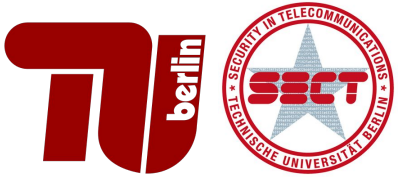
JMPscare

Dominik Maier & Lukas Seidel

Introspection for Binary-Only Fuzzing

4th Workshop on Binary Analysis Research
San Diego, USA (Virtual Event)



JMPscare

Introspection for Binary-Only Fuzzing

Motivation

- Complicated targets require carefully crafted harnesses
  - Analyzing fuzzer's behavior is difficult
  - Often, development stops when fuzzing begins
- But: human needs to stay in the loop

Goals

- Provide deep insight into whole fuzzing queues (thousands of executions)
- Find limitations of your fuzzer/harness

  More concrete:
  find interesting conditional jumps the fuzzer is not able to overcome (*frontiers*)

=> human-in-the-loop can use insights to improve fuzzer, mutator, and harness

The JMPscare Toolkit

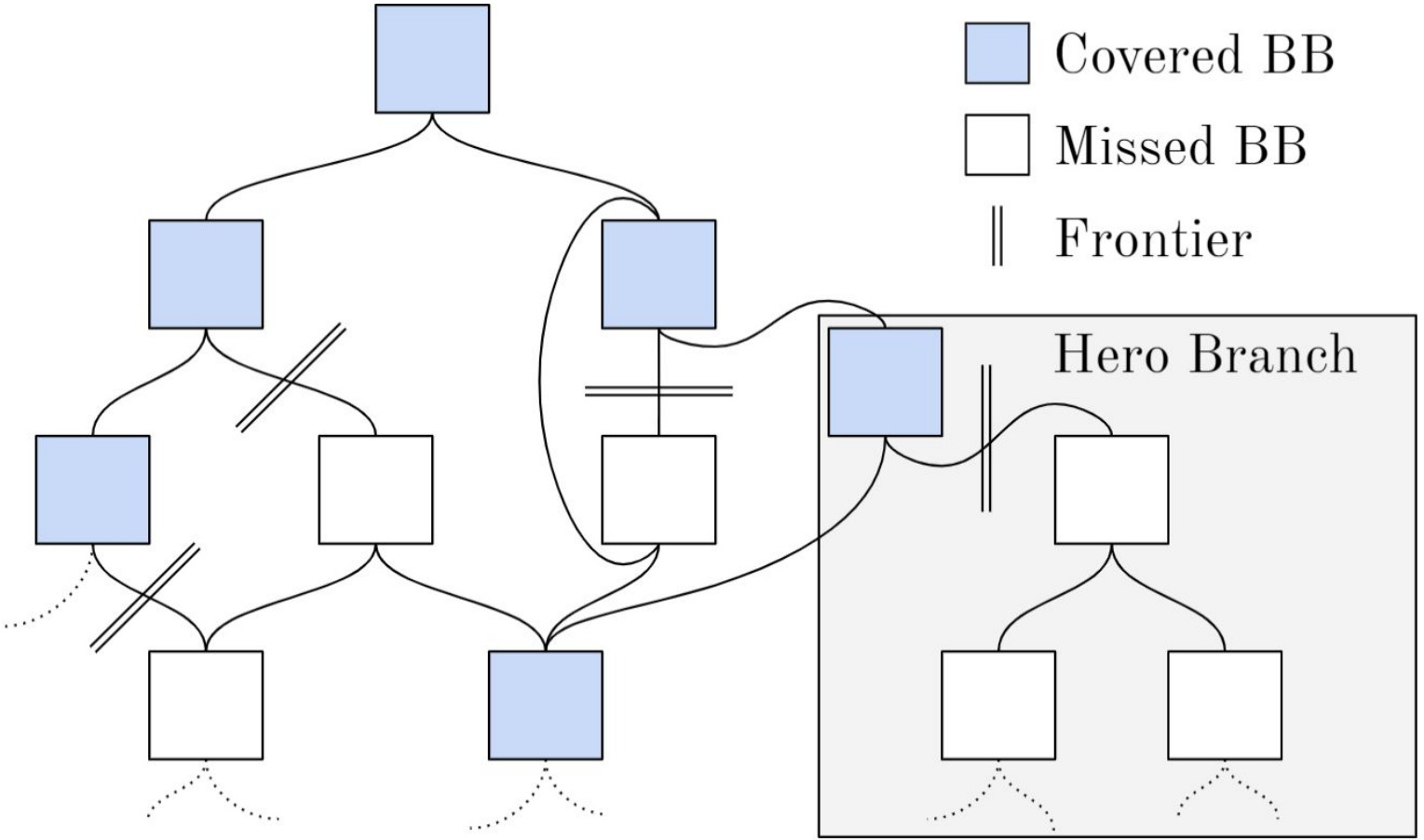
1. Trace Collection
2. Analysis
3. Disassembler Plugin

Trace Collection

- Operates on simplistic execution trace:

  ⇒ one address per line
- Can be e.g., collected during emulation

- We offer plug-and-play solutions for *unicornafl*
  - Python library
  - Rust crate


Covered BB
Missed BB
Frontier
Hero Branch

Automated Frontier Analysis

- Determines
  - which conditional jumps were taken
  - which basic blocks were reached
- Efficiently cross-analyzes thousands of traces
- Result: list of unidirectional jumps which were traversed, but *always* or *never* taken

- Supports x86_64, MIPS, and 32 bit ARM (incl. thumb2)

Potential New Coverage Analysis

- Heuristic for **impact/interestingness** of frontier

- Traverse edges *N* times, count unseen Blocks
- User-defined weighting for unresolvable function calls (`blx r3`)

```
1   pnc_score = 0
2   new_basic_blocks = [frontier_addr]
3   while i < N:
4       for bb_addr in new_basic_blocks:
5           if bb_addr not in SEEN:
6               curr_addr = bb_addr
7               while curr_insn not in JUMPS:
8                   curr_insn = disas(curr_addr)
9                   curr_addr++
10              if curr_insn in UNRES_CALLS:
11                  pnc_score += call_weight
12              else:
13                  pnc_score += 1
14              new_basic_blocks.add(curr_addr.target)
15              new_basic_blocks.remove(bb_addr)
16      i++
17
```

Analysis Output File

- Details about all road-block jumps
  - Address
  - Condition
  - Whether it is taken always or never
  - PNC score

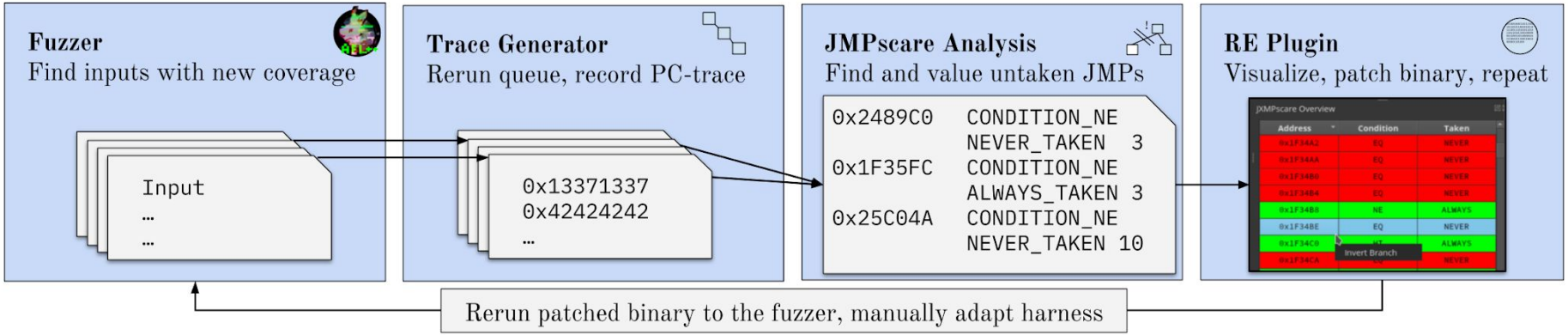
```
0x1172 CONDITION_LT NEVER_TAKEN 15
```

Binary Ninja Plugin

Binary Ninja Plugin (cont.)

- Concise overview of frontier details
- Highlights blocking instructions in disassembly
- Facilitates Forced Execution by auto-patching (through branch inversion)

Complete Pipeline



Frontiers and Basic Block Classification

1) *Reached*.

2) *Reachable*. Fuzzer has capabilities to reach block in a reasonable time (find long path, input satisfying a certain condition…)

3) *Reachable Behind Frontier*. Path exists, unreasonable amount of time required (CF altering state becomes too complex, e.g., deeply nested structs with multiple pointer indirections). Manual aid required.

Frontiers and Basic Block Classification (cont.)

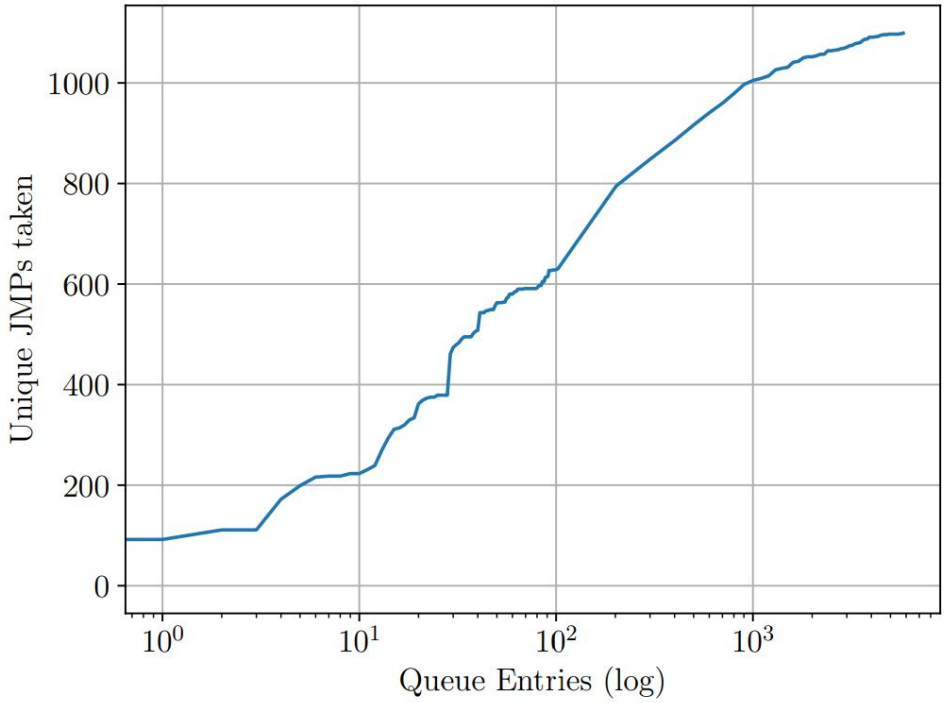
4) *Reachable Altering Precondition*. Control flow to block exists, but hidden behind state that cannot be changed by mutating the input. Solutions: change harness, use different snapshot

5) *Unreachable*.

Evaluation

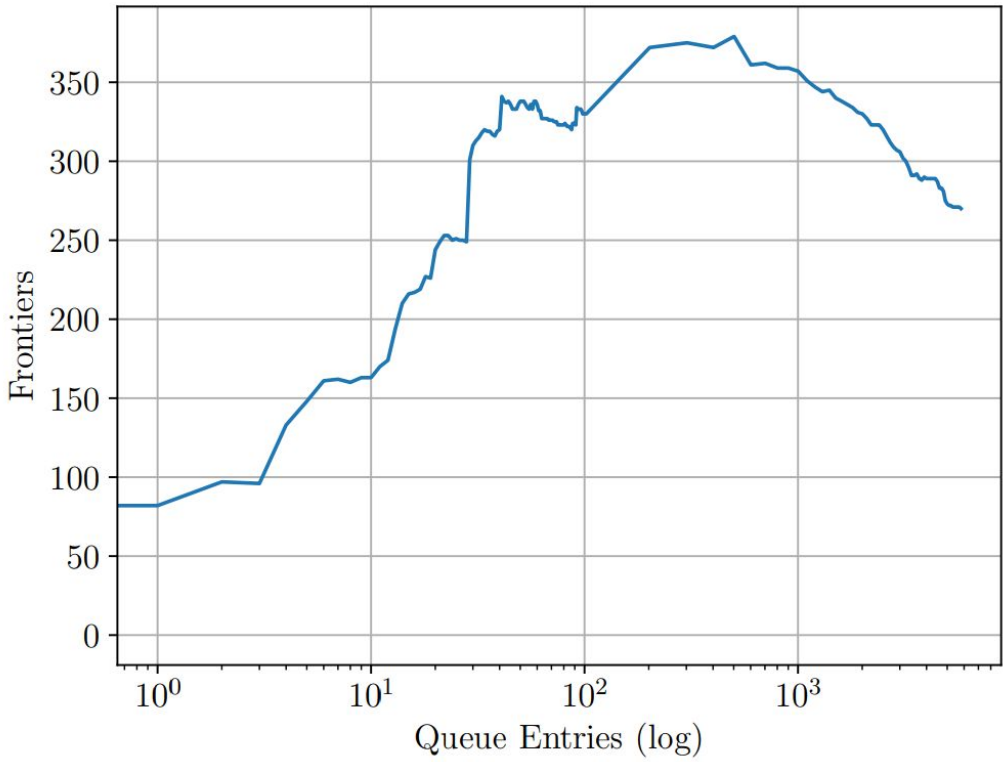
- *BaseSAFE* fuzz queue (MediaTek Helio X10 ARM firmware)
- 5902 inputs
- 5860 executed instructions on average
- 4.16 million jumps
  - 1099 unique
  - 270 frontiers

- Growth stagnates after initial increase
  - more and more difficult for fuzzer to find new coverage yielding inputs



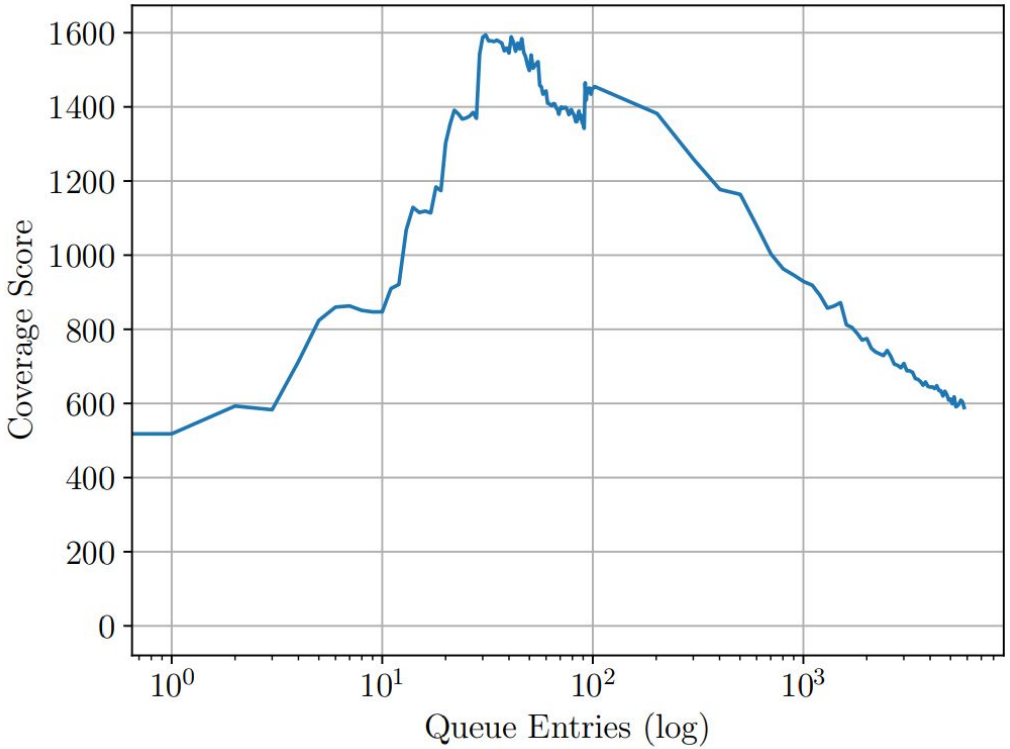
- Additional execution traces
  => more coverage
  => more jumps

- Tipping point:
  more condition-switching inputs than previously unknown jumps are found



- Low number of traces
  => only marginal coverage
  => every new observed frontier may lead to huge new program part

- Increased coverage
  => earlier termination during traversal of unseen edges (PNC analysis)

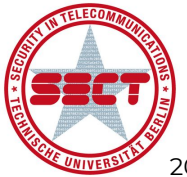


Conclusion

- JMPscare provides insights into the queue
- It finds explorable parts of the binary
- Which helps the tester to improve their harness

Open Source at

https://github.com/fgsect/JMPscare



```
while (questions());

char buf[16];
strncpy(buf, ""
    "Thank you for your attention."
    "\n", sizeof(buf));
printf("%s", buf);
```