
    NOT (z = 0hex0)
);

QUERY(FALSE);
COUNTEREXAMPLE;
```

```
% OUTPUT
ASSERT( x = 0x4 );
ASSERT( y = 0x1 );
ASSERT( z = 0x4 );
Invalid.
```

CVC (3)

For more information on STP and CVC: `https://github.com/stp/stp/blob/master/docs/cvc-input-language.rst`

<table>
<tr><th>CVC</th><th>normal</th></tr>
<tr><td>AND / OR / NOT</td><td>&& / || / !</td></tr>
<tr><td>| / & / ˜</td><td>| / & / ˜</td></tr>
<tr><td>BVXOR(a, b)</td><td>a ˆb</td></tr>
<tr><td>BVPLUS(a, b)</td><td>$a + b$</td></tr>
<tr><td>BVMULT(a, b)</td><td>$a * b$</td></tr>
<tr><td>BVSUB(a, b)</td><td>$a - b$</td></tr>
</table>

CVC	normal
0hex5/0bin0110	0x5/0b0110
x :BITVECTOR(n)	$x \in \{0, 1\}^n$
a @ b	concatenation
a[4:1]	extraction
<<	left shift
>>	right shift

TC03

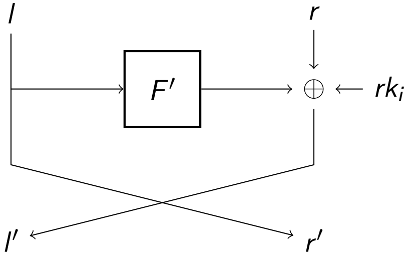
TC03 is a Feistel network with a block size of 8 bits, and a key size of 64-bit.

Round Function

$F'(w) = ((w \lll 1) \& (w \lll 2)) \oplus w$

Key Schedule

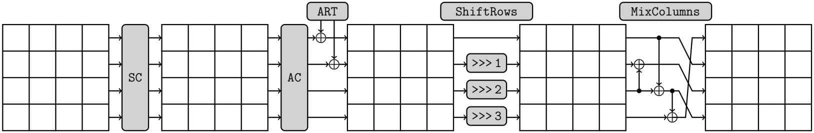
$K = k_0 | k_1 | k_2 | k_3 | \ldots | k_{15}$
The i-th round key is given by: $rk_i = k_{(i \mod 16)}$



CVC (4)

- Overkill for finding MitM attacks, but is interesting for finding differential/linear charactersitics.
- Very verbose (no quantifiers).
- Write a python script to create CVC description of the cipher.

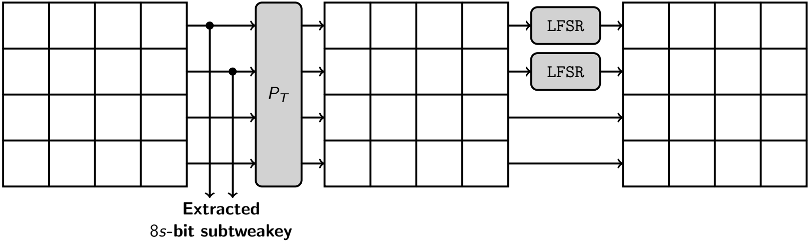
SKINNY Round Function



$$S_4 = [\texttt{C 6 9 0 1 A 2 B 3 8 5 D 4 E 7 F}]$$

$$M = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

SKINNY Tweakey Schedule



$$P_T = [9\ 15\ 8\ 13\ 10\ 14\ 12\ 11\ 0\ 1\ 2\ 3\ 4\ 5\ 6\ 7]$$

$$\mathtt{LFSR}_{TK2} = (x_3||x_2||x_1||x_0) \rightarrow (x_2||x_1||x_0||x_3 \oplus x_2)$$

Skinny with STP

- Model knowledge on nibble level instead of bitlevel.
- Also model the Key schedule.
- Upperbound the key weight to find 'best' attacks.
- We can find all attacks by removing instances from the search space and retrying until no valid attacks are possible.

The End?

- STP is powerfull, but for example getting the minimum number of keybits is not (natively) possible. Better to use MILP (Mixed Integer Linear Programming).
- MitM attacks are powerful, but as we will see next week there exist better attacks (more rounds).
- Only the basics of MitM attacks, we can squeeze out a bit more if we really want.

For nextnext week

- Next week **no** class!
- Do this weeks exercises (deadline 3rd of april).
- Play a bit with STP (Hint: If you find your attack on `TC02` with STP you get extra points).