SAT Solvers and their Applications

You don’t invert matrixes on paper.
So, don’t invert functions on paper!

SAT solvers

- Automated resolution engines
- Extremely fast problem solvers
- Very simple but powerful input format
- SMT: an abstraction over SAT
  - Translates higher-level abstractions (e.g. 32b integers) for input into SAT solver

SMT 101

- Declare variable: a: BITVECTOR(32);
- Set variable: ASSERT(a = 0hex67452301);
- Bitwise AND: ASSERT(c = a & b);
- Addition: ASSERT(d = BVPLUS(32, a, b));
- Static Single Assignment
- Query: Is there a solution? Tell me one/all solutions

What can I use this for?

- LLVM IR -> SMT https://github.com/xiw/stack/blob/master/src/ValueGen.cc
- Can this code path be ever reached? *Goto FAIL?*
- Does this function always return the same value? *LLVM optimization pass*
- Can I create colliding hash values? *Hash lookup of n values: O(n^2) instead of O(nlogn)*

Resources

- STP – Simple Theorem Prover https://github.com/stp/stp
- KLEE – Input generator to explore all code paths https://github.com/klee/klee
- Optimization-unstable code https://github.com/xiw/stack/
- CryptoMiniSat – advanced SAT solver https://github.com/msoos/cryptominisat