Joy of Pwning

Sophia d'Antoine
October 26th, 2017

whoami

Masters in CS from
Rensselaer Polytechnic Institute
- Exploiting Intel's CPU pipelines

Work at Trail of Bits
- Senior Security Researcher

DEFCON (CTF), CSAW

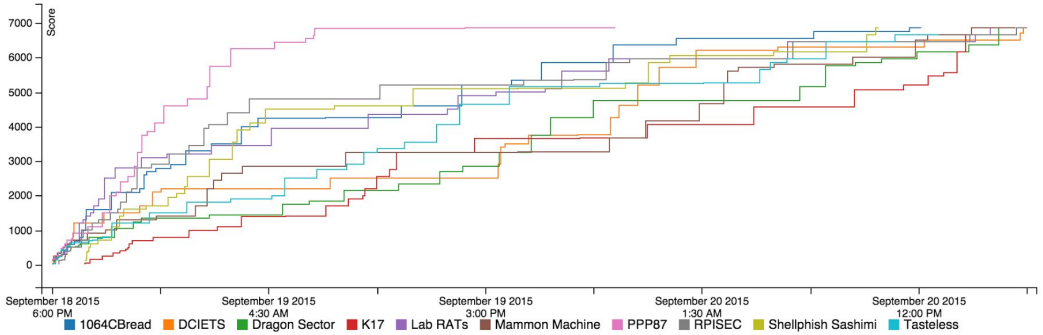
Stats
- 12 Conferences Worldwide
- 3 Program Committees
- 2 Security Panels
- 1 Paper Published
- 1 Keynote

Hacker Training

The World of Wargame & CTFs

CSAW Home Rules Judges Teams Scoreboard Challenges Statistics CSAW.TV Archives Login

Scoreboard



All | Qualifying | High School | Industry | Undergraduate | Graduate | Other

Place	Team	Bracket	Country	Score
1	1064CBread	Undergraduate	United States	6860
2	DCIETS	Undergraduate Stacked	Canada	6860
3	RPISEC	Undergraduate Stacked	United States	6860
4	PPP1	Undergraduate	United States	5810
5	Shellphish Nigiri	Undergraduate Stacked	United States	5810


LEGIT BS-2017

Hacking Competition

- Wargames
  - Several weeks
  - Individual
  - FLARE Reversing Challenges (FLARE team @ FireEye)
- CTFs: 24-36 hours (no sleep)
  - 24-36 hours (no sleep!)
  - Team sport
  - CSAW CTF (Hosted @ NYU)
  - CyberSeed (**$50,000** in prizes)
- Why? Internet points and hacker fame (also skillz)

Further information: “Automatic Exploit Generation, an Odyssey”, CanSecWest 2016

pwnable.kr


input
argv[1]
NABLE.KR
Shell we play a game?
[Grotesque]
fail . . .
3
checks
proxy server
rootkit
dos4fun
ascii
aeg
coin2
maze
wtf
fail . . .
sudoku
starcraft
. . . 15 more
functions . . .
memcpy

Current Status: CTF Field Guide

CTF Field Guide

> “Knowing is not enough; we must apply. Willing is not enough; we must do.” - Johann Wolfgang von Goethe

https://trailofbits.github.io/ctf/

Security Engineering & Tooling

Why do it if you can make a program do it

Post Graduation: Jobs

- Pentesting/ RE/ VR/ Forensics/ IT Sec/ ...
  - Growing Field!
  - Specialization
- Public/ Private Sector
  - Private sector trends
- Security Engineering & Research
  - Tool development to better aid in security
  - Find bugs and exploit automatically
  - ML to detect Network anomalies
  - Static, Dynamic analysis
  - LLVM/ Clang compiler based research <3

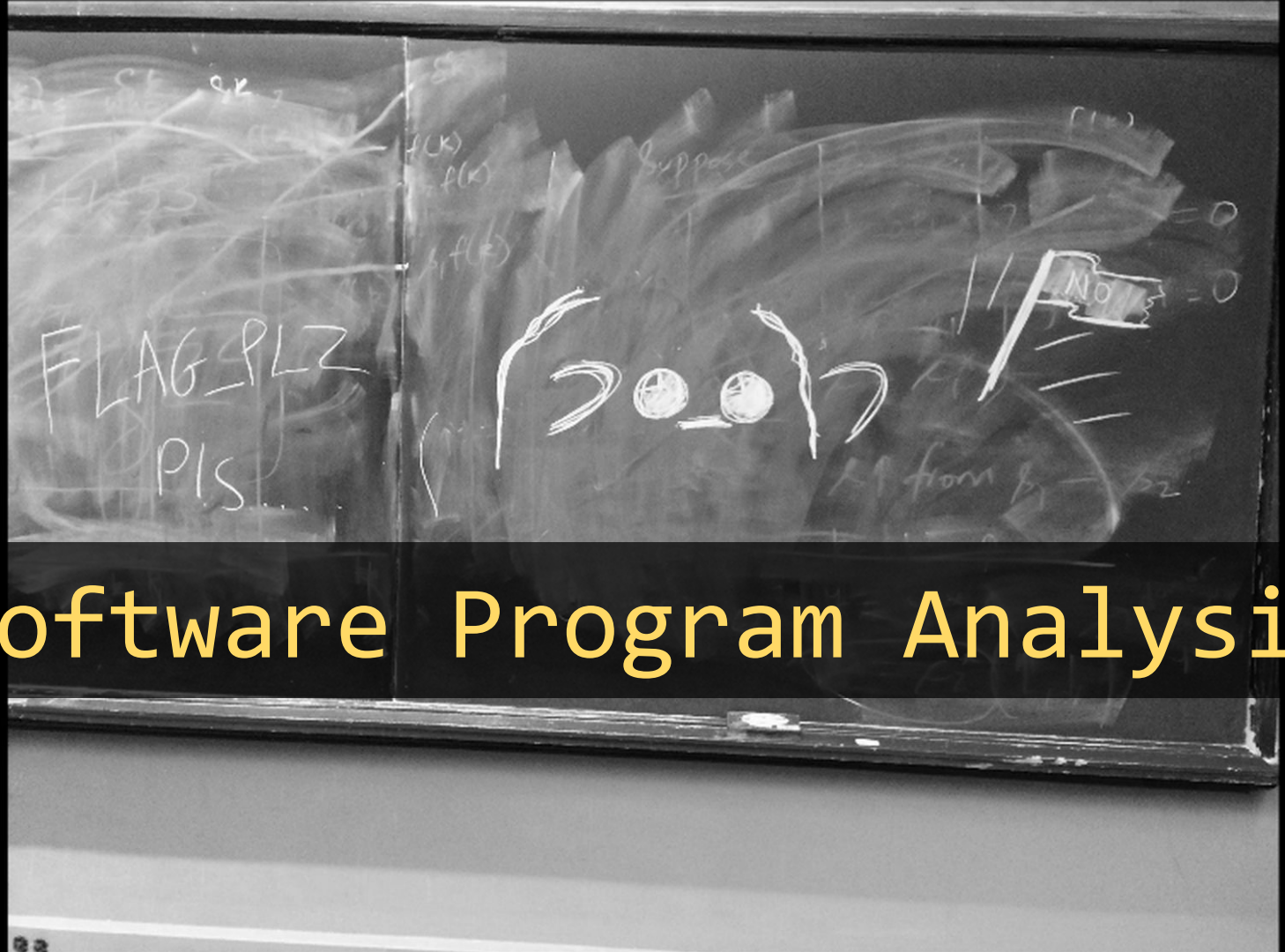
Further information: “Be a Binary Rockstar, an Intro to Program Analysis with Binary Ninja”, Infiltrate 2017

Example: Pintool

```
$pin -t inscount0.so -- binary
```

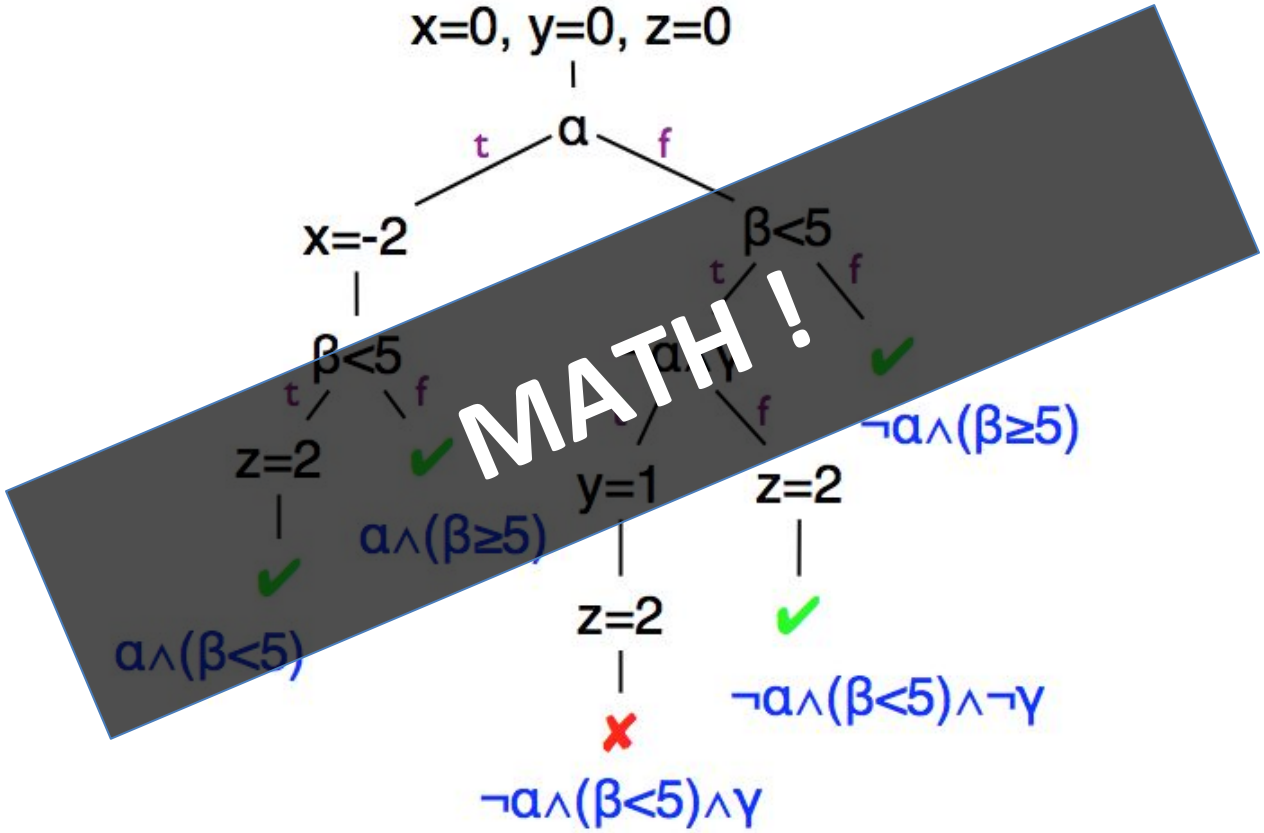
- Binary Level
  - Inject incrementer code after each instruction
- Still Brute Force :<
  - Return total instructions for fuzzed input
  - Only true for that 1 executed path (the possible CFG space may be very large)

Further information: “Be a Binary Rockstar, an Intro to Program Analysis with Binary Ninja”, Infiltrate 2017



Software Program Analysis!

Symbolic/ Concolic Execution



```
[SYMBOL] a, b, c
[INT] x, y, z = 0;
```

```
fun( int a, b, c ){
        if (a) {
            x = -2;
        }

        if (b < 5) {
            if (!a && c) {
                y = 1;
            }
            z = 2;
        }
        assert(x+y+z!=3) }
```

```
. . .
fun( 0, 3, 1 );
. . .
```

Old Method:
Try all inputs until assert

[WARNING] inputs unbounded!

Current Status: Manticore (Symbolic, Concolic)

Manticore: Symbolic execution for humans

POST APRIL 27, 2017 2 COMMENTS

Earlier this week, we open-sourced a tool we rely on for dynamic analysis: Manticore! Manticore helps us quickly take advantage o symbolic execution, taint analysis, and instrumentation to analyze binaries. Parts of Manticore underpinned our symbolic execution capabilities in the Cyber Grand Challenge. As an open-source tool, w ope that others can take advantage of these capabilities in their own proj ts.

We prioritized simplicity and usability while building Manticore. We used minimal external dependencies and our API should look familiar to anyone with an exploitation or reversing background. If you have never used such a tool before, give Manticore a try.

McSema

Interdisciplinary Research

Security research applies to everything


40 UNDER 40: INNOVATORS WHO RULE THE WORLD
SEPT. 1, 2017
FORTUNE.COM
FORTUNE
AND
WHY EVERYONE FROM WALL STREET TO WALMART IS BETTING ON IT
READ!
HOW THIS REVOLUTIONARY TECHNOLOGY IS TRANSFORMING BUSINESS
BLOCK-CHAIN MANIA!
PLUS
HOW THE BITCOIN BOOM TRIGGERED A CRIME SPREE
WARNING
STOCKS ARE WAY TOO EXPENSIVE
INSIDE KEVIN HART'S COMEDY EMPIRE
INTERNATIONAL EDITION / NUMBER 11

Ethereum Smart Contracts

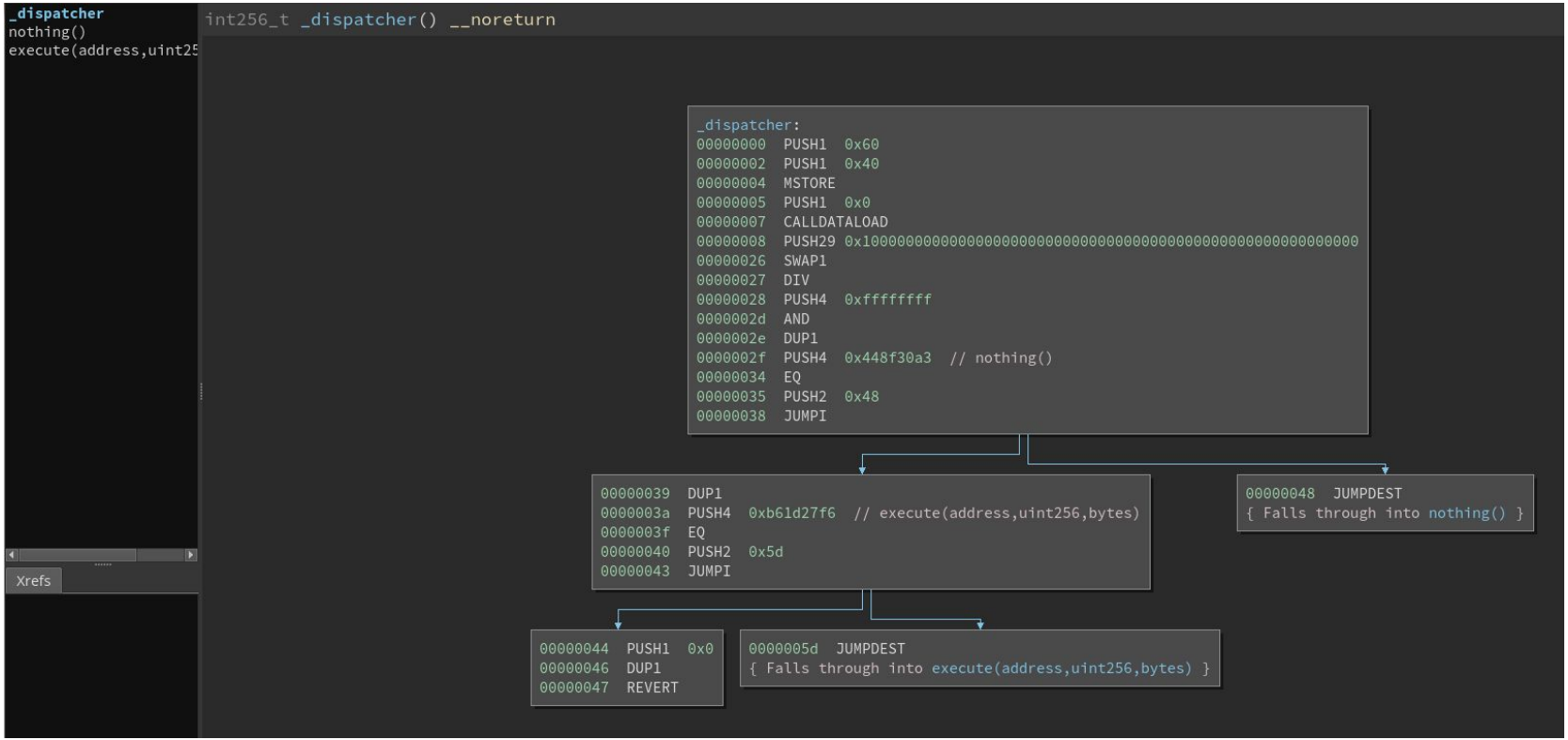
Smart Contracts are Literal Programs!

- **Similar to BTC**
- **... except for Solidity**
  - Simplified Operating Systems (EVM)
  - The EVM is not a register machine but a stack machine
  - Formal Verification is possible
- **Auditing**
  - Find bugs in contracts
  - Possible to hack money
- **DAO attack**
  - "recursive call bug"

Room for New Security Research & Tooling

- Manticore, a symbolic emulator capable of simulating complex multi-contract and multi-transaction attacks against EVM bytecode.
- Not So Smart Contracts, a collection of example Ethereum smart contract vulnerabilities, including code from real smart contracts, useful as a reference and a benchmark for security tools.
- Ethersplay, a graphical Binary Ninja-based EVM disassembler capable of method recovery, dynamic jump computation, source code matching, and bytecode diffing.
- Slither, a static analyzer for the Solidity AST that detects common security issues in reentrancy, constructors, method access, and more.
- Echidna, a property-based tester for EVM bytecode with integrated shrinking that can rapidly find bugs in smart contracts in a manner similar to fuzzing.

ethersplay


_dispatcher
nothing()
execute(address,uint25
Xrefs
int256_t _dispatcher() __noreturn
_dispatcher:
00000000 PUSH1 0x60
00000002 PUSH1 0x40
00000004 MSTORE
00000005 PUSH1 0x0
00000007 CALLDATALOAD
00000008 PUSH29 0x100000000000000000000000000000000000000000000000000000000
00000026 SWAP1
00000027 DIV
00000028 PUSH4 0xffffffff
0000002d AND
0000002e DUP1
0000002f PUSH4 0x448f30a3 // nothing()
00000034 EQ
00000035 PUSH2 0x48
00000038 JUMPI
00000039 DUP1
0000003a PUSH4 0xb61d27f6 // execute(address,uint256,bytes)
0000003f EQ
00000040 PUSH2 0x5d
00000043 JUMPI
00000048 JUMPDEST
{ Falls through into nothing() }
00000044 PUSH1 0x0
00000046 DUP1
00000047 REVERT
0000005d JUMPDEST
{ Falls through into execute(address,uint256,bytes) }

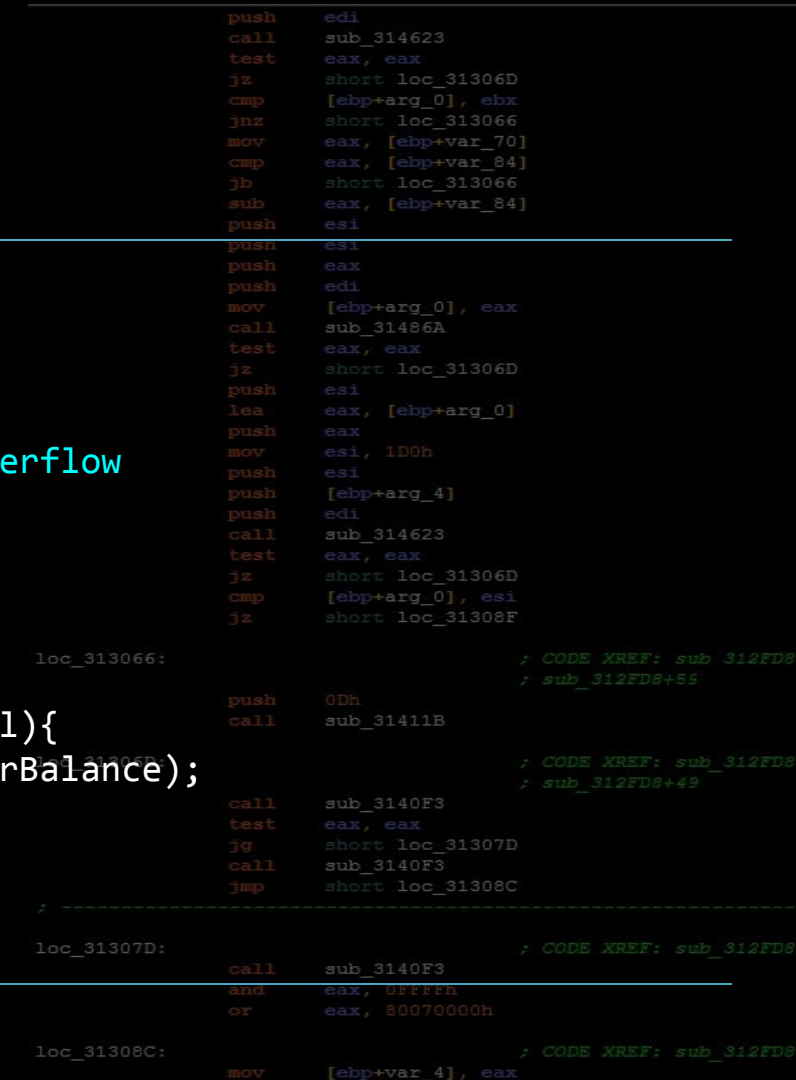
[INT OVERFLOW] Solidity

```
contract Overflow {
    uint private sellerBalance = 0;

    function add(uint value) returns (bool){
        sellerBalance += value; // possible overflow

        // possible auditor assert
        // assert(sellerBalance >= value);
    }

    function safe_add(uint value) returns (bool){
        require(value + sellerBalance >= sellerBalance);
        sellerBalance += value;
    }
}
```



Current Status: Ongoing

Trail of Bits Overview

High-end security research with a real-world attacker mentality

- Security Research & Development firm specializing in:
  - High-assurance software **Development**
  - Low-level software security **Assessments**
  - Applied software security **Research**
- 26 people with offices in NYC, Chicago, Austin, and Toronto
- 10 people with US government security clearances
- Founded in 2012 by 3 expert hackers, no investment capital taken to date

Any Questions?

IRC: quend

email: sophia@trailofbits.com

website: www.sophia.re