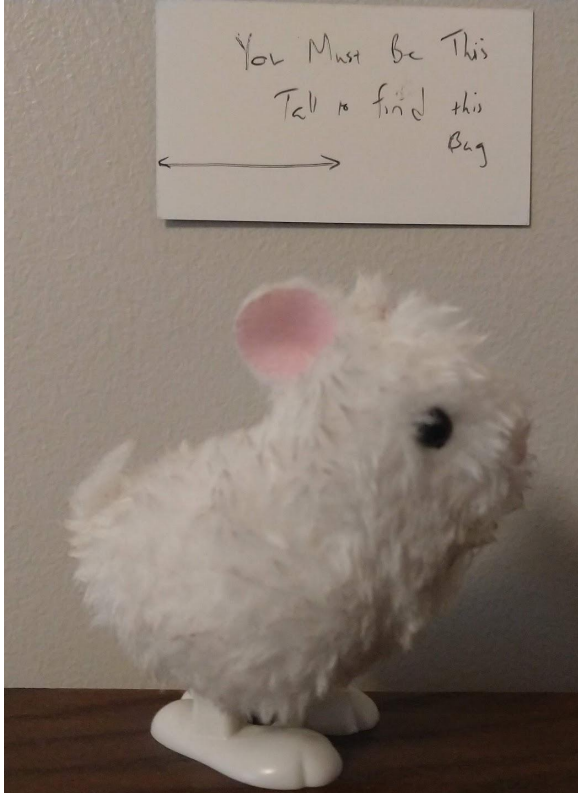
PIPE Cleaner:

Configurable PIPE-Based Policy-Informed Root-Cause Fuzzing

Allison Naaktgeboren
Portland State University



"You must be this tall to find this bug"
Photo: Allison

Terminology Cheat sheet:
https://github.com/anaaktge/talks/blob/main/fuzzingtalkglossary.md

Agenda

Goal: Help you understand…

1. Fuzzing holistically
2. Where PL might help
3. Maybe my own research

Topics

- About Me
- Motivation & What is fuzzing ?
- What is the root cause problem and who cares?
- What is PIPE & TaggedC?
- POC: PVI Violation
- PIPE Cleaner: The Fuzzer I hope to build

Image by https://bishopfox.com/blog/fuzzing-aka-fuzz-testing

Not a [tenure-track] Professor!

Currently: Draper Labs Intern, working on VMF fuzzer

Doctoral Researcher, Portland State, 2020-Present (MS, cybersec)

Under Dr. Andrew Tolmach (QuickChick, SF coauthor)

RPE (Quals) “Sowing the Seeds of Fuzz: Does the Influence of the Initial Seed Corpus Follow a Universal Law?” Jun 22

CTF teams: void * vikings, founder + advisor. 侍 minion

Instructor, Lecturer, Intro to Systems (+ all legal obligations)

BS in CS Carnegie Mellon University, 2004-2008 (SCS ‘08)

Undergraduate research in robotics, roboclub officer

Founding TA 15-440, 15-213, 15-111, 16-311

(Senior) Code Monkey, Hiring Manager, etc ~2009-2019

Cisco, Amazon, Factset, Mozilla (Firefox), Signal Sciences (Fastly)

The best regalia accessories are minions who bring you donuts and coffee
Photo: kbrosnan

My prior PL background…

https://en.wikipedia.org/wiki/Oolong_(rabbit)

Fuzzers are often named after fuzzy things like bunnies

Motivation: Vulnerabilities Suck (and cost $$$)

Security Bulletin: IBM Security Guardium is affected by Open Source libxml2 vulnerabilities

Amazon Linux Security Center

Security Bulletin: Vulnerability in libxml2 affects IBM InfoSphere Streams. (CVE-2015-8317)

Heartbleed bug 'will cost millions'

How much will it cost to secure open-source software? OpenSSF says $147.9M

Affected Packages

Platform
Amazon Linux 2
Amazon Linux 1

From my RPE and guest lecture

Automation to the Rescue? Fuzzing Better than Santa Claus?

The Register®
Biting the hand that feeds IT

Security

Linus Torvalds lauds fuzzing for improving Linux security

But he's not at all keen on Santa Claus or fairies

By Simon Sharwood, APAC Editor 16 Oct 2017 at 07:03

ZDNet MENU US

MUST READ A POPULAR VIRTUAL KEYBOARD APP LEAKS 31 MILLION USERS' PERSONAL DATA

Linux security: Google fuzzer finds ton of holes in kernel's USB subsystem

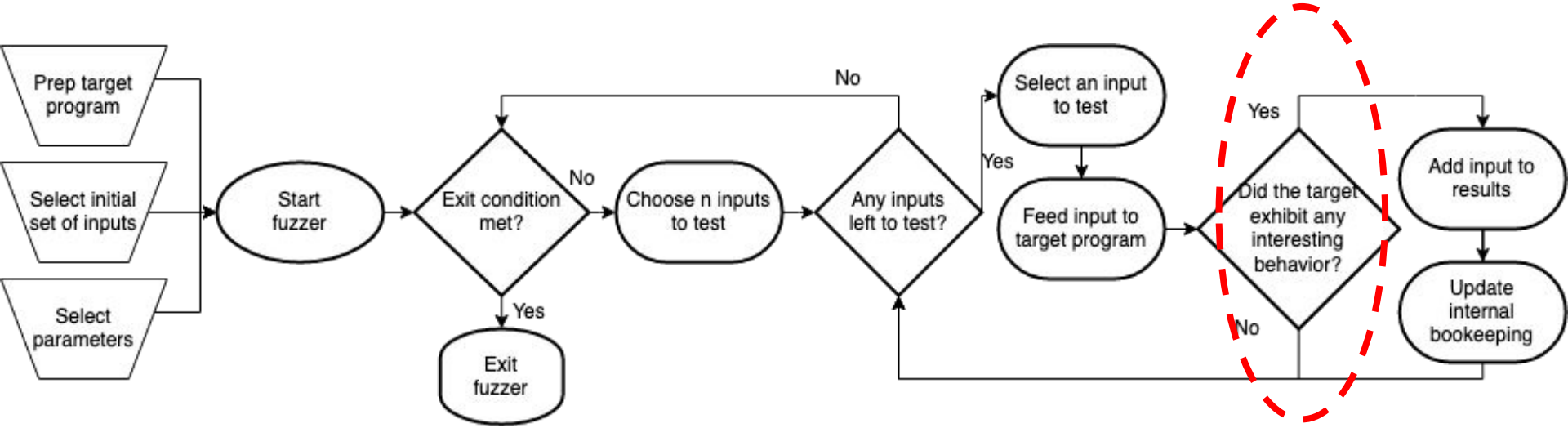
A Google-developed kernel fuzzer has helped locate dozens of Linux security flaws.

By Liam Tung | November 8, 2017 -- 12:43 GMT (04:43 PST) | Topic: Security

From Allison's lecture, CS 492/592 Malware Reverse Engineering

Overview of Fuzzing

Does not include why the target did something interesting!



- Aims to thoroughly explore the input space of the fuzzee (victim, target) looking for inputs (seeds) that cause interesting behavior
- Definition of “interesting” varies. Classic definition is crashes
- A stochastic (probabilistic) dynamic software-testing technique
- Property based testing, symbolic/concolic execution are arguably subclasses

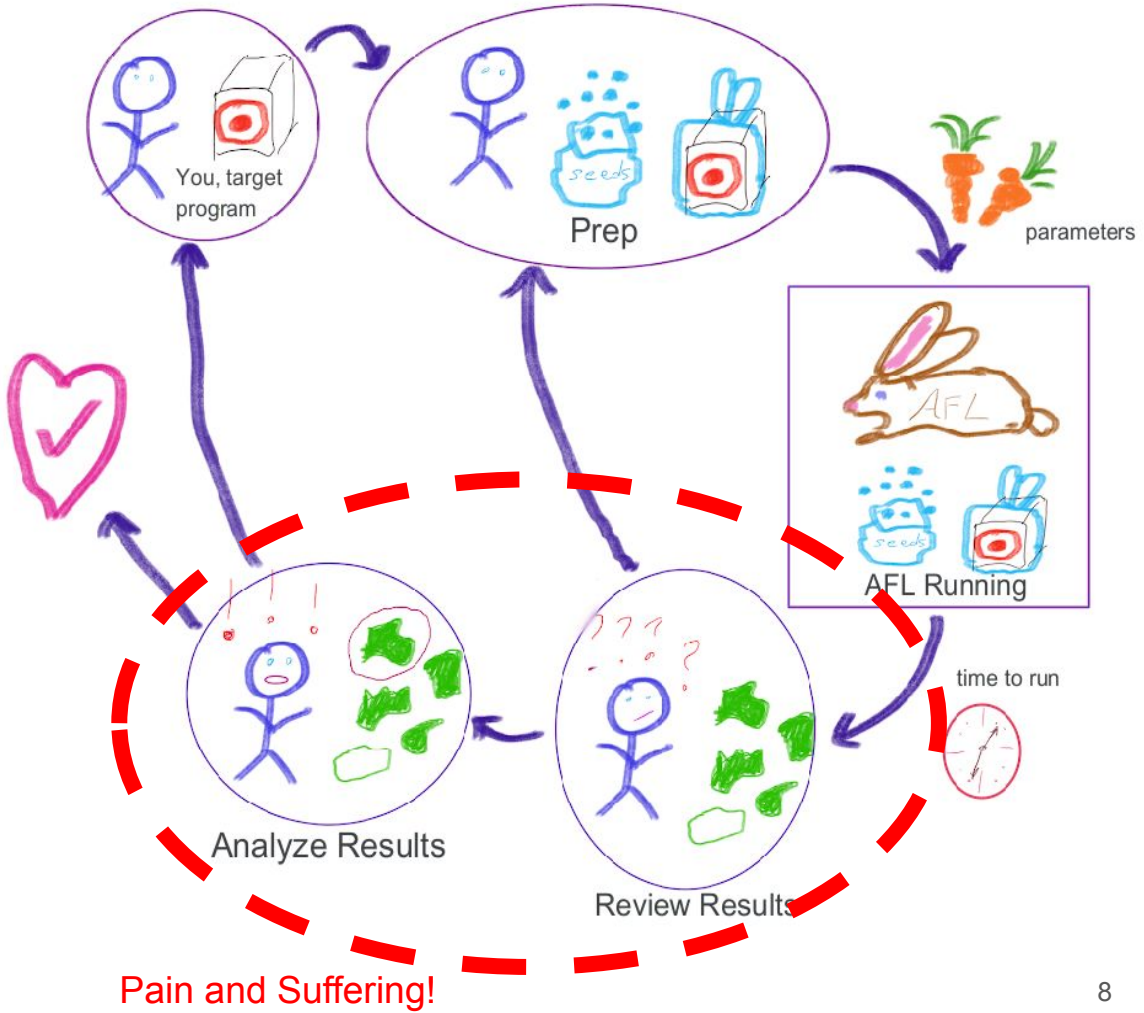
The Problem of Root Cause

"Developers generally appreciate bug reports, but they can sometimes be a bit **less enthusiastic about a flood of reports from automated fuzzing** systems." https://lwn.net/Articles/904293/

"..you have an ethical and moral responsibility to do some of the work to narrow down and **identify the cause of the failure**, not just throw them at someone to do all the work" https://lwn.net/Articles/917762/

How bad is it? (audience participation)
If a fuzzer produced 3,020 crashes, how many bugs do you think there were?

Ans: there were 15 bugs (FuzzerAid)



Not Everyone Cares about Root Cause

Offensive (Red):
Vulnerability Researcher, Penetration Tester, Bug Bounty Hunter

Goals:

- Finding at least one vulnerability or chain
- ASAP! Preferably before anyone else
- Rarity, stability good
- **Don't care about 'why'**

Defensive (Blue):
Application Security Engineer, Security Consultant

Goals:

- Find all the vulnerabilities
- Find them all before release
- **Care about 'why'**

Software Owners:
Engineering Manager, Software Engineer, QA Engineer

Goals:

- Find all vulnerabilities
- Find all bugs
- Find them before release
- **Really care about 'why'**

Academics: Phd Student, Faculty, Masters thesis

Goals:

- A previous group's goals
- Studying Software Testing
- Studying another aspect of CS
- Clear, easy metrics

PIPE (Programmable Interlocks for Policy Enforcement)

- Is a Tag based hardware security reference monitor
  - ISA extension
- A tag represents arbitrary metadata associated with some data
  - Ex “This is a pointer to object a”, “This is low security data”, “The active function is f”, “this instruction can only be used by high security data’
- All data have tags
  - Check relevant tag rules on each execution step
  - if policy is violated (maybe) failstop
- PIPE, doesn’t care about language, high or low
- Problem: hardware doesn’t entirely exist yet, and what does isn’t widely available .

THE

PROGRAMMING LANGUAGE

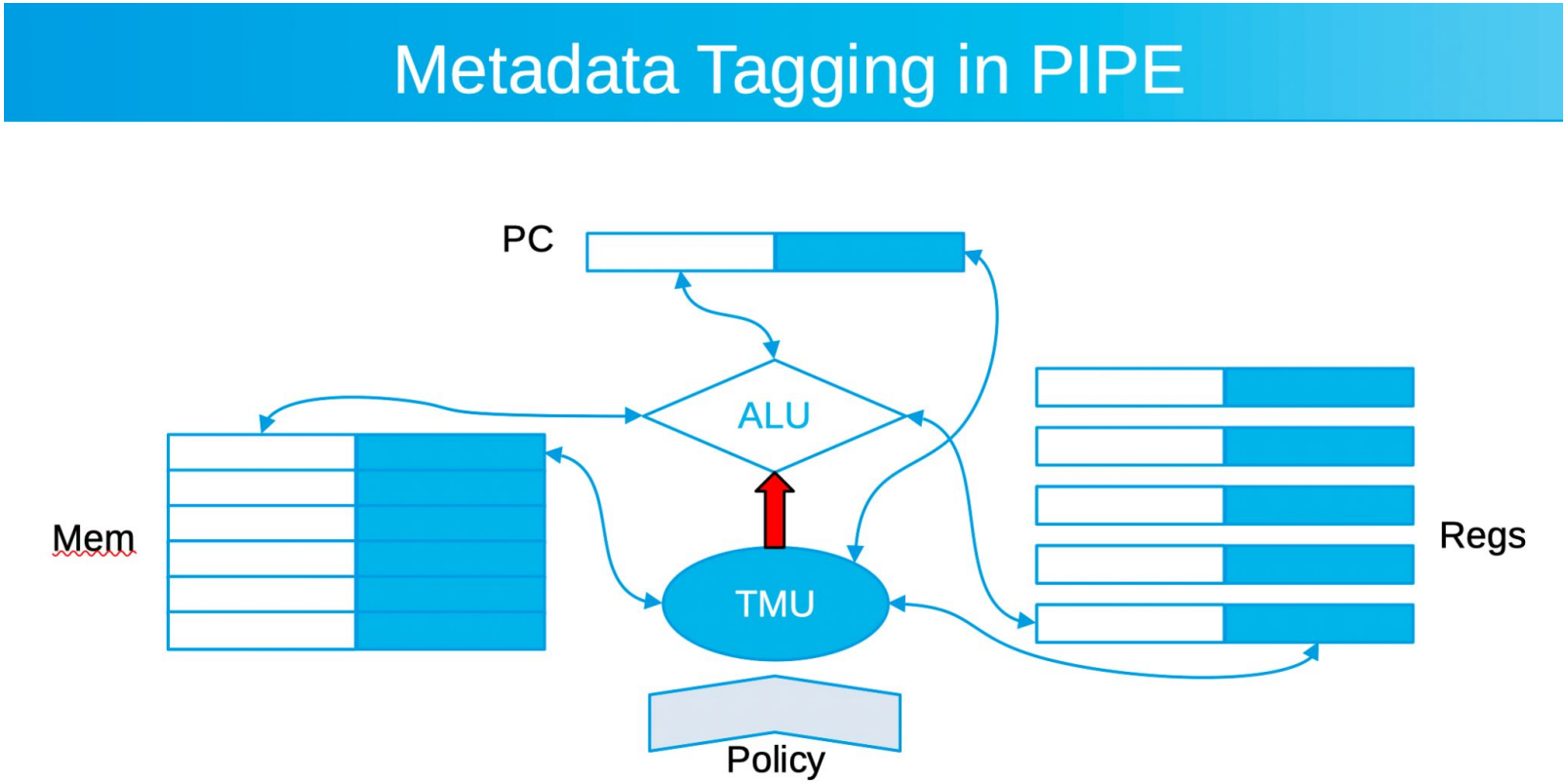
the leading cause of security analysts

From emj3

TaggedC

- Built on Compcert's Interpreter
  - Give semantics for all undefined behavior
- PIPE policies are encoded at source level
  - Developers rarely speak assembly or machine code
- PL-y magic happens here
  - Sean Anderson's dissertation work
- Formalizing Stack Safety as a Security Property
  - https://arxiv.org/abs/2105.00417 preprint
  - Sean is presenting it at a conference later this month
  - We have another paper in submission which discusses TaggedC in more detail
- Verified compilation part
  - Towards formally verified compilation of tag-based policy enforcement https://dl.acm.org/doi/10.1145/3437992.3439929

Possible PIPE Implementation



From Sean Anderson's deck

TaggedC POC Policy Violation

- Provence via Integer (PVI)
  - Supports arbitrary integer arithmetic on (int-cast) pointers
  - Does not support crossing object bounds
  - Memory model from optimization literature
- Why would you do this in C?
  - Use lower order bits as flag
  - E.g. Cheney's garbage collection algorithm

```c
int main() {
  int x[10];
  int y[10];

  *(x + 10) = 42;
  return y[0];
}
```

```
a@pyrite tagged-c % ./violation1
zsh: abort      ./violation1
```

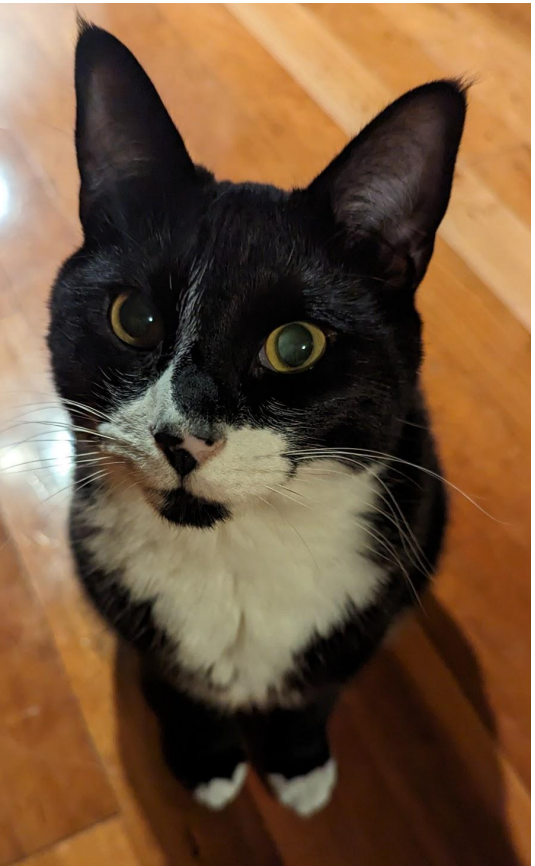
VS

```
a@pyrite tagged-c % ./ccomp -p pvi violation1.c
Failstop on policy PVI::LoadT X Failure
```

The Dissertation Fuzzer I Hope To Build: PIPE Cleaner

- Assuming I have cleverly designed policies in TaggedC
  - Specifying these formally is a nontrivial PL sort of challenge
    - especially beyond memory corruption
    - Formal speak: interested in Hyperproperties
- Fuzzer searches for PIPE Policy violations rather than segfaults
  - Fuzzer has root cause at time of fault, not something you get from a segfault
  - Encoding “interesting behavior” as policies means I can change what to fuzz for without changing the fuzzer
  - Policies can better represent things like SQL injection that don’t make sense as segfaults
- Ideally, first fuzzer to have no false positives by construction
  - Imagine instead of 3,020 crash reports, you only have to read 15 reports?
  - instead of consigning fuzzer report to oblivion or summer interns, bugs might get fixed
  - Fuzzing is always unsound (false negatives. May miss bugs)

Thank you for Listening <3



Questions & Unsolicited Advice

1. Lock/close your fsck-ing laptops!
   a. You wash your hands after you use the bathroom, right?
   b. It's a possible FERPA violation if you teach in the US
2. Use MFA, especially on school email
3. Your (risky) behavior matters more than OS.
   a. Stay Patched.
4. **Vulnerabilities are *expensive*, your mistakes (or laziness) are *free!***
5. Have reason for students to learn PL beyond "purity". I had to wait 18 years for a reason. They won't.