Solving Low-Compexity Ciphers with Optimized SAT Solvers

UNIVERSITY of VIRGINIA

Karsten Nohl, Mate Soos

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE | INRIA

Premise: **SAT solvers can invert weak cryptographic ciphers efficiently when both the solver and the cipher representation are tweaked**

* Cheap, proprietary cryptographic functions are common in embedded applications such as RFID tickets, cell phones, and theft protection
* These ciphers are often found to be weak to hand-crafted statistical and algebraic attacks
* Algebraic attacks are a powerful tool for finding cryptographic weaknesses such as insufficient non-linearity of diffusion
* **SAT solver can be an automated cryptanalysis tool**. Some of SAT solvers' potential is unlocked through the following two sets of ideas

1 Extend MiniSat and undersand its behaviour

MiniSat does not understand the XOR function natively, so we extended its input language

$x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6$

$\downarrow$

$\bar{y_1} \oplus x_1 \oplus x_2 \oplus x_3$

$\bar{y_2} \oplus x_4 \oplus x_5 \oplus x_6$

$y_1 \oplus y_2$

$a \oplus b \oplus c$

$\downarrow$

$a \vee b \vee c$ (1)

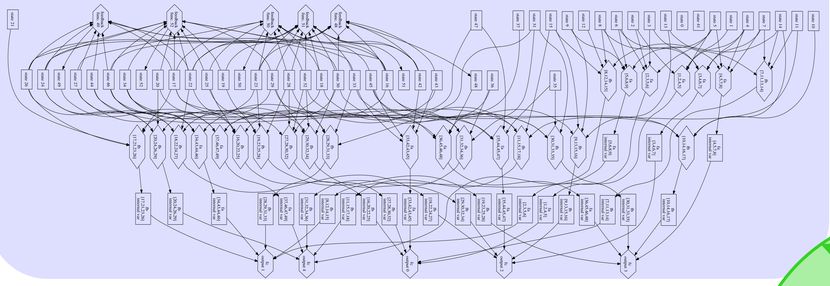
$a \vee \bar{b} \vee c$ (2)

$a \vee b \vee \bar{c}$ (3)

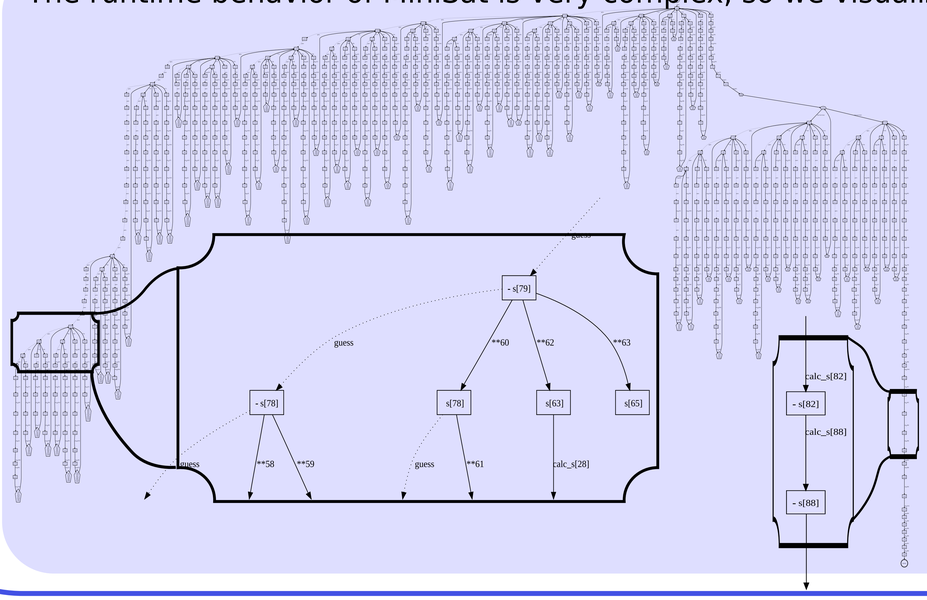
$a \vee \bar{b} \vee \bar{c}$ (4)

Need to find a good mix of xor-clauses and standard CNF

It's difficult to know how the cipher is represented inside MiniSat, so visualized the relationship between clauses and variables



The runtime behavior of MiniSat is very complex, so we visualized it



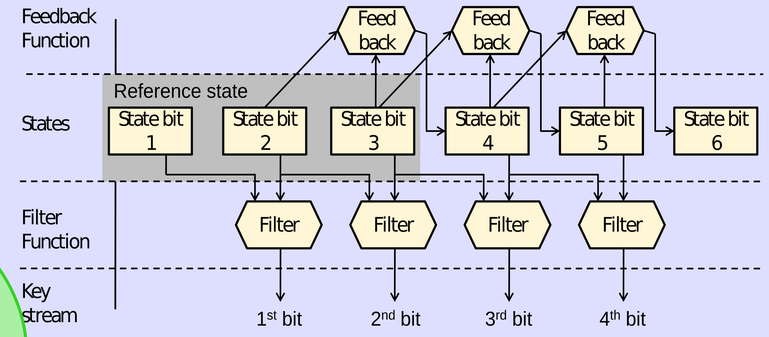
Calculated statistics :
* Average depth
* Most conflicted clauses
* No of guesses/branches
* Most guessed variables
* Most propagated clauses

Visualized details:
* Guesses
* Propagations
* Clause groups
* Generated leart clauses
* Clause groups causing the propagation

Several Iterations improve the attacks

2 Tweak the problem statement i.e. the cipher representation

To give more structural information to the solver, we represent the problem as a Logical circuit



* SAT solving becomes faster as more structural information is included in the problem
* Therefore, we model a cipher as a circuit instead of a group of unconnected functions

Stats calculated:
* Depth
* Dependency number
* Function complexity

To minimize the problem space, we optimized the shift registers ' representation

Reference state of shift registers must be optimized for:
* Least number of dependencies per output
* Least overall circuit depth
* Least number of unused registers

Solution:
reference state set should be in the middle of the register set

Reference

To speed up function evaluation, we optimized the representation of non-linear functions

$x_1 \oplus x_1x_2 \oplus x_2x_3 \oplus x_1x_3$

$x_1 \vee \bar{x}_3$

$\bar{x}_2 \vee x_3$

$x_1 \vee x_2$

Karnaugh-based

$x_1 \oplus i_1 \oplus i_2 \oplus i_3$

$i_1 = x_1x_2$

$i_2 = x_2x_3$

$i_3 = x_1x_3$

Xor-based

A mix of the two is also possible

Use the improvements to solve ciphers

* Using these ideas, **we broke several ciphers faster** than was previously possible, including widely used RFID ciphers and an academic toy cipher
* Next step: Implementing attacks on the standard stream ciphers Trivium and Grain

	Previous best attack	Our attack
Crypto-1	200s	40s
HiTag2	N/A	$2^{14.5}$s
Bivium	$2^{42.7}$s	$2^{36.5}$s