PROGRAM ANALYSIS

Manual

versus

Automatic
CORE CONCEPTS

- The Halting Problem
- Rice’s Theorem
- Code Analysis
- Automatic Testing
- Program Introspection
- Code Translation and Representation
- Code Coverage
- Data Flow Tracking
- Abstract Interpretation
- Symbolic Execution
- Constraint Solving
- Formal Verification
SYMBOLIC EXECUTION
CASE STUDY: SAGE
**SAGE: Whitebox Fuzzing for Security Testing**

**Basic idea:**
1. Run the program with first inputs,
2. Gather constraints on inputs at conditional statements,
3. Use a constraint solver to generate new test inputs,
4. Repeat - possibly forever!

**SAGE was developed in collaboration with CSE**

**Impact:** since 2007
- 200+ machine years (in largest fuzzing lab in the world)
- 1 Billion+ constraints (largest SMT solver usage ever!)
- 100s of apps, 100s of bugs (missed by everything else...)
- Ex: 1/3 of all Win7 WEX security bugs found by SAGE
- Bug fixes shipped quietly (no MSRCs) to 1 Billion+ PCs
- Millions of dollars saved (for Microsoft and the world)
- SAGE is now used daily in Windows, Office, etc.

**The SAGE team:**
MSR: E. Bounimova, P. Godefroid, D. Molnar
CSE: M. Levin, Ch. Marsh, L. Fang, S. de Jong,... + thanks to all the SAGE users!
Office: T. Gallagher, E. Jarvi, O. Timofte

**SAGE is the first whitebox fuzzer**

**Research Challenges:**
- How to recover from imprecision? PLDI’05, PLDI’11
- How to scale to billions of x86 instructions? NDSS’08
- How to check many properties together? EMSOFT’08
- How to leverage grammar specifications? PLDI’08
- How to deal with path explosion? POPL’07, TACAS’08
- How to reason precisely about pointers? ISSTA’09
- How to deal with floating-point instr.? ISSTA’10
- How to deal with input-dependent loops? ISSTA’11
+ research on constraint solvers

**How bugs were found (Win7 WEX Security)**
- Regression + Random testing
- All Others
- SAGE
CASE STUDY: KLEE