Limits of SAT Solvers in Cryptography

MATE SOOS

PRESENTATION AT CASED

21ST OF JULY 2011

SECURITY RESEARCH LABS

Story Line

1. Introduction to SAT Solvers and Cryptography
2. Advantages of SAT in Crypto
3. Limitations of SAT in Crypto

What is a SAT Solver

Solves a problem in CNF

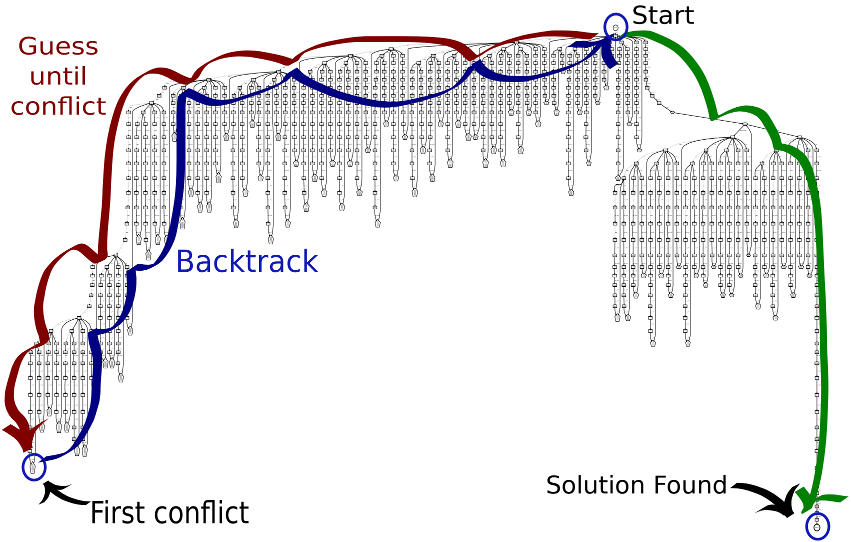
CNF is an "`and` of `or`-s"

$$\neg x_1 \vee \neg x_3 \qquad \neg x_2 \vee x_3 \qquad x_1 \vee x_2$$

Uses DPLL(φ) algorithm

1. If φ trivial, return SAT/UNSAT
2. call DPLL($v \leftarrow$ value)
3. if SAT, output solution
4. if UNSAT, call DPLL($v \leftarrow$ opposite value)
5. if SAT, output solution
6. return UNSAT

Example Search Tree



SAT Solver Internals

- Lazy data structures: watchlists, fast backtracking
- Learning (and forgetting): what to learnt and what to forget?
- Picking variables: which ones to branch on, and in what order?
- Restarts: when to restart, how far to restart

Cryptographic Problems

Stream ciphers

- Generates pseudorandom keystream given public IV and secret key
- Step-by-step iteration easy to describe in ANF
- Easy to model in CNF

Block ciphers

- Encodes a plaintext to a chipertext given a secret key
- Relatively difficult internal parts e.g. S-box
- May be difficult to model in CNF

Hash functions

- Generates one-way, (second)preimage-resistant fingerprint
- Usually has difficult internal parts e.g. adder
- Difficult to model in CNF

Advantages of SAT in Crypto

- Find good points of entry: picking variables
- Partially evaluate the function: lazy data structures
- Effectively store explored search space: learnt clauses

Limitations of SAT in Crypto

Structure lost

- CNFs is "plain" – adders, multipliers not evident
- Higher-level reasoning is very difficult
- Cannot find out that, e.g.
  $v1 \oplus v2 \oplus v3 = \texttt{true}, v1 \oplus v2 \oplus v4 = \texttt{true}$
  $\therefore v3 = v4$
- Gauss elim. needs exponential resolution operations

Probabilities difficult to handle

- All clauses must be true
- Example: $P(v10 \vee v11 \vee v12 = \texttt{true}) = 0.4$. How to model?
- Introduce indicator variable, make it depend on multiple low-probability events

One-to-one Translation has Limited Potential

Past

- One-to-one translation has been tried on many crypto-primitives
- With varying sophistication levels
- Disappointing results on strong primitives (e.g. SHA1, AES, MD5)

Future

- Don't model the algorithm one-to-one
- Model a particular aspect of it, e.g. differential path
- Challenges: what to model, how to model, how to interpret results

Conclusions

Concluding remarks

- SAT solvers can be effective on some crypto problems
- Can break simple cryptographic routines automatically
- But it's far from plug-and-play for complex crypto-primitives

Future work

- Make the plug-and-play experience better for simple problems
- Find crypto-primitive properties to model that could lead to attacks
- Refine properties modelled, refine modelling techniques

Thank you for your time

Any questions?