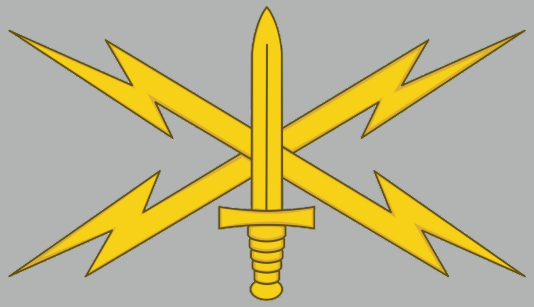
MEASURING FUZZERS

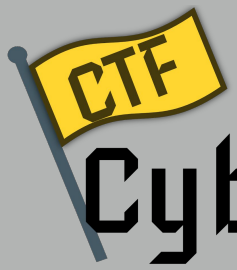
A SUMMER WITH INDUSTRY

WHO



Summer With Industry @

ARMY CYBER INSTITUTE AT WEST POINT



All-Army CyberStakes

TRAIL OF BITS

DISCLAIMER

The views expressed in this talk are those of the speaker and do not reflect the official policy or position of the Army Cyber Institute, the Department of the Army, the Department of Defense or the United States Government...

STARTING POINT

```
                    american fuzzy lop 0.47b (readpng)

 process timing                                                overall results
        run time : 0 days, 0 hrs, 4 min, 43 sec                 cycles done : 0
   last new path : 0 days, 0 hrs, 0 min, 26 sec                 total paths : 195
 last uniq crash : none seen yet                               uniq crashes : 0
  last uniq hang : 0 days, 0 hrs, 1 min, 51 sec                  uniq hangs : 1
 cycle progress                                  map coverage
  now processing : 38 (19.49%)                     map density : 1217 (7.43%)
 paths timed out : 0 (0.00%)                    count coverage : 2.55 bits/tuple
 stage progress                                  findings in depth
  now trying : interest 32/8                    favored paths : 128 (65.64%)
 stage execs : 0/9990 (0.00%)                    new edges on : 85 (43.59%)
 total execs : 654k                             total crashes : 0 (0 unique)
  exec speed : 2306/sec                           total hangs : 1 (1 unique)
 fuzzing strategy yields                                       path geometry
   bit flips : 88/14.4k, 6/14.4k, 6/14.4k                        levels : 3
  byte flips : 0/1804, 0/1786, 1/1750                           pending : 178
 arithmetics : 31/126k, 3/45.6k, 1/17.8k                       pend fav : 114
  known ints : 1/15.8k, 4/65.8k, 6/78.2k                       imported : 0
       havoc : 34/254k, 0/0                                    variable : 0
        trim : 2876 B/931 (61.45% gain)                          latent : 0
```

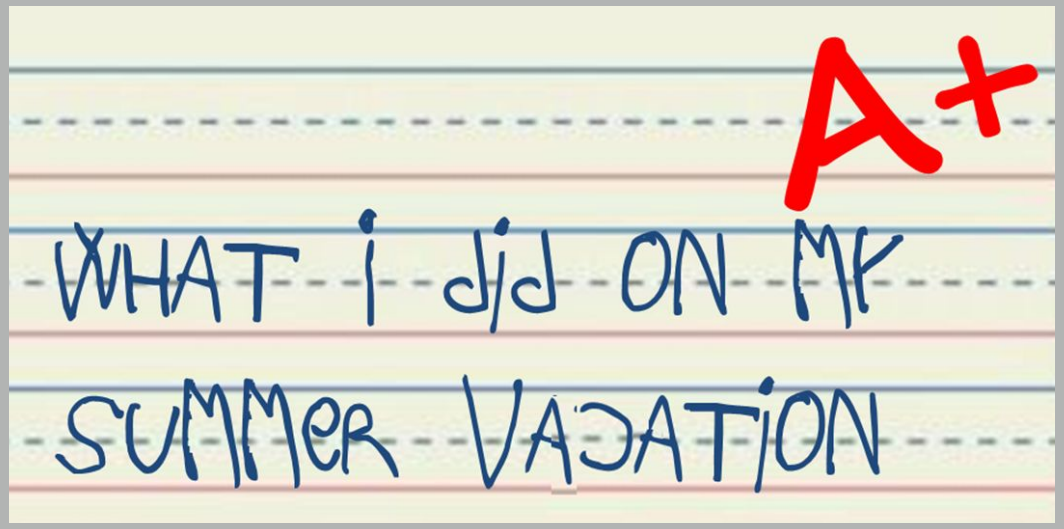
WHAT'S A LOP?

AFL IS RUNNING. I THINK?

OWN THE BUG-O-RAMA TROPHY CASE

A QUESTION... A JOURNEY...

Where do fuzzers spend their time?



THE STORY

Fuzzers, are highly effective and heavily researched

But, there is still a lot of art and intuition

So, we measure

"assessed the experimental evaluations carried out by 32 fuzzing papers. We found problems in every evaluation we considered."

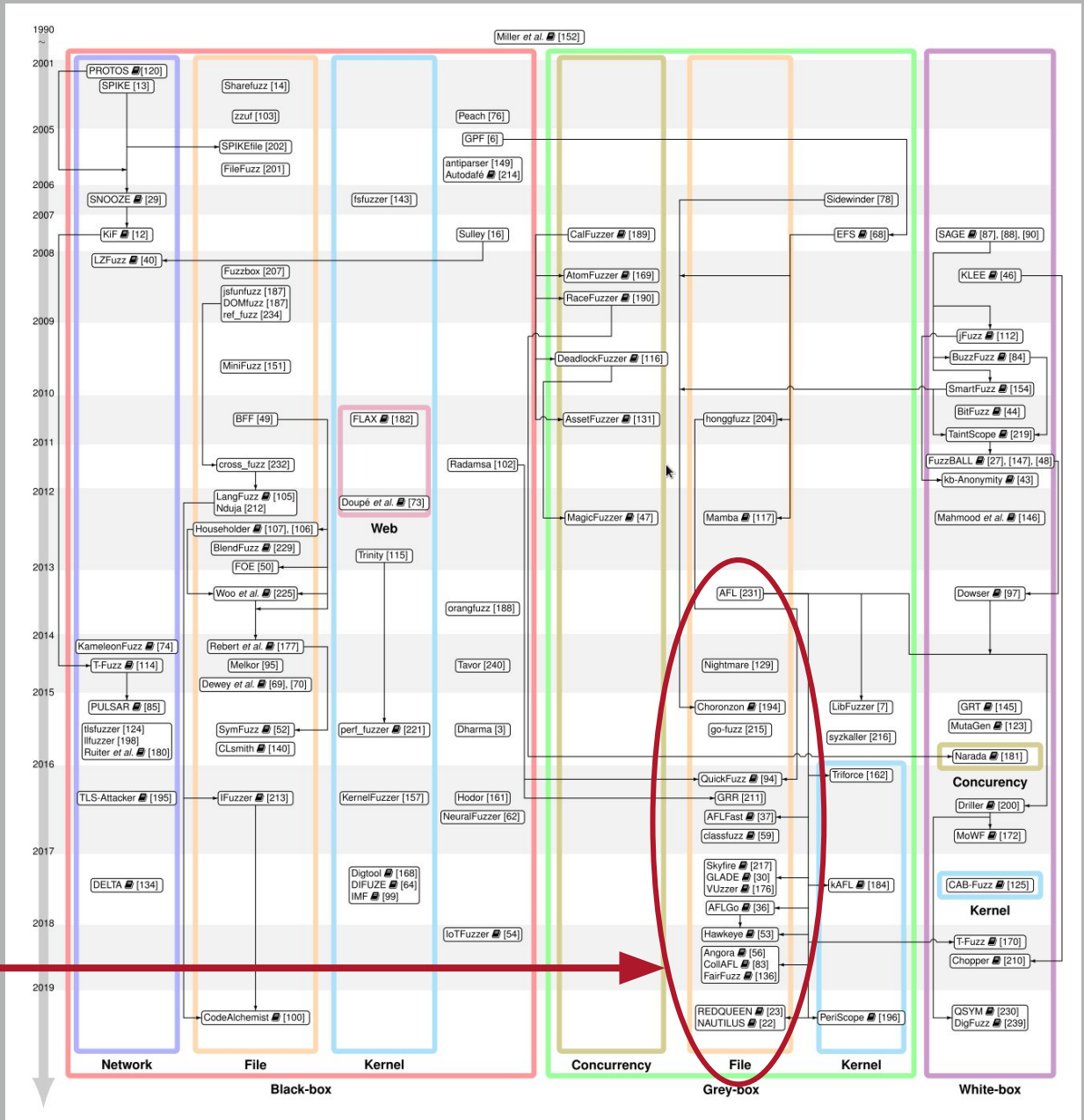
- *Evaluating Fuzz Testing* , Klees, et al.

THE LANDSCAPE

The Art, Science, and Engineering of Fuzzing: A Survey

Manes, et al.

coverage guided mutational fuzzers



THE BACKSTORY

fuzzer

tests a program on many inputs

THE BACKSTORY

***mutational* fuzzer**

produce inputs by modify existing seed (corpus)

THE BACKSTORY

coverage guided
mutational fuzzer

instrument the program to inform selection/mutation

IN THE WILD: AFL

```
                american fuzzy lop 0.47b (readpng)

┌─ process timing ─────────────────────────────────────┬─ overall results ─────┐
│        run time : 0 days, 0 hrs, 4 min, 43 sec       │  cycles done : 0      │
│   last new path : 0 days, 0 hrs, 0 min, 26 sec       │  total paths : 195    │
│ last uniq crash : none seen yet                      │ uniq crashes : 0      │
│  last uniq hang : 0 days, 0 hrs, 1 min, 51 sec       │   uniq hangs : 1      │
├─ cycle progress ────────────────────┬─ map coverage ─┴───────────────────────┤
│  now processing : 38 (19.49%)       │    map density : 1217 (7.43%)          │
│ paths timed out : 0 (0.00%)         │ count coverage : 2.55 bits/tuple       │
├─ stage progress ────────────────────┼─ findings in depth ────────────────────┤
│  now trying : interest 32/8         │ favored paths : 128 (65.64%)           │
│ stage execs : 0/9990 (0.00%)        │  new edges on : 85 (43.59%)            │
│ total execs : 654k                  │ total crashes : 0 (0 unique)           │
│  exec speed : 2306/sec              │   total hangs : 1 (1 unique)           │
├─ fuzzing strategy yields ───────────┴───────────────┬─ path geometry ────────┤
│   bit flips : 88/14.4k, 6/14.4k, 6/14.4k            │    levels : 3          │
│  byte flips : 0/1804, 0/1786, 1/1750                │   pending : 178        │
│ arithmetics : 31/126k, 3/45.6k, 1/17.8k             │  pend fav : 114        │
│  known ints : 1/15.8k, 4/65.8k, 6/78.2k             │  imported : 0          │
│       havoc : 34/254k, 0/0                          │  variable : 0          │
│        trim : 2876 B/931 (61.45% gain)              │    latent : 0          │
└─────────────────────────────────────────────────────┴────────────────────────┘
```

```
$ ./afl-fuzz -i testcase_dir -o findings_dir ./program
```

The "sales pitch"

- It is fast.
- It’s rock solid.
- No tinkering required.

http://lcamtuf.coredump.cx/afl/

IN THE WILD: LIBFUZZER

```
// fuzz_target.cc
extern "C" int LLVMFuzzerTestOneInput(const uint8_t *Data, size_t Size) {

    DoSomethingInterestingWithMyAPI(Data, Size);
    return 0;

}
```

“LibFuzzer is an in-process, coverage-guided, evolutionary fuzzing engine.”

https://llvm.org/docs/LibFuzzer.html

THE IVORY TOWER

"Designing New Operating Primitives to Improve Fuzzing Performance"	*parallel fuzzing + better os (fork, fs, shared log)*
"Full-speed Fuzzing: Reducing Fuzzing Overhead through Coverage-guided Tracing"	*UnTracer = afl + lighter instrumentation*
"Coverage-based Greybox Fuzzing as Markov Chain"	*AFLFast = afl + better selection*
"VUzzer: Application-aware Evolutionary Fuzzing"	*+ control and data-flow*

TARGET MATTERS

"fuzzing performance may vary with the target program, so it is important to evaluate on a diverse, representative benchmark suite"

- *Evaluating Fuzz Testing* , Klees, et al.

Synthetic Bugs in Real Programs

Unknown Bugs in Real Programs

Real Bugs in Real Programs

SYNTHETIC BUGS

LAVA:
Large-scale Automated Vulnerability Addition

https://github.com/panda-re/lava

comparison!

Of Bugs and Baselines

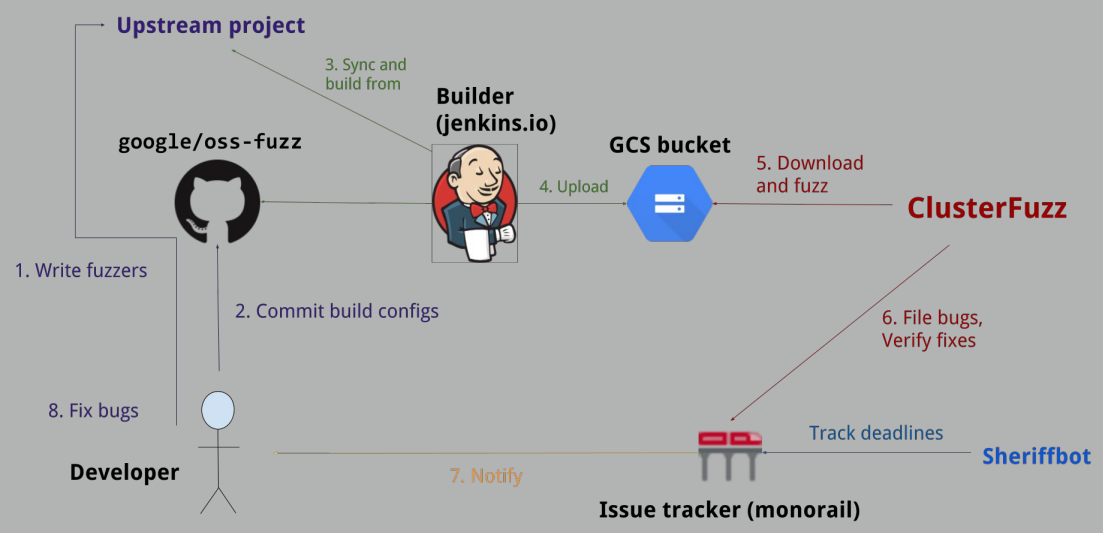
"recently published results on the LAVA-M synthetic bug dataset are exciting. However, [...] we need to be cautious in our evaluations and not rely too much on getting a high score on a single benchmark"

- Brendan Dolan-Gavitt @moyix

http://moyix.blogspot.com/2018/03/of-bugs-and-baselines.html

UNKNOWN BUGS

OSS-Fuzz: Continuous Fuzzing for Open Source Software



+200 fuzz ready targets (libfuzzer, afl)

standardized configuration and build process

Scale. speed + coverage!

https://github.com/google/oss-fuzz

REAL BUGS

Fuzzer Test Suite

“a set of fuzzing benchmarks derived from real-life libraries that have interesting bugs, hard-to-find code paths, or other challenges for bug finding”

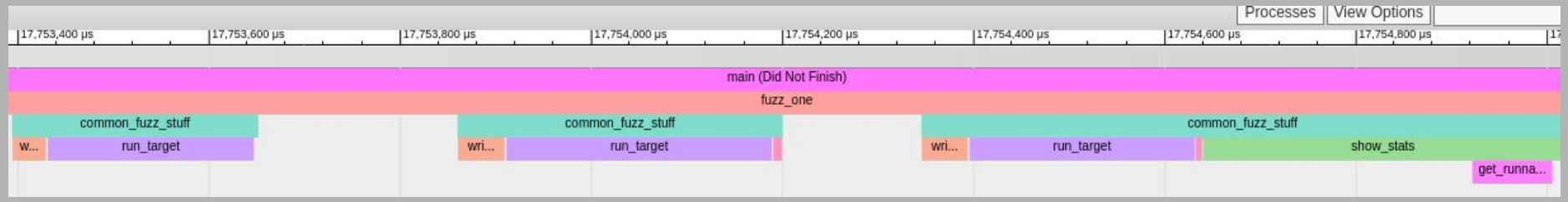
~20 validated programs
known vulnerabilities, reproducers, and seeds

ground truth!

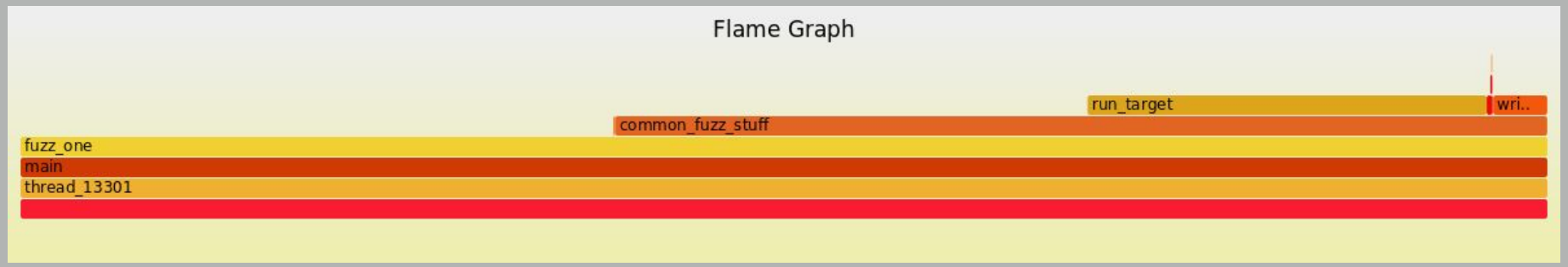
https://github.com/google/fuzzer-test-suite

HOW TO TIME?

LLVM XRay



"XRay is a function call tracing system which combines compiler-inserted instrumentation points and a runtime library that can dynamically enable and disable the instrumentation"



https://llvm.org/docs/XRay.html

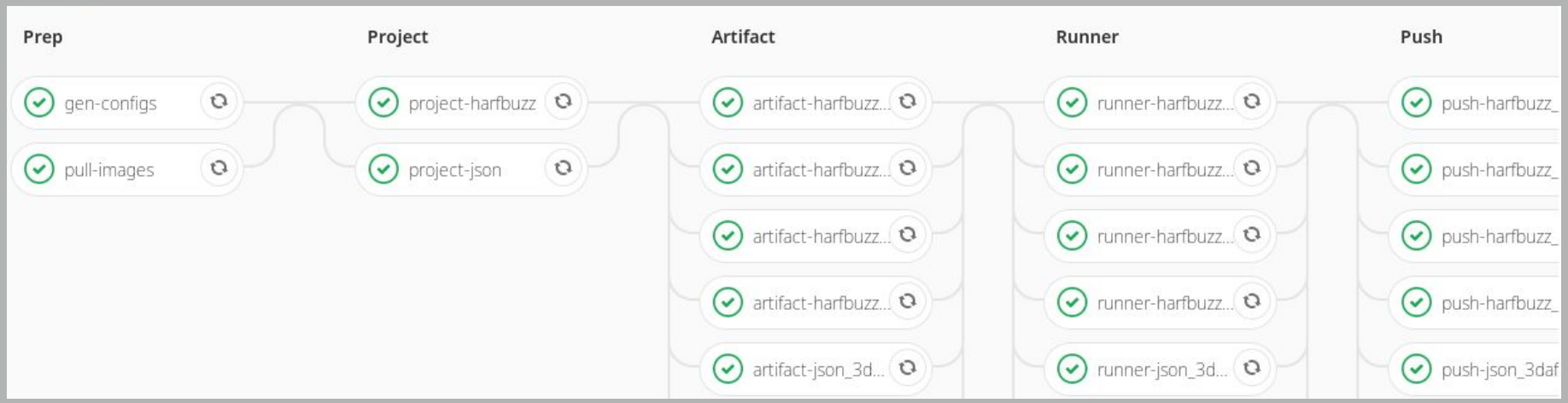
LET'S PUT IT ALL TOGETHER

```
# experiment.yml

options:
    engine: [afl, libfuzzer]
    asan: [yes, no]

targets:
    - harfbuzz
    - json
```

```
$ python helper.py ./experiment.yml build_experiment
```



REPORT CARD

Where do fuzzers spend their time?

- ✅ **Fuzzers** (afl, libfuzzer)
- ✅ **Targets** (lava, oss-fuzz, fuzzer-test-suite)
- ✅ **Timing** (llvm XRay)
- ✅ **Automation** (“fuzz ready” containers)
- ☐ **Go!**

THANK YOU!

Fuzzers are great.

Still a lot of art.

Let’s measure.

https://gitlab.com/royragsdale/fuzzing-measurement/