

KLEE Workshop 2018

Feeding the Fuzzers

with KLEE

Marek Zmysłowski

MOBILE SECURITY TEAM
SAMSUNG
R&D INSTITUTE POLAND

**This presentation was created with help and
commitment of the
Samsung R&D Poland Mobile Security team.**

KLEE and AFL

Multimedia (video)

What is KLEE

If you don't know what KLEE is,
you are probably on the wrong conference 😊

What is AFL : short version

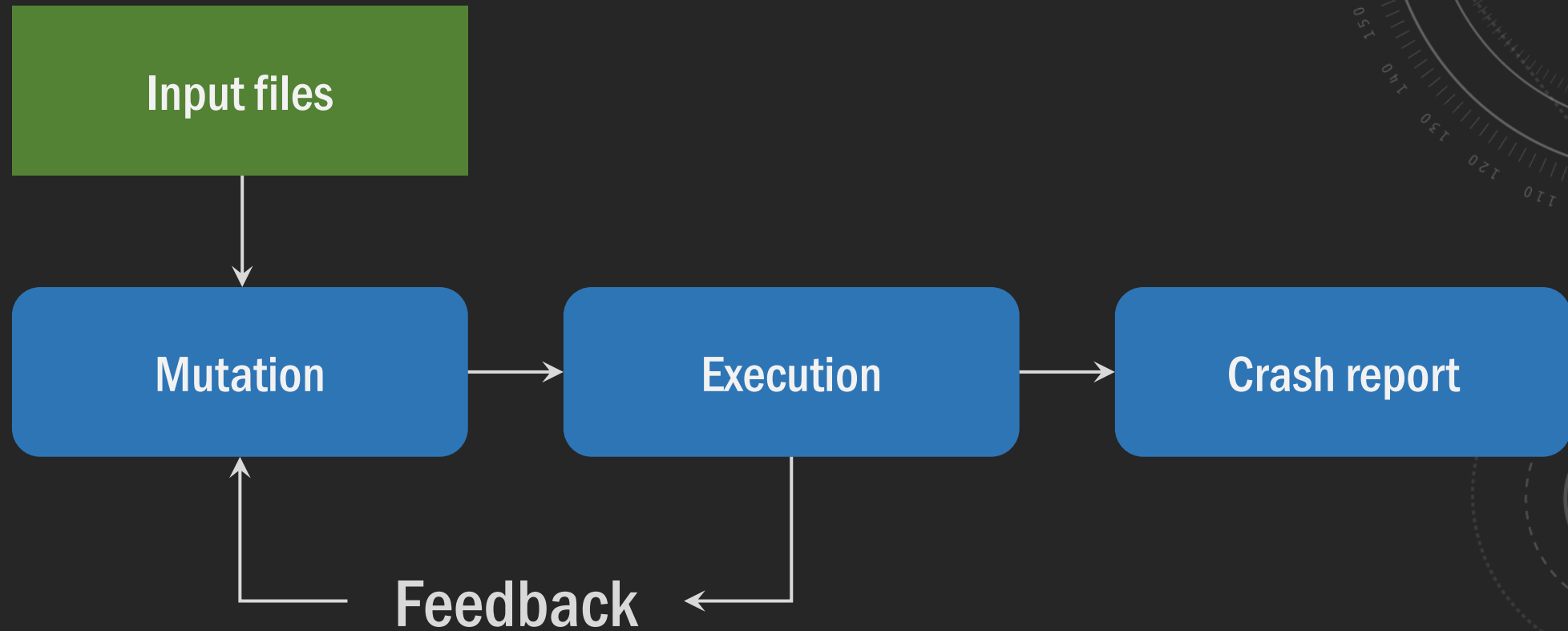
Security-oriented fuzzer that employs a novel type of compile-time instrumentation and genetic algorithms to automatically discover clean, interesting test cases that trigger new internal states in the targeted binary.*

* <http://lcamtuf.coredump.cx/afl/>

What is AFL : long version

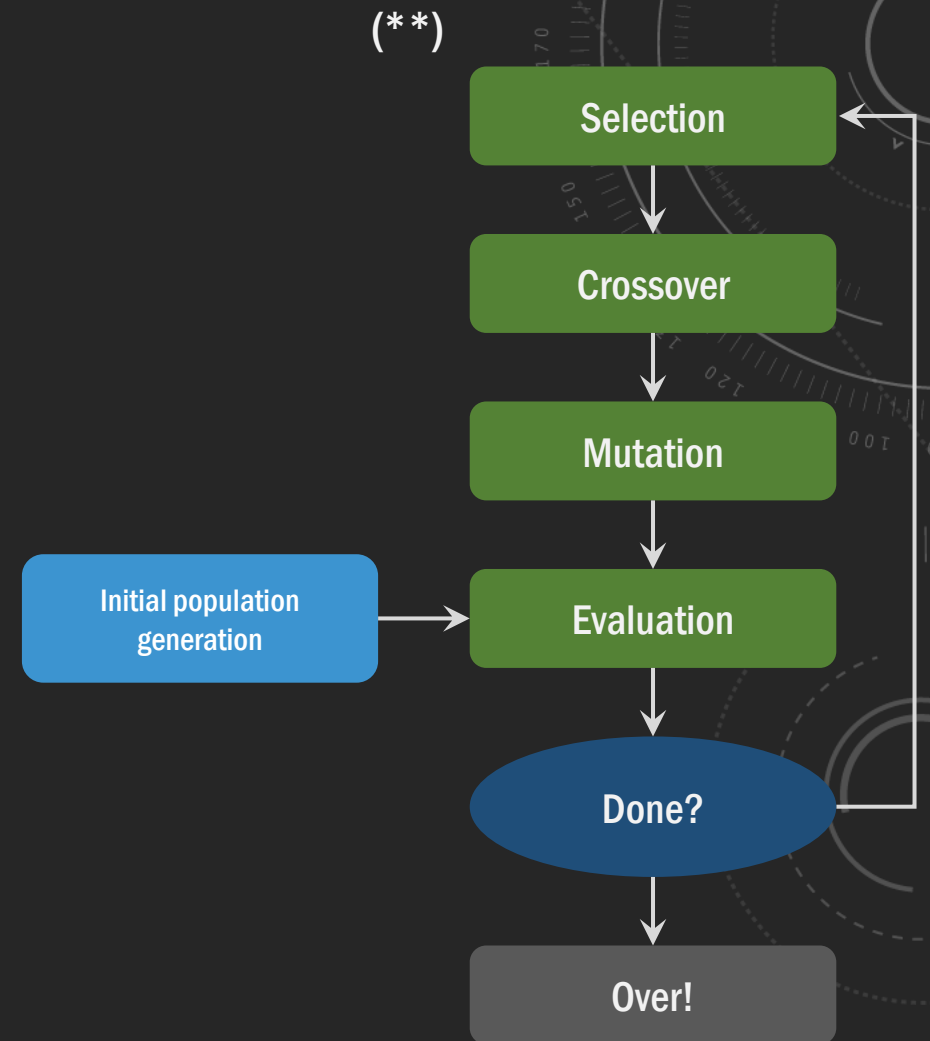
- **FUZZER** - automated software testing tool that involves providing invalid, unexpected, or random data as inputs to a computer program
- **COMPILED-TIME INSTRUMENTED** – it uses calls to user-defined functions on edge level. It is used to improve the functional coverage for the fuzzed code
- **USES GENETIC ALGORITHMS** – the genetic algorithm are used to create more robust input for the testing

AFL Scheme



Genetic Algorithm

The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Often, the **initial population** is generated randomly, **allowing the entire range of possible solutions** (the search space). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.*



* https://en.wikipedia.org/wiki/Genetic_algorithm

** <https://genetic.io/en/introduction-genetic-algorithms>

Initial Input Set - Corpus

„Coverage-guided fuzzers like libFuzzer rely on a corpus of sample inputs for the code under test. This corpus should ideally be seeded with a varied collection of valid and invalid inputs for the code under test;”*

* <https://llvm.org/docs/LibFuzzer.html>

Corpus – How to get it

- Start with empty corpus
- Generated with other tools
- Downloaded from the Internet

Not everything can be done by AFL

```
volatile int* noopt_p = NULL;
int main(int argc, char** argv) {

    unsigned char x[10] = {};
    unsigned int *i;

    if (argc < 2) return 1;

    int fd = open(argv[1], O_RDONLY);
    if (fd < 0) return 1;

    int r = read(fd, x, 10);
    if (r < 4) return 2;

    i = (unsigned int*)x;
    if (*i == 0xdeadbeef) {
        *noopt_p = 0x1234;
        printf("Crash\n");
    }
}
```

Not everything can be done by AFL

american fuzzy lop 2.52b (a.out)

```
process timing
  run time : 1 days, 0 hrs, 12 min, 24 sec
  last new path : 1 days, 0 hrs, 12 min, 23 sec
  last uniq crash : none seen yet
  last uniq hang : none seen yet
cycle progress
  now processing : 1 (50.00%)
  paths timed out : 0 (0.00%)
stage progress
  now trying : splice 5
  stage execs : 21/22 (96.88%)
  total execs : 749M
  exec speed : 8408/sec
fuzzing strategy yields
  bit flips : 0/56, 0/54, 0/50
  byte flips : 0/7, 0/5, 0/1
  arithmetics : 0/387, 0/3, 0/0
  known ints : 0/40, 0/139, 0/44
  dictionary : 0/0, 0/0, 0/0
  havoc : 1/260M, 0/488M
  trim : n/a, 0.00%
overall results
  cycles done : 509k
  total paths : 2
  uniq crashes : 0
  uniq hangs : 0
map coverage
  map density : 0.00% / 0.00%
  count coverage : 1.00 bits/tuple
findings in depth
  favored paths : 2 (100.00%)
  new edges on : 2 (100.00%)
  total crashes : 0 (0 unique)
  total tmouts : 0 (0 unique)
path geometry
  levels : 2
  pending : 0
  pend fav : 0
  own finds : 1
  imported : n/a
  stability : 100.00%
[cpu004: 69%]
```

What if we have unknown/rare multimedia files

- Start with empty
- Download from Internet
- Use an encoder if provided
- Reverse the decoder
- Use KLEE

Why don't use KLEE

- Path explosion
- Program-dependent efficacy
- Environment interactions

Why don't use KLEE

- Path explosion
- Program-dependent efficacy
- Environment interactions

Symbolically executing all feasible program paths does not scale to large programs. The number of feasible paths in a program grows exponentially with an increase in program size and can even be infinite in the case of programs with unbounded loop iterations.

Why don't use KLEE

- Path explosion
- **Program-dependent efficacy**
- Environment interactions

Symbolic execution is used to reason about a program path-by-path which is an advantage over reasoning about a program input-by-input as other testing paradigms use (e.g. Dynamic program analysis). However, if few inputs take the same path through the program, there is little savings over testing each of the inputs separately.

Why don't use KLEE

- Path explosion
- Program-dependent efficacy
- Environment interactions

Programs interact with their environment by performing system calls, receiving signals, etc. Consistency problems may arise when execution reaches components that are not under control of the symbolic execution tool (e.g., kernel or libraries).

Why don't use KLEE

- Resource consumption
- Issue with building the whole environment – there are assembly blobs optimized for particular architecture (ARM, AARCH64, x86)
- Multithread
- External libraries
- No scalable problem
- Distributed environment with many nodes

Multimedia

- Current multimedia codecs can be very complicated (including multiple different options)
- Highly optimized for the architecture (many assembler blobs)
- Multi layers – NAL, bitstreams, frames. Can have different relation inside layers

Program-dependent efficacy

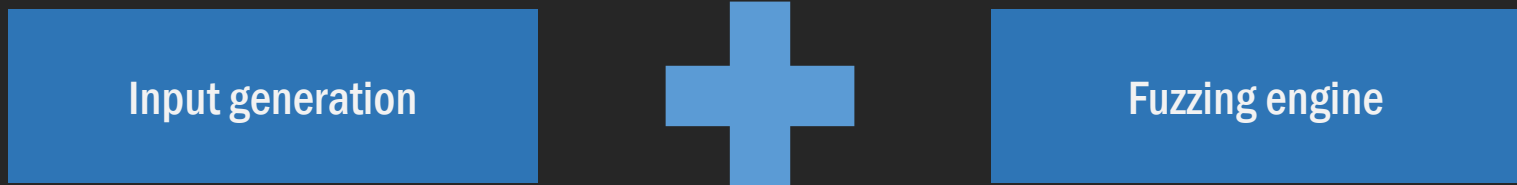
Environment interaction

Path explosion

KLEE - Advantages

- No need to build whole library
- No need to run through whole application – it can be run in limited amount of time with different search algorithm to generate initial set of files.
- Open source 😊

Solution



Solution



Why is it better than Concolic Execution?

- It is not.
- It is just different approach developed for better use of the resource and fuzzing capabilities.
- You can treat this a 0 level concolic execution.

Examples

Multimedia (video)

Free Lossless Audio Codec (flac)

An audio format similar to MP3, but lossless, meaning that audio is compressed in FLAC without any loss in quality.

It is open source and available at:

- <https://xiph.org/flac/>

The c written encoder was used as an example.

flac – Initial Set

Empty – one file with single character was created.

Search: nurs:md2u, nurs:covnew

File size: 16

Max forks: 4096

Result:

KLEE: done: total instructions = 3029103

KLEE: done: completed paths = 4097

KLEE: done: generated tests = 97

flac - 1 vs 1

```
american fuzzy lop 2.52b (example_c_decode_file)

process timing
  run time : 17 days, 22 hrs, 28 min, 12 sec
  last new path : 0 days, 6 hrs, 53 min, 6 sec
  last uniq crash : none seen yet
  last uniq hang : none seen yet
cycle progress
  now processing : 919* (74.29%)
  paths timed out : 0 (0.00%)
stage progress
  now trying : arith 16/8
  stage execs : 405k/1.09M (37.07%)
  total execs : 492M
  exec speed : 272.3/sec
fuzzing strategy yields
  bit flips : 43/27.6M, 21/27.6M, 22/27.5M
  byte flips : 2/3.44M, 2/2.07M, 0/2.09M
  arithmetics : 70/115M, 2/25.2M, 0/4.77M
  known ints : 7/9.66M, 8/53.7M, 14/86.8M
  dictionary : 0/0, 0/0, 21/98.0M
  havoc : 1024/8.14M, 0/0
  trim : 8.28%/328k, 40.18%

overall results
  cycles done : 14
  total paths : 1237
  uniq crashes : 0
  uniq hangs : 0

map coverage
  map d
  count co
  finding
  favored
  new edg
  total cr
  total t
```

```
american fuzzy lop 2.52b (example_c_decode_file)

process timing
  run time : 12 days, 20 hrs, 0 min, 13 sec
  last new path : 0 days, 8 hrs, 51 min, 32 sec
  last uniq crash : none seen yet
  last uniq hang : none seen yet
cycle progress
  now processing : 1522* (98.83%)
  paths timed out : 0 (0.00%)
stage progress
  now trying : bitflip 2/1
  stage execs : 198k/472k (41.89%)
  total execs : 696M
  exec speed : 11.95/sec (zzzz...)
fuzzing strategy yields
  bit flips : 76/67.2M, 26/66.8M, 28/66.8M
  byte flips : 4/8.35M, 3/2.14M, 3/2.23M
  arithmetics : 43/116M, 6/66.9M, 6/36.3M
  known ints : 5/8.14M, 8/44.5M, 16/82.5M
  dictionary : 0/0, 0/0, 20/112M
  havoc : 1199/14.6M, 0/0
  trim : 7.01%/828k, 74.93%

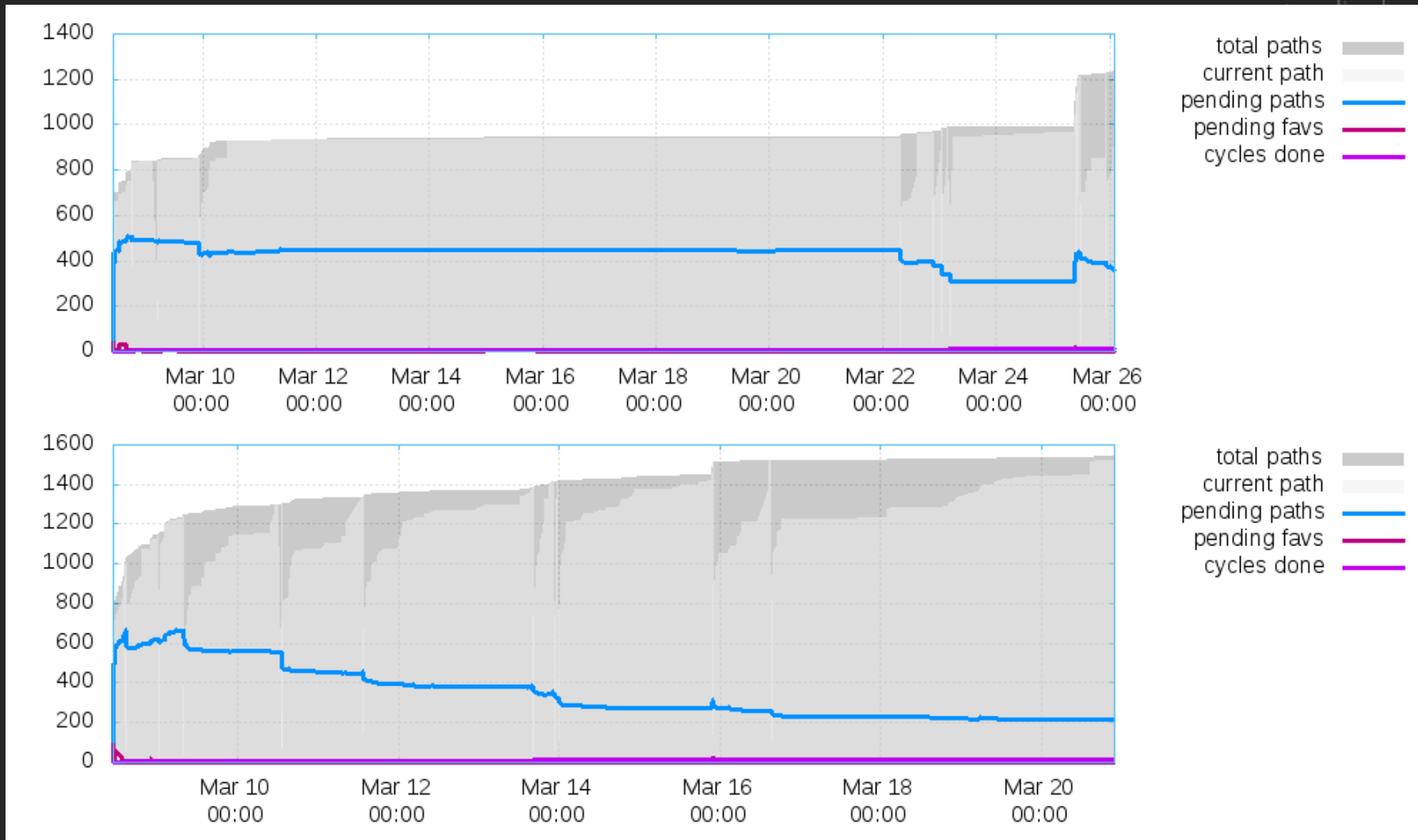
overall results
  cycles done : 15
  total paths : 1540
  uniq crashes : 0
  uniq hangs : 0

map coverage
  map density : 0.42% / 1.06%
  count coverage : 5.14 bits/tuple
findings in depth
  favored paths : 166 (10.78%)
  new edges on : 253 (16.43%)
  total crashes : 0 (0 unique)
  total tmounts : 0 (0 unique)

path geometry
  levels : 21
  pending : 214
  pend fav : 0
  own finds : 1443
  imported : n/a
  stability : 100.00%
```

[cpu002: 50%]

flac - 1 vs 1



flac – Master-Slave Android

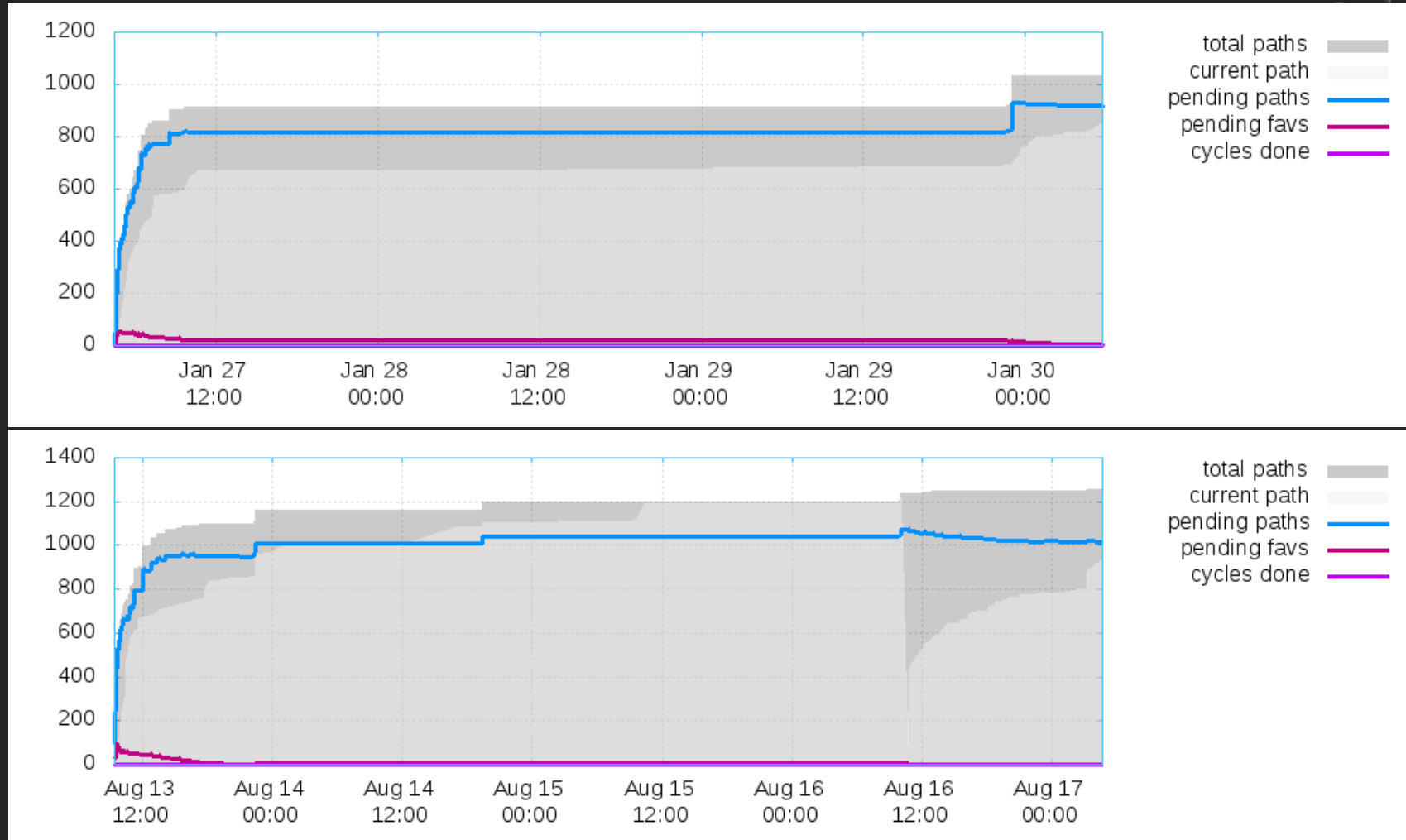
Masters

american fuzzy lop 2.52b (master)

process timing run time : 6 days, 2 hrs, 8 min, 56 sec last new path : 3 days, 2 hrs, 5 min, 55 sec last uniq crash : none seen yet last uniq hang : none seen yet	overall results cycles done : 0 total paths : 1035 uniq crashes : 0 uniq hangs : 0	
cycle progress now processing : 877* (84.73%) paths timed out : 0 (0.00%)	map coverage map density : 0.40% / 0.73%	
stage progress now trying : bitflip 16/8 stage execs : 26.0k/97.3k (26.72%) total execs : 21.7M exec speed : 13.02/sec (zzzz...)	american fuzzy lop 2.52b (master-klee)	
fuzzing strategy yields bit flips : 0/1.99M, 0/1.99M, 0/1.99M byte flips : 0/248k, 0/78.7k, 0/79.3k arithmetics : 1/4.39M, 0/579k, 0/243k known ints : 0/446k, 1/2.10M, 0/3.39M dictionary : 0/0, 0/0, 0/3.88M havoc : 78/211k, 0/0 trim : 1.63%/21.4k, 47.93%	process timing run time : 4 days, 0 hrs, 31 min, 2 sec last new path : 0 days, 6 hrs, 37 min, 57 sec last uniq crash : none seen yet last uniq hang : none seen yet	
	overall results cycles done : 1 total paths : 1254 uniq crashes : 0 uniq hangs : 0	
	map coverage map density : 0.42% / 0.88% count coverage : 5.24 bits/tuple	
	findings in depth favored paths : 134 (10.69%) new edges on : 219 (17.46%) total crashes : 0 (0 unique) total tmouts : 0 (0 unique)	
	stage progress now trying : arith 32/8 stage execs : 31.8k/417k (7.61%) total execs : 23.5M exec speed : 62.42/sec (slow!)	path geometry levels : 3 pending : 1018 pend fav : 1 own finds : 174 imported : 984 stability : 100.00%
	fuzzing strategy yields bit flips : 15/1.75M, 6/1.75M, 4/1.75M byte flips : 0/218k, 1/85.0k, 2/87.9k arithmetics : 1/4.66M, 0/2.71M, 4/932k known ints : 0/375k, 1/1.67M, 4/3.18M dictionary : 0/0, 0/0, 1/3.97M havoc : 135/291k, 0/0 trim : 8.72%/48.4k, 61.63%	

[cpu001: 48%]

flac – Master-Slave Android



flac – Master-Slave Android

Slaves

american fuzzy lop 2.52b (slave)

process timing

run time : 6 days, 2 hrs, 8 min, 1 sec
last new path : 0 days, 0 hrs, 12 min, 34 sec
last uniq crash : none seen yet
last uniq hang : none seen yet

cycle progress

now processing : 870 (80.71%)
paths timed out : 0 (0.00%)

stage progress

now trying : splice 1
stage execs : 28/64 (43.75%)
total execs : 43.9M
exec speed : 107.3/sec

fuzzing strategy yields

bit flips : n/a, n/a, n/a
byte flips : n/a, n/a, n/a
arithmetics : n/a, n/a, n/a
known ints : n/a, n/a, n/a
dictionary : n/a, n/a, n/a
havoc : 553/15.5M, 496/27.9M
trim : 4.71%/547k, n/a

overall results

cycles done : 194
total paths : 1078

american fuzzy lop 2.52b (slave-klee)

process timing

run time : 4 days, 0 hrs, 31 min, 43 sec
last new path : 0 days, 0 hrs, 13 min, 25 sec
last uniq crash : none seen yet
last uniq hang : none seen yet

cycle progress

now processing : 1012* (81.55%)
paths timed out : 0 (0.00%)

stage progress

now trying : havoc
stage execs : 940/1152 (81.60%)
total execs : 28.4M
exec speed : **71.77/sec (slow!)**

fuzzing strategy yields

bit flips : n/a, n/a, n/a
byte flips : n/a, n/a, n/a
arithmetics : n/a, n/a, n/a
known ints : n/a, n/a, n/a
dictionary : n/a, n/a, n/a
havoc : 475/9.92M, 630/17.8M
trim : 4.66%/676k, n/a

overall results

cycles done : 97
total paths : 1241
uniq crashes : 0
uniq hangs : 0

map coverage

map density : 0.43% / 0.88%
count coverage : 5.24 bits/tuple

findings in depth

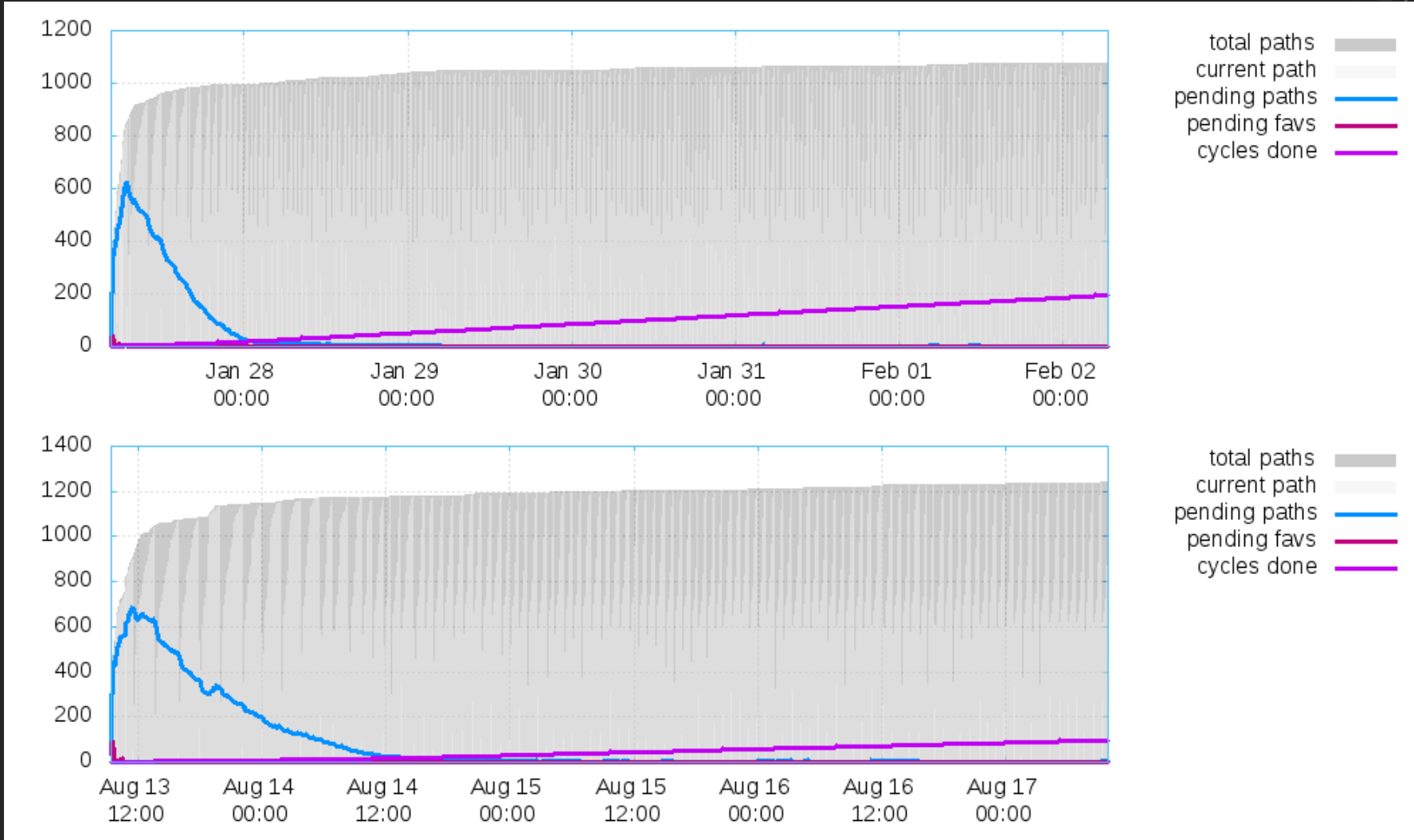
favorable paths : 136 (10.96%)
new edges on : 217 (17.49%)
total crashes : 0 (0 unique)
total tmouts : 0 (0 unique)

path geometry

levels : 19
pending : 5
pend fav : 0
own finds : 1105
imported : 40
stability : 100.00%

[cpu000: 54%]

flac – Master-Slave Android



flac – Fuzzing Farm

Empty – one file with single character was created.

1

Search: nurs:md2u, nurs:covnew

File size: 32

Max time: 3600

Max depth: 16

KLEE: clang 3.6, thread support

Result:

KLEE: done: total instructions = 1103352

KLEE: done: completed paths = 17948

KLEE: done: generated tests = 66

3

Search: nurs:md2u, nurs:covnew

File size: 16

Max time: 3600

Max depth: 16

Result:

KLEE: done: total instructions = 1035881

KLEE: done: completed paths = 15152

KLEE: done: generated tests = 64

2

Mixed – both generated sets.

4

flac – Fuzzing Farm

<u>Fuzzers alive</u>	40			
<u>Finished cycles</u>	40 / 40			
	Σ Total	□ Min	□ Median	□ Max
<u>Run time</u>		60:36:45		60:39:14
<u>Execs</u>	283,222,312	3,141,010	7,405,974	8,382,719
<u>Paths</u>		581	1,084	1,130
<u>Imported</u>		413	700	825
<u>Cumulative speed</u>	1,331.6	4.9	37.8	46.8
<u>Pending faves</u>	0	0	0	0
<u>Pending paths</u>	3,167	0	1	555
<u>Hangs occurred</u>		0	0	0
<u>Crashes found</u>		0	0	0

1

flac – Fuzzing Farm

<u>Fuzzers alive</u>	40			
<u>Finished cycles</u>	40 / 40			
	Σ Total	□ Min	□ Median	□ Max
<u>Run time</u>		18:29:09		18:32:22
<u>Execs</u>	78,096,758	1,228,177	1,862,380	2,928,963
<u>Paths</u>		557	1,154	1,261
<u>Imported</u>		363	681	868
<u>Cumulative speed</u>	1,212.8	1.4	36	50.4
<u>Pending faves</u>	34	0	0	20
<u>Pending paths</u>	8,133	16	124	764
<u>Hangs occurred</u>		0	0	0
<u>Crashes found</u>		0	0	0

2

flac – Fuzzing Farm

<u>Fuzzers alive</u>	40			
<u>Finished cycles</u>	40 / 40			
	Σ Total	□ Min	□ Median	□ Max
<u>Run time</u>		18:30:09		18:33:21
<u>Execs</u>	83,819,466	1,629,392	2,043,049	2,891,598
<u>Paths</u>		816	1,181	1,226
<u>Imported</u>		393	704	828
<u>Cumulative speed</u>	1,394.3	2.4	40.6	51
<u>Pending faves</u>	19	0	0	8
<u>Pending paths</u>	5,817	4	73	650
<u>Hangs occurred</u>		0	0	0
<u>Crashes found</u>		0	0	0

3

flac – Fuzzing Farm

<u>Fuzzers alive</u>	40			
<u>Finished cycles</u>	40 / 40			
	Σ Total	□ Min	□ Median	□ Max
<u>Run time</u>		19:14:14		19:18:21
<u>Execs</u>	92,702,204	1,311,417	2,370,559	2,899,352
<u>Paths</u>		727	1,229	1,336
<u>Imported</u>		265	669	797
<u>Cumulative speed</u>	1,377.6	1.9	39	48
<u>Pending faves</u>	60	0	0	25
<u>Pending paths</u>	16,656	266	354	701
<u>Hangs occurred</u>		0	0	0
<u>Crashes found</u>		0	0	0

4

test app

Internal application for testing media codecs.
Details cannot be revealed.

test app- Initial Set

Empty – one file with single character was created.

Search: nurs:md2u, nurs:covnew

Max time: 7200

Max size: 16

Result:

KLEE: done: total instructions = 15503460

KLEE: done: completed paths = 1693

KLEE: done: generated tests = 36

test app

Empty Set	KLEE Set
12 Days 22 Hours	10 Days 1 Hour
Total Paths: 1902	Total Paths: 2181
Total Execs: 5.76M	Total Execs: 3.71M

Conclusions

- KLEE can be very helpful when the corpus needs to be generated.
- The effectiveness highly depends on the application structure.

Thank you

Special thanks for Filip Zarzyński