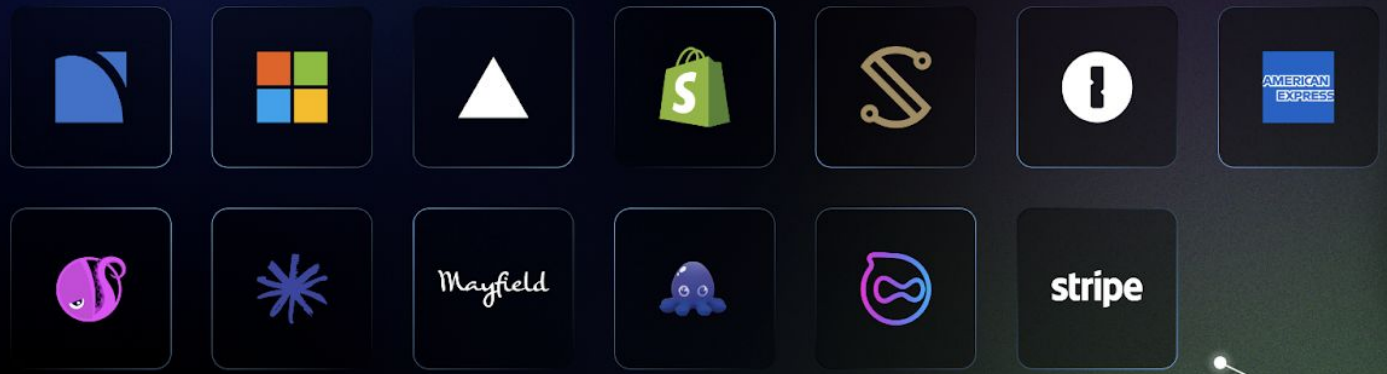
GitHub Secure Open Source Fund

powered by GitHub Sponsors



GitHub Secure Open Source Fund
powered by GitHub Sponsors

Module: Fuzzing

GitHub Secure Open Source Fund
powered by GitHub Sponsors

About Me

Antonio Morales
GitHub
Senior Security Researcher

GitHub Secure
Open Source Fund
powered by GitHub Sponsors

About Me

- Working at GitHub since 2019
- Mainly focused on fuzzing and C/C++ vulnerability research
- Previously worked as an Ethical Hacker (red team, penetration tester)

Motivation

T-Fuzz: fuzzing by program transformation

Hui Peng
Purdue University
peng124@purdue.edu

Yan Shoshitaishvili
Arizona State University
yans@asu.edu

Mathias Payer
Purdue University
mathias.payer@nebelwelt.net

Program	AFL	T-Fuzz
pngfix + libpng (1.7.0)	0	11
tiffinfo + libtiff (3.8.2)	53	124
magick + ImageMagick (7.0.7)	0	2*
pdftohtml + libpoppler (0.62.0)	0	1*

Motivation

Enhancing Memory Error Detection for Large-Scale Applications and Fuzz Testing

Wookhyun Han, KAIST
Byunggill Joe, KAIST
Byoungyoung Lee, Purdue University
Chengyu Song, University of California, Riverside
Insik Shin, KAIST

App.	Description	Popularity		Complexity	Total execs (K)		Total unique crashes		
		GitHub$^{\alpha}$	Debian$^{\beta}$	(LoC)	ASAN	MEDS	ASAN	MEDS	Improv
PH7	PHP interpreter	(35, 321)	-	43K	387	360	8	29	256%
lci	LCODE interpreter	(61, 355)	-	50K	413	820	21	54	157%
picoc	C interpreter	(161, 1240)	-	68K	4,535	6,020	108	231	114%
ImageMagick	Image tool	(212, 933)	-	622K	110	58	9	14	56%
wren	Sciprt language	(190, 1991)	-	13K	340	222	92	110	20%
espruino	JS interpreter	(359, 1157)	-	18K	167	143	260	295	13%
tinyvm	Tiny virtual macine	(123, 1154)	-	30K	182	170	73	80	10%

Motivation

REDQUEEN: Fuzzing with Input-to-State Correspondence

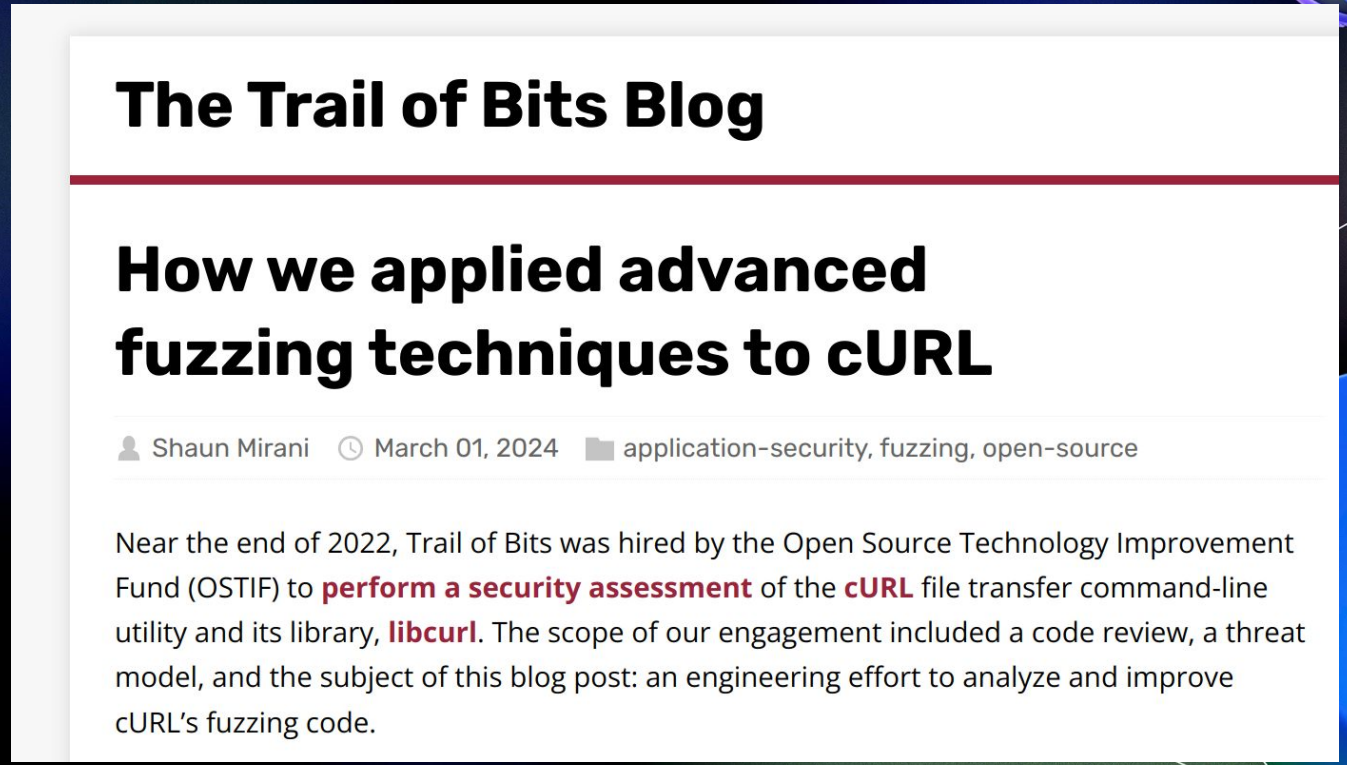
Cornelius Aschermann, Sergej Schumilo, Tim Blazytko, Robert Gawlik and Thorsten Holz
Ruhr-Universität Bochum

TABLE IV: Reported bugs found by REDQUEEN

Application	Version	Bugs
ld-new	binuitils-2.30-15ubuntu1	3
as-new	binuitils-2.30-15ubuntu1	8
gprof	binuitils-2.30-15ubuntu1	2
nm-new	binuitils-2.30-15ubuntu1	1
cxxfilt	binuitils-2.30-15ubuntu1	1
objdump	binuitils-2.30-15ubuntu1	11
tiff2ps	libtiff-4.0.9	4
jhead	3.00-6	13
fdk-acc	v0.1.6	2
ImageMagick	7.0.7-29	3
wine	wine-2.0.2-2ubuntu1	3

GitHub Secure Open Source Fund
powered by GitHub Sponsors

Motivation



The Trail of Bits Blog

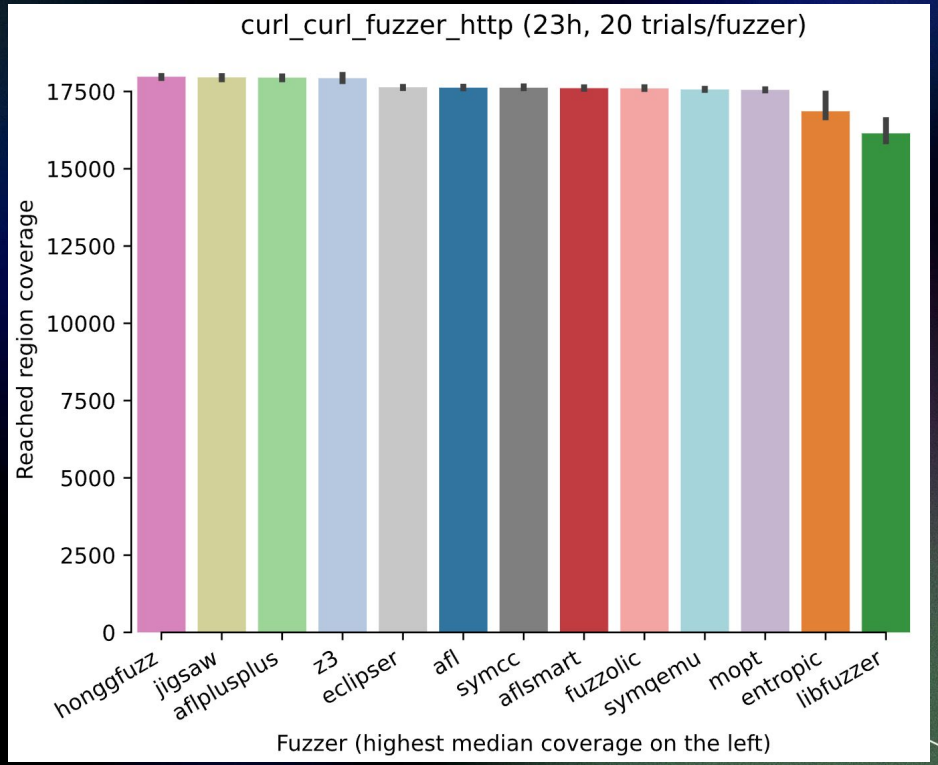
How we applied advanced fuzzing techniques to cURL

Shaun Mirani March 01, 2024 application-security, fuzzing, open-source

Near the end of 2022, Trail of Bits was hired by the Open Source Technology Improvement Fund (OSTIF) to **perform a security assessment** of the **cURL** file transfer command-line utility and its library, **libcurl**. The scope of our engagement included a code review, a threat model, and the subject of this blog post: an engineering effort to analyze and improve cURL's fuzzing code.

GitHub Secure Open Source Fund
powered by GitHub Sponsors

Motivation



Motivation

LLVM Fuzzing Audit

Fuzzing Audit Report

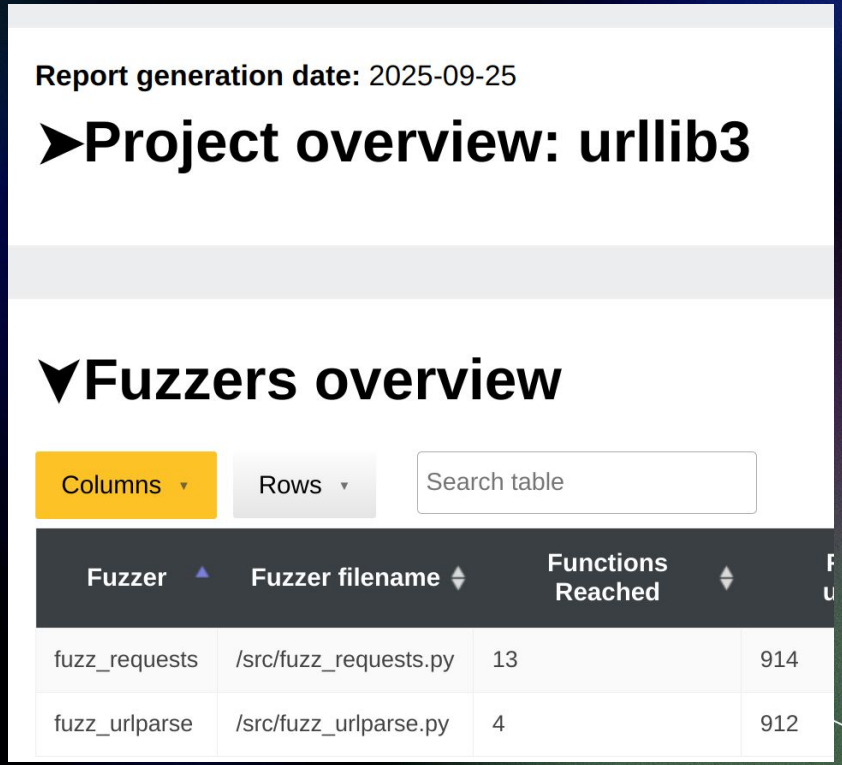
Adam Korczynski, David Korczynski

2024-01-11

4 Issues found and fixed

Motivation

GitHub Secure Open Source Fund
powered by GitHub Sponsors



Report generation date: 2025-09-25

➤Project overview: urllib3

➤Fuzzers overview

Columns ▾ Rows ▾ Search table

Fuzzer	Fuzzer filename	Functions Reached	F u
fuzz_requests	/src/fuzz_requests.py	13	914
fuzz_urlparse	/src/fuzz_urlparse.py	4	912

3rd edition

GitHub Secure
Open Source Fund
powered by GitHub Sponsors

2021 — Fuzzing workshop

2022 — Fuzzing workshop 2nd edition

2025 March — GitHub Secure - 1st edition

2025 June — GitHub Secure - 2nd edition

Workshop origins

- My original workshop focused on C/C++ and I made use of the AFL fuzzer
- Included 3 challenges that got progressively harder
- Presented at several security conferences, including:
  - Ekoparty
  - Hack In Paris
  - Hackfest

GitHub Secure
Open Source Fund
powered by GitHub Sponsors

1st edition

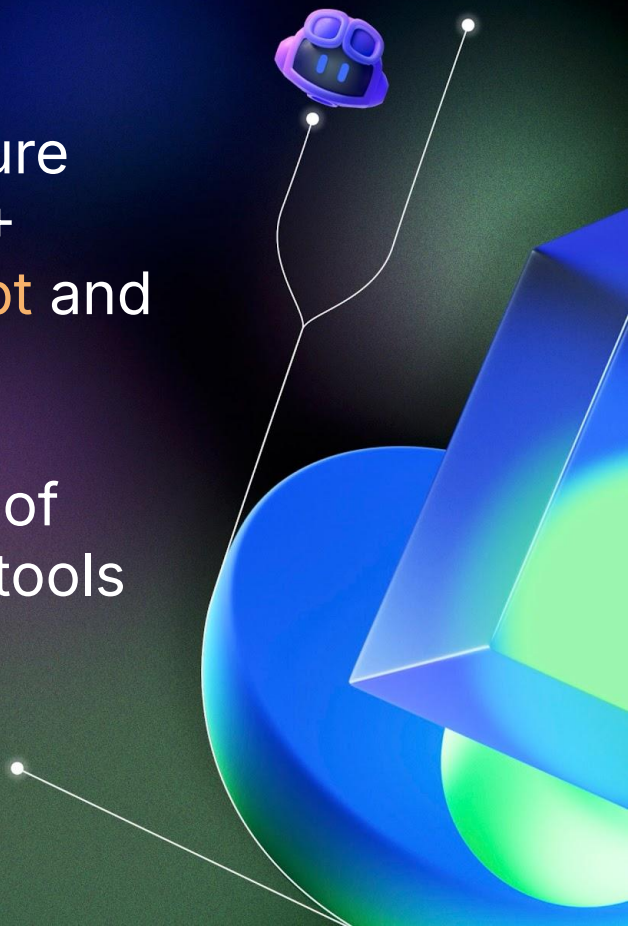
- For the first edition of the GitHub Secure Fuzzing Workshop, I simplified my original workshop, removing 2 of the 3 challenges
- Even with these changes, most participants still found it difficult to follow
- The main feedback was the lack of C/C++ background

GitHub Secure
Open Source Fund
powered by GitHub Sponsors

2nd edition

- For the 2nd edition of the GitHub Secure fuzzing workshop I removed all C/C++ content and replaced it with JavaScript and Python examples

- The main complaint was the difficulty of setting up the correct versions of the tools (Java, Node.js, Python, etc.)



GitHub Secure
Open Source Fund
powered by GitHub Sponsors

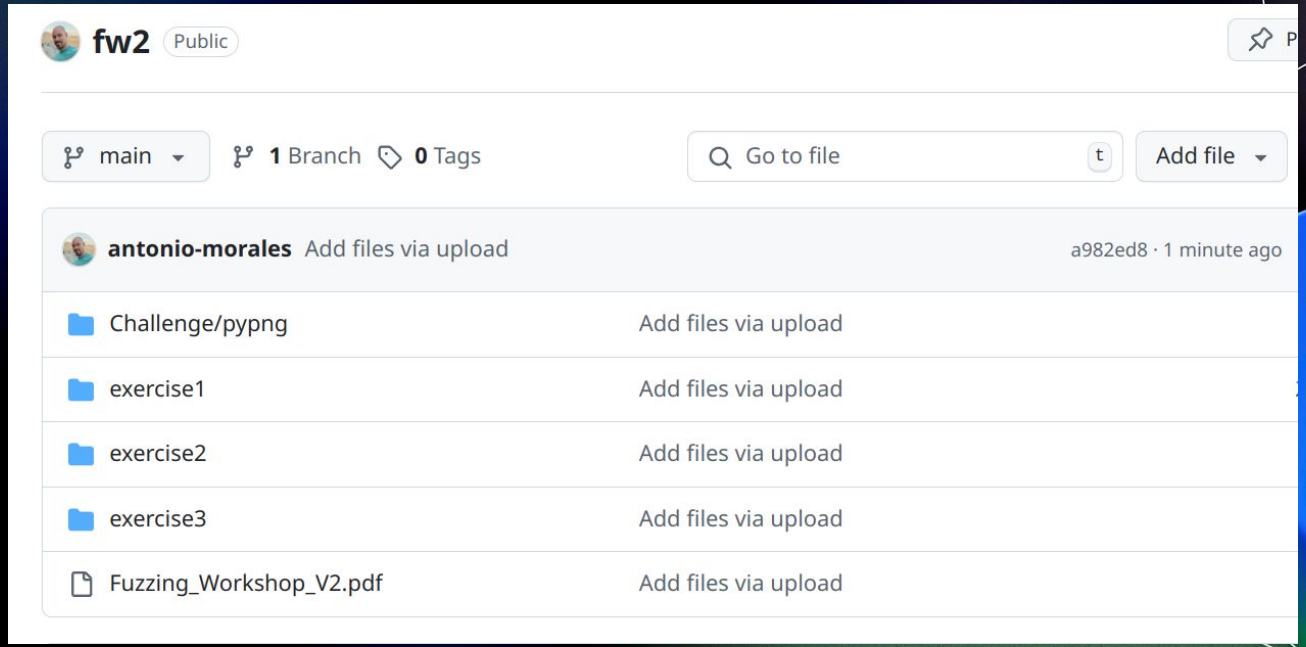
3rd edition

- For this 3rd edition, I simplified the fuzzing workshop even further and added a *devcontainer* to automate tools setup
- I still recommend using a host Linux environment (Ubuntu 24.04 LTS) for the best fuzzing experience
- If you don’t have access to a Linux machine, GitHub Codespaces is a good option

GitHub Secure Open Source Fund
powered by GitHub Sponsors

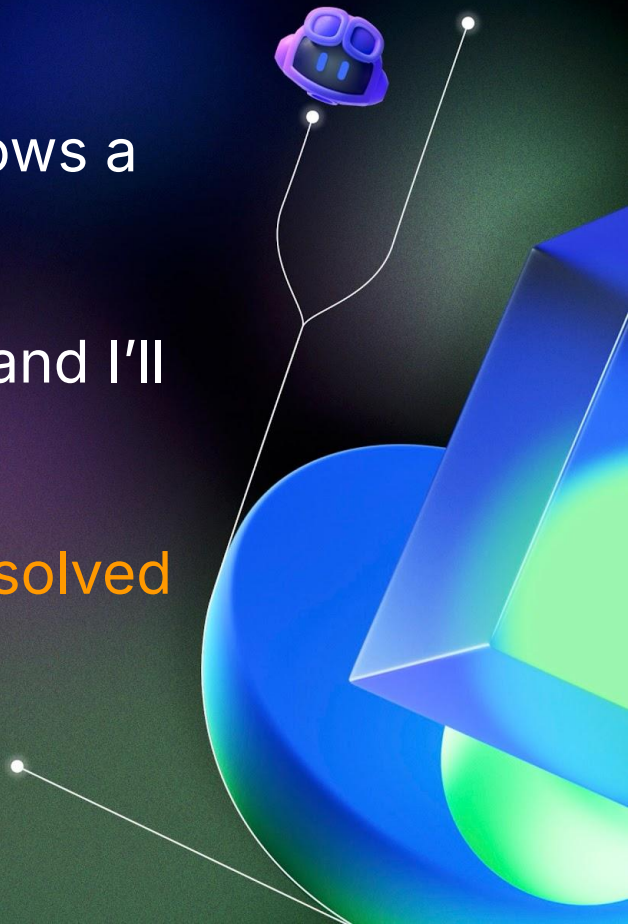
Workshop Repository

https://github.com/antonio-morales/fw2



Workshop Format

- This is a hands-on workshop that follows a learning-by-doing approach
- You will learn by tackling challenges, and I'll be here to guide you along the way
- These challenges are designed to be solved using fuzzing, but feel free to use any method you prefer (good luck xD)



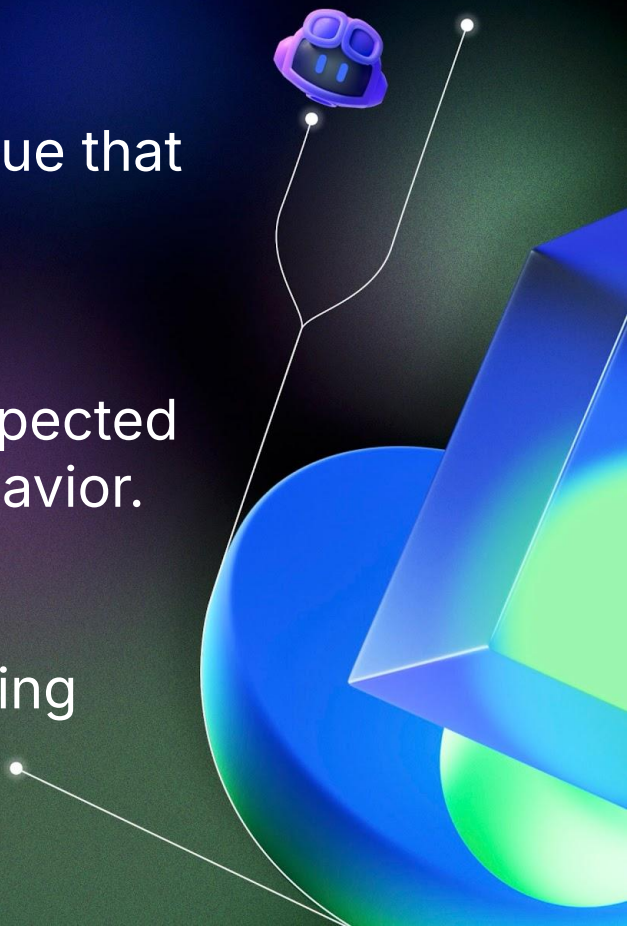
GitHub Secure Open Source Fund
powered by GitHub Sponsors

Agenda

1. Fuzzing introduction
2. Coverage-guided fuzzing
3. Exercise 1: Introduction to Java fuzzing
4. Exercise 2: Introduction to Javascript fuzzing
5. Exercise 3: Introduction to Python fuzzing
6. More fuzzing
7. The fuzzing workflow
8. Challenge
9. Next steps for learning
10. Homework
11. Your questions

1. Fuzzing introduction

- Fuzzing is an automated testing technique that helps find software bugs and security vulnerabilities
- It works by injecting malformed or unexpected inputs into a program to observe its behavior.
- In recent years, its use has consistently increased due to its success in uncovering vulnerabilities



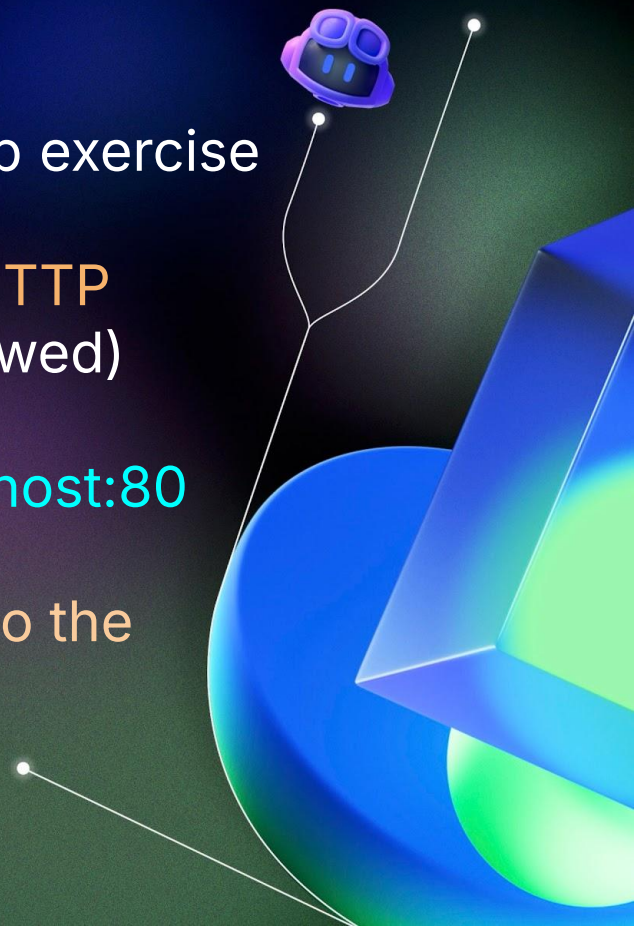
The simplest fuzzer

```
cat /dev/urandom | ./your_application
```

- cat reads random data from */dev/urandom* and pipes it into *your_application*
- In 2009, Charlie Miller used this fuzzing technique to discover critical vulnerabilities in iPhones

Warm-up exercise

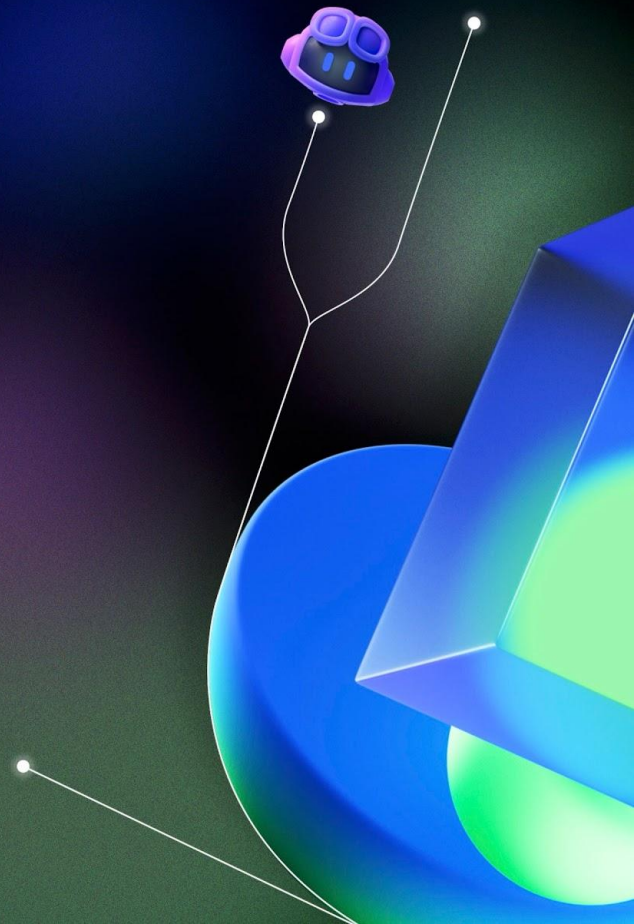
- Let’s start the workshop with a warm-up exercise
- You have 10 minutes to build a simple HTTP fuzzer (any programming language allowed)
- The fuzzer must connect to http://localhost:80
- Submit your solution as a Pull Request to the /warmup folder



GitHub Secure Open Source Fund
powered by GitHub Sponsors

Warm-up exercise

10 minutes



Smart Fuzzing

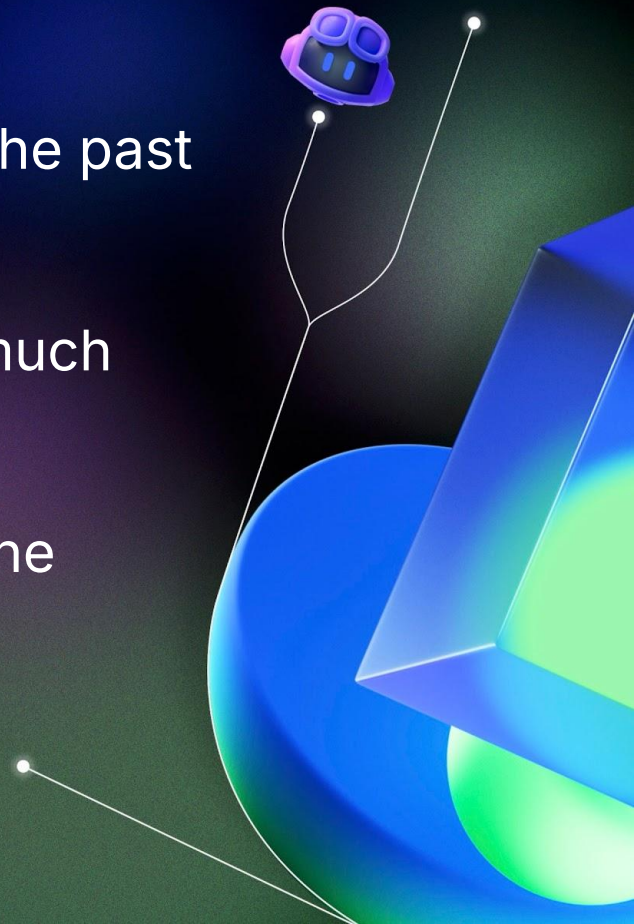
GitHub Secure Open Source Fund
powered by GitHub Sponsors

Dumb Fuzzing

Smart Fuzzing

Smart Fuzzing

- Fuzzing has significantly evolved over the past 15 years
- New fuzzing techniques have made it much more effective
- The release of the AFL fuzzer marked the beginning of the “smart fuzzing” era

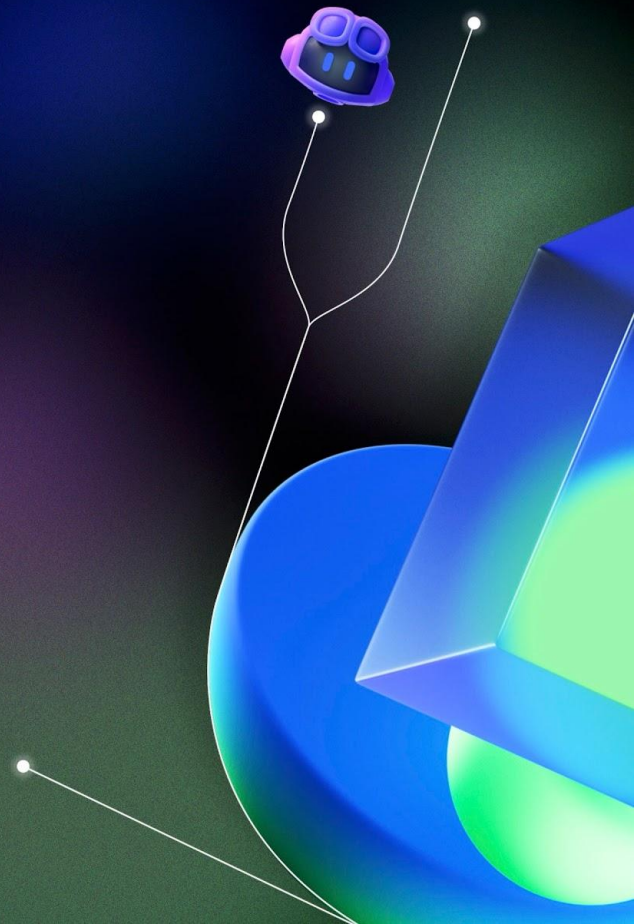


AFL

GitHub Secure Open Source Fund
powered by GitHub Sponsors

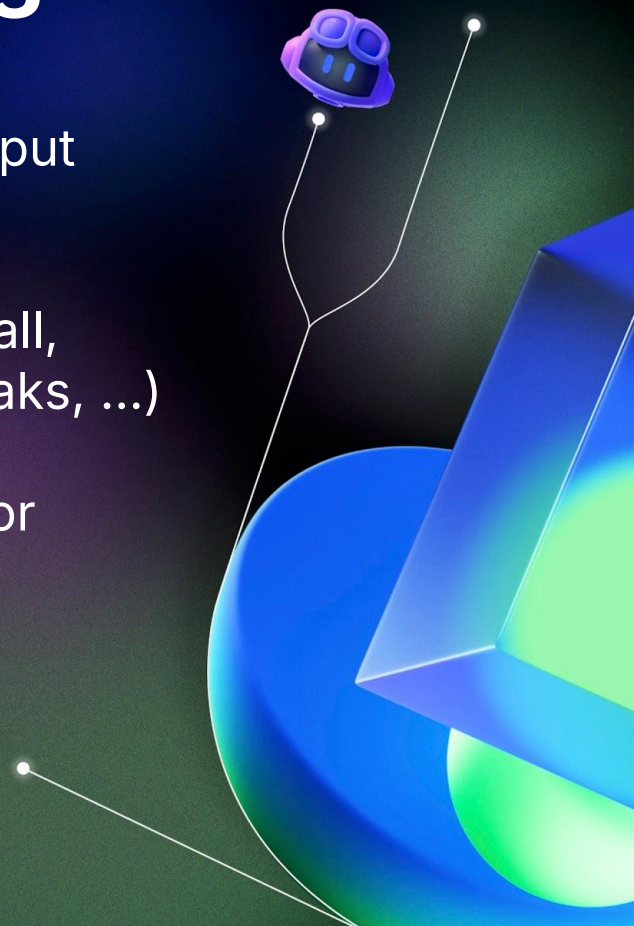
- AFL is an

Evolutionary
Mutation-based
Coverage-guided
fuzzer



2. Mutation-based fuzzing

- A mutation-based fuzzer starts with valid input samples (the corpus)
- It generates new test cases by applying small, random changes (bit/byte flips, integer tweaks, ...)
- The goal is to trigger unexpected behavior or crashes

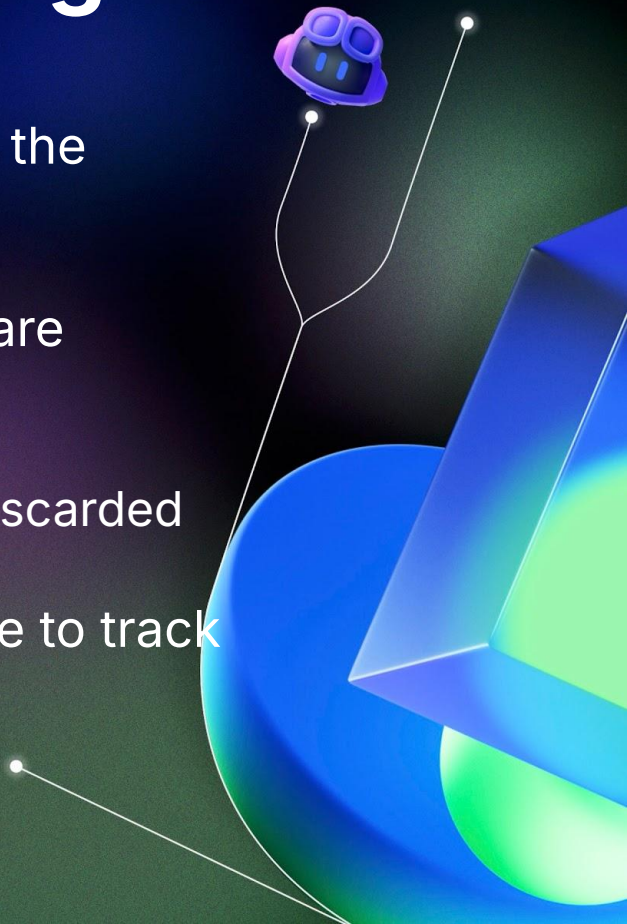


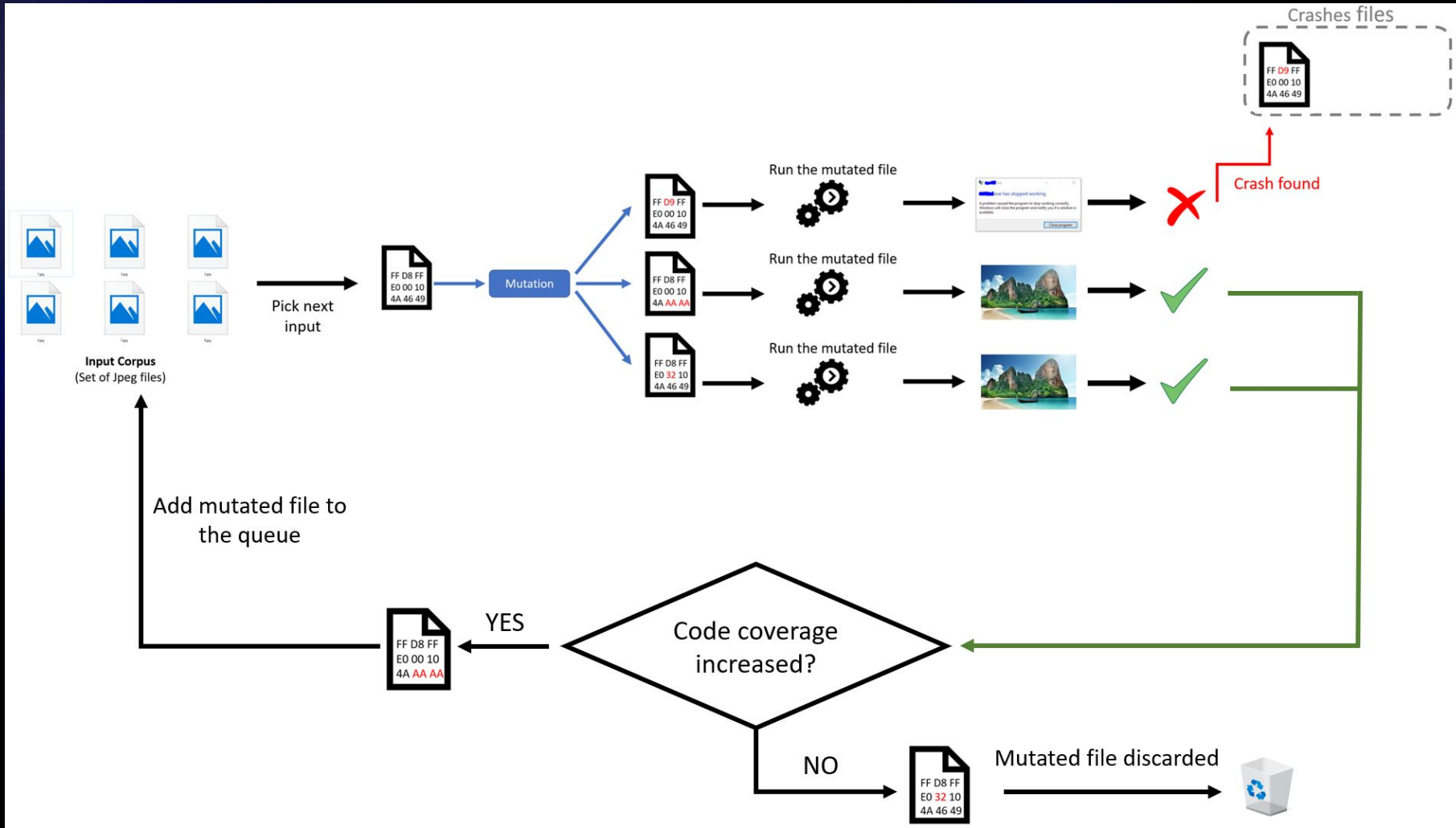
2. Evolutionary fuzzing

- Based on evolutionary algorithms: inputs evolve over generations according to the principles of natural selection
- It uses some kind of “feedback” from the target program to guide future mutations
- This helps generate inputs that reach deeper and more complex program behavior

2. Coverage-guided fuzzing

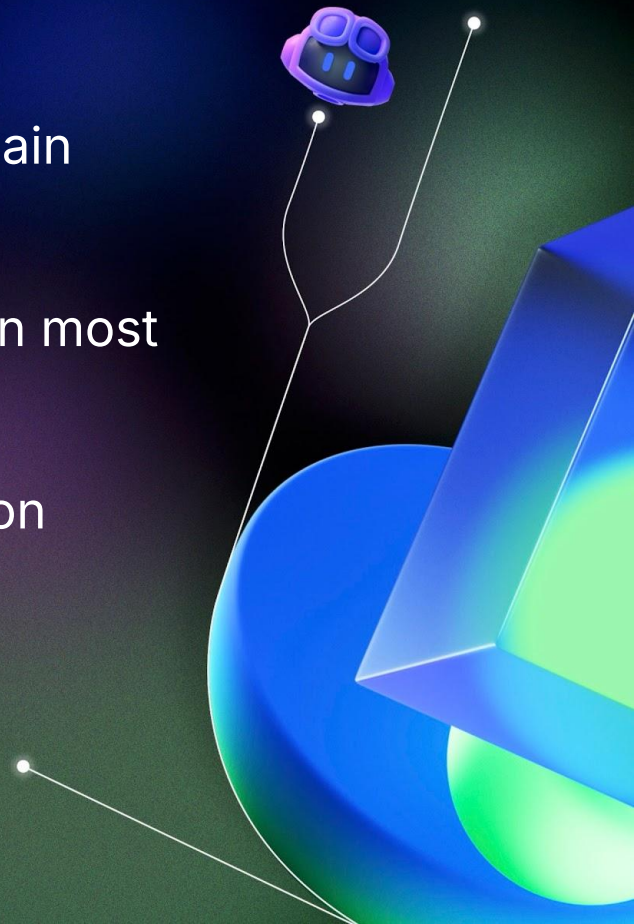
- It uses code coverage as feedback to guide the fuzzing process
- Mutations that trigger new execution paths are added to the input queue
- Mutations that don't discover new paths are discarded
- These fuzzers need to “instrument” the code to track coverage




Crashes files
FF D9 FF
E0 00 10
4A 46 49
Crash found
Run the mutated file
Run the mutated file
Run the mutated file
FF D9 FF
E0 00 10
4A 46 49
FF D8 FF
E0 00 10
4A AA AA
FF D8 FF
E0 32 10
4A 46 49
Mutation
FF D8 FF
E0 00 10
4A 46 49
Pick next
input
Input Corpus
(Set of Jpeg files)
Add mutated file to
the queue
FF D8 FF
E0 00 10
4A AA AA
YES
Code coverage
increased?
NO
FF D8 FF
E0 32 10
4A 46 49
Mutated file discarded

LibFuzzer

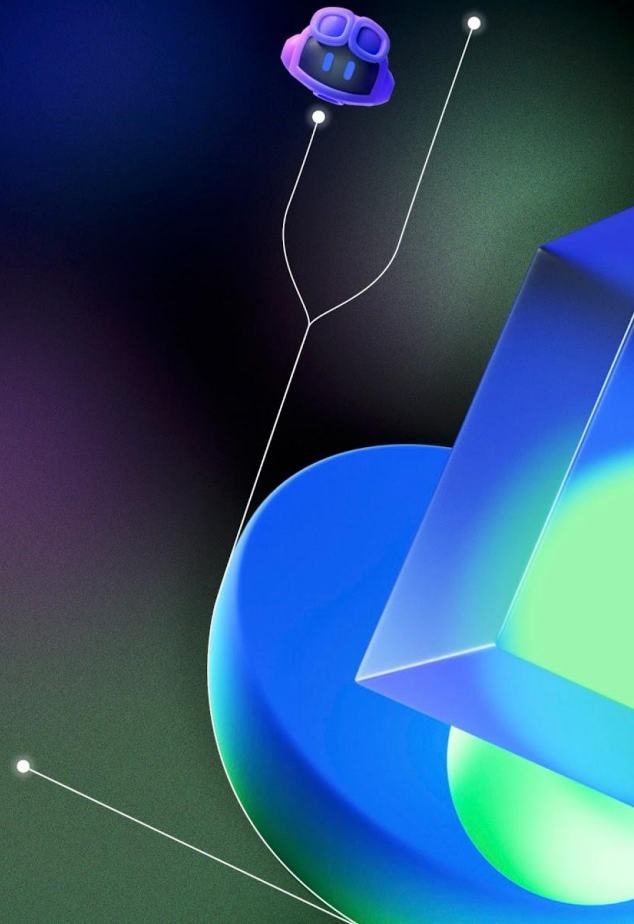
- Developed by Google, it's considered the main "rival" to AFL
- It's part of the LLVM project and available on most platforms
- All the fuzzers we'll cover today are based on LibFuzzer



Exercise 1

GitHub Secure Open Source Fund
powered by GitHub Sponsors

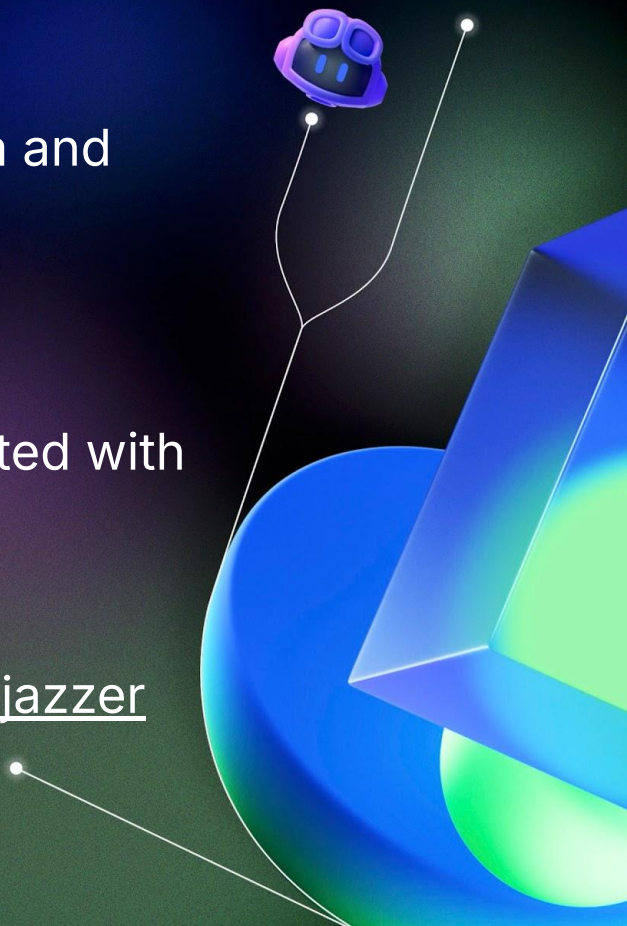
JAZZER



GitHub Secure Open Source Fund
powered by GitHub Sponsors

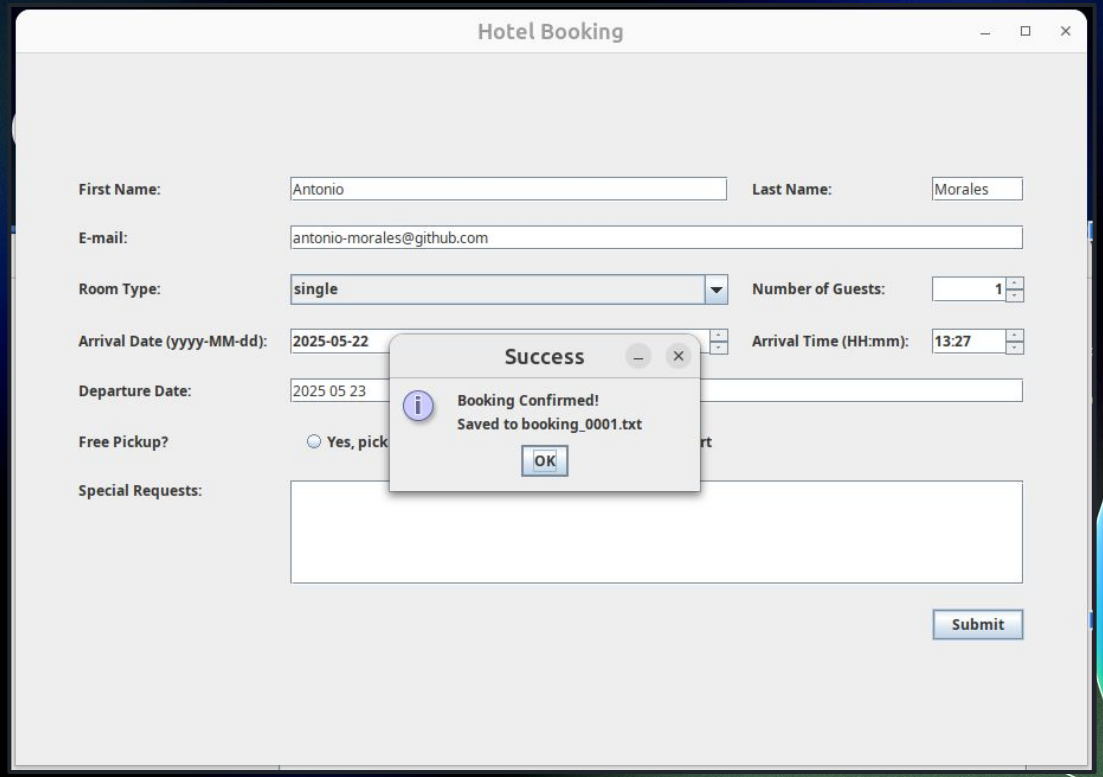
Jazzer

- Jazzer is a coverage-guided fuzzer for Java and other JVM languages
- It's built on libFuzzer
- Developed by Code Intelligence and integrated with Google's OSS-Fuzz
- Available at https://github.com/CodeIntelligenceTesting/jazzer



Exercise 1

GitHub Secure Open Source Fund
powered by GitHub Sponsors



Exercise 1

```
$ java --version
openjdk 21.0.3 2024-04-16
OpenJDK Runtime Environment (build 21.0.3+9-Ubuntu-1ubuntu1)
OpenJDK 64-Bit Server VM (build 21.0.3+9-Ubuntu-1ubuntu1, mixed mode, sharing)
```

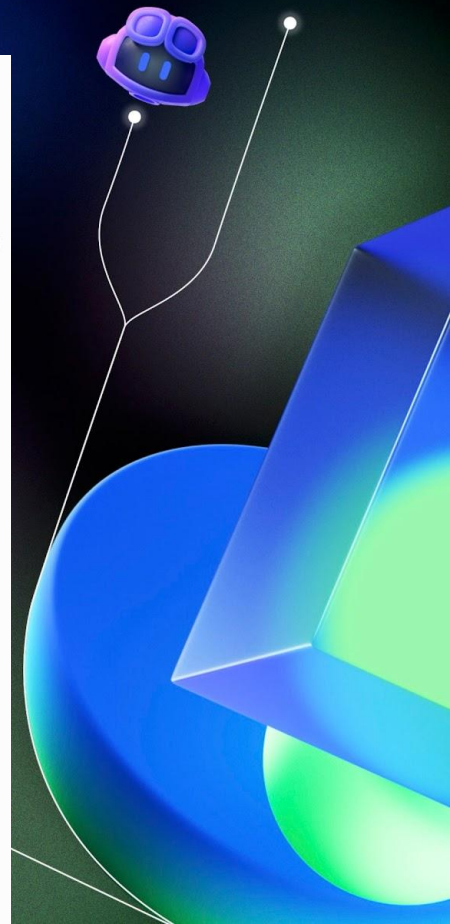
```
javac BookingFormGUI.java BookingForm.java
```

```
java BookingFormGUI
```

Exercise 1

GitHub Secure Open Source Fund
powered by GitHub Sponsors

```java
import java.io.StringReader;

public class fuzzer {

    // Jazzer expects a public static method named fuzzerTestOneInput
    public static void fuzzerTestOneInput(byte[] data) {

        String s = new String(data);

        try {

          BookingForm.parseBooking(s);

        } catch (ArrayIndexOutOfBoundsException e) {

          throw e;

        } catch (Exception e) {
            // Ignore
        }
    }
}
```



Exercise 1

GitHub Secure Open Source Fund
powered by GitHub Sponsors

```
javac -d out BookingForm.java fuzzer.java

chmod +x ./jazzer/jazzer

./jazzer/jazzer --cp=out --target_class=fuzzer
```

Exercise 1

```
./jazzer/jazzer --cp=out --target_class=fuzzer corpus
```

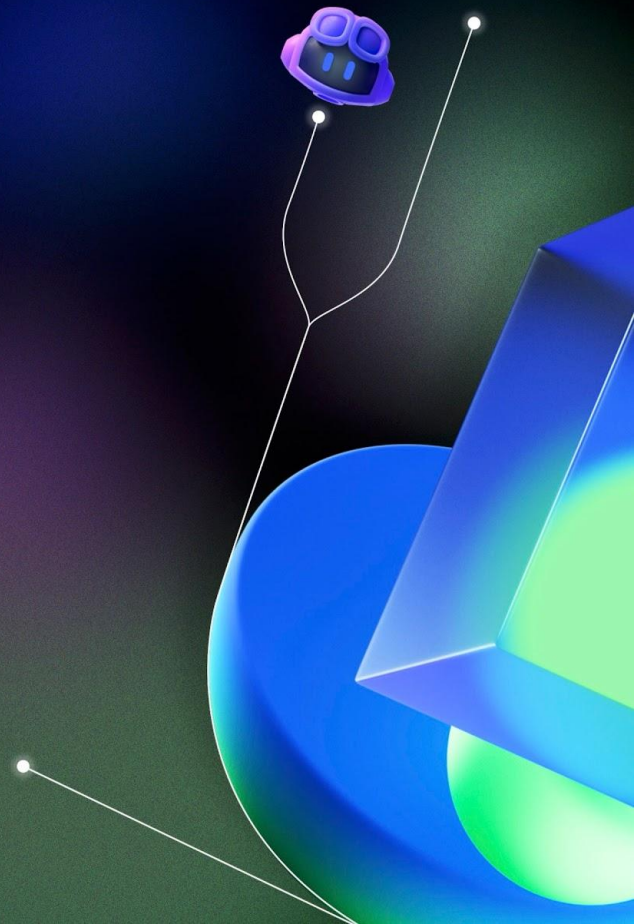
Exercise 1

```
#3782366        REDUCE cov: 67 ft: 67 corp: 21/665b lim: 4096 exec/s: 343851 rss: 1024Mb L: 45/50 MS: 4 ChangeBit-Custom-EraseBytes-Custom-
#4011532        REDUCE cov: 67 ft: 67 corp: 21/664b lim: 4096 exec/s: 334294 rss: 1024Mb L: 45/50 MS: 2 EraseBytes-Custom-
#4194304        pulse  cov: 67 ft: 67 corp: 21/664b lim: 4096 exec/s: 349525 rss: 1024Mb
#8388608        pulse  cov: 67 ft: 67 corp: 21/664b lim: 4096 exec/s: 335544 rss: 1024Mb

== Java Exception: java.lang.ArrayIndexOutOfBoundsException: Index 11 out of bounds for length 11
        at BookingForm.parseDate(BookingForm.java:195)
        at BookingForm.parseBooking(BookingForm.java:117)
        at fuzzer.fuzzerTestOneInput(fuzzer.java:12)
DEDUP_TOKEN: 2812ded32f26d6bb
== libFuzzer crashing input ==
MS: 4 CopyPart-Custom-PersAutoDict-Custom- DE: "ie"-; base unit: 4f5626fcb1b6c23d2b5fa2367f599fe08094c2ba
0x6e,0x3b,0x4d,0x3b,0x6c,0x40,0x6d,0x3b,0x73,0x75,0x69,0x74,0x65,0x3b,0x31,0x3b,0x32,0x30,0x32,0x35,0x2d,0x30,0x35,0x2d,0x32,0x31,0x54,0x32,0
,0x33,0x3b,0x66,0x61,0x6c,0x73,0x65,0x3b,
n;M;l@m;suite;1;2025-05-21T21:48;ie 05 2323;false;
artifact_prefix='./'; Test unit written to ./crash-e7651f9f08c6dc8be1f2eb8d97f310037d5cd54f
Base64: bjtNO2xAbTtzdWl0ZTsxOzIwMjUtMDUtMjFUMjE6NDg7aWUgMDUgMjMyMztmYWxzZTs=
reproducer_path='.'; Java reproducer written to ./Crash_e7651f9f08c6dc8be1f2eb8d97f310037d5cd54f.java
```

Exercise 2

GitHub Secure Open Source Fund
powered by GitHub Sponsors

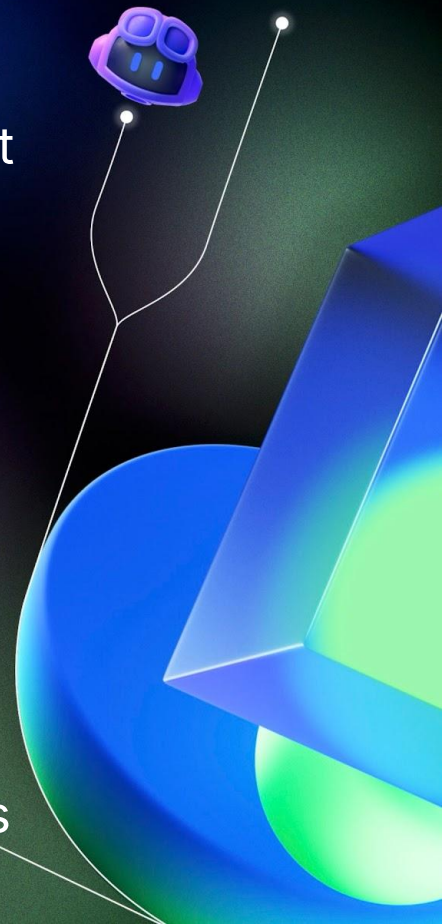
JAZZER.JS



GitHub Secure Open Source Fund
powered by GitHub Sponsors

Jazzer.js

- Jazzer.js is a coverage-guided fuzzer for JavaScript and Node.js applications
- It’s built on libFuzzer and uss V8's built-in code coverage feedback
- Developed by Code Intelligence
- Good for fuzzing JS libraries and APIs
- Available at https://github.com/CodeIntelligenceTesting/jazzer.js



GitHub Secure Open Source Fund
powered by GitHub Sponsors

Moment.js

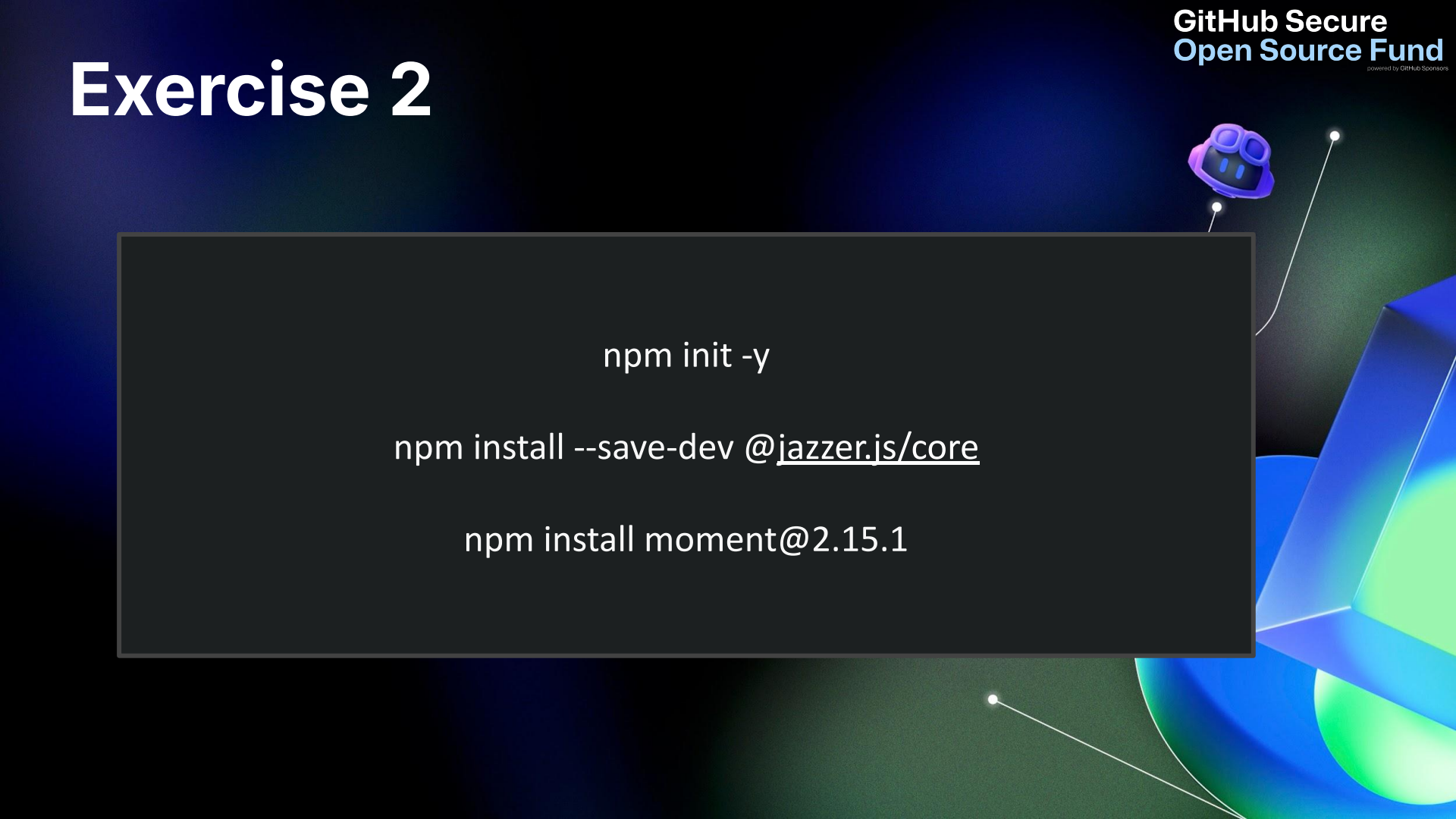
- Moment.js is a library to parse, validate, manipulate, and display dates and times in JavaScript
- Source code is available at: github.com/moment/moment
- We'll use Jazzer.js to find a Regular Expression Denial of Service (ReDoS) vulnerability in version 2.15.1

Exercise 2

```
$ npm -v
9.2.0
$ node -v
v18.19.1
```

nvm install 18.19.1
nvm ls
nvm use 18.19.1

Exercise 2



```
npm init -y

npm install --save-dev @jazzer.js/core

npm install moment@2.15.1
```

Exercise 2

```
node sample.js
```

Exercise 2

GitHub Secure Open Source Fund
powered by GitHub Sponsors

```
const { FuzzedDataProvider } = require('@jazzer.js/core');
const moment = require('moment');

// Set locale to one with vulnerable standalone month parsing
moment.locale('be');

/**
 * The fuzz target: Jazzer.js will call this function with arbitrary input.
 */
function fuzz(data) {

  const provider = new FuzzedDataProvider(data);

  const fmtString = provider.consumeRemainingAsString();

  try {
    moment().format(fmtString);
  } catch (e) {
    // Ignore errors; we're interested in performance slowdown
  }
}

module.exports = { fuzz };
```

Exercise 2

GitHub Secure Open Source Fund
powered by GitHub Sponsors

```
$ npm -v
9.2.0
$ node -v
v18.19.1
```

npx jazzer fuzzer

Exercise 2

GitHub Secure
Open Source Fund
powered by GitHub Sponsors

```
Dictionary: 4 entries
INFO: Running with entropic power schedule (0xFF, 100).
INFO: Seed: 1735645154
INFO: Loaded 1 modules   (512 inline 8-bit counters): 512 [0x7ffff3cff010, 0x7ffff3cff210),
INFO: Loaded 1 PC tables (512 PCs): 512 [0x7fffe9000010,0x7fffe9002010),
INFO: -max_len is not provided; libFuzzer will not generate inputs larger than 4096 bytes
INFO: A corpus is not provided, starting from an empty corpus
#2      INITED cov: 3 ft: 3 corp: 1/1b exec/s: 0 rss: 142Mb
#65536  pulse  cov: 3 ft: 3 corp: 1/1b lim: 652 exec/s: 32768 rss: 158Mb
#131072 pulse  cov: 3 ft: 3 corp: 1/1b lim: 1300 exec/s: 32768 rss: 177Mb
#262144 pulse  cov: 3 ft: 3 corp: 1/1b lim: 2611 exec/s: 29127 rss: 216Mb
ALARM: working on the last Unit for 7 seconds
       and the timeout value is 5 (use -timeout=N to change)
MS: 5 InsertRepeatedBytes-ManualDict-InsertRepeatedBytes-CrossOver-ChangeByte- DE: "__proto__"-; base unit: adc83b19e793491b1c6ea0fd8b46cd9f32e592fc
0xcd,0xcd,0xcd,0xcd,0xcd,0x5f,0x5f,0x70,0x72,0x6f,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8
d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0xa,0x8d,0x8d,0x8d,0x8d,0x8d,0x44,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x
8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,
0x74,0x6f,0x5f,0x5f,0xcd,0xcd,0xa,
\315\315\315\315\315__pro\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\2
15\215\215\215\215\215\215\215\012\215\215\215\215\215D\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215
\215\215\215\215\215\215\215\215\215\215\215\215to__\315\315\012
artifact_prefix='./'; Test unit written to ./timeout-e7844097aec01beee0651295dfab04373bee3636
Base64: zc3Nzc1fX3ByD42NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjQqNjY2NjUSNjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY10b19fzc0K
==273907== ERROR: libFuzzer: timeout after 7 seconds
SUMMARY: libFuzzer: timeout
```

GitHub Secure Open Source Fund
powered by GitHub Sponsors

Exercise 2

```
npx jazzer fuzzer -- -seed=1735645154
```

```
Dictionary: 4 entries
INFO: Running with entropic power schedule (0xFF, 100).
INFO: Seed: 1735645154
INFO: Loaded 1 modules   (512 inline 8-bit counters): 512 [0x7ffff3cff010, 0x7ffff3cff210),
INFO: Loaded 1 PC tables (512 PCs): 512 [0x7fffe9000010,0x7fffe9002010),
INFO: -max_len is not provided; libFuzzer will not generate inputs larger than 4096 bytes
INFO: A corpus is not provided, starting from an empty corpus
#2      INITED cov: 3 ft: 3 corp: 1/1b exec/s: 0 rss: 142Mb
#65536  pulse  cov: 3 ft: 3 corp: 1/1b lim: 652 exec/s: 32768 rss: 158Mb
#131072 pulse  cov: 3 ft: 3 corp: 1/1b lim: 1300 exec/s: 32768 rss: 177Mb
#262144 pulse  cov: 3 ft: 3 corp: 1/1b lim: 2611 exec/s: 29127 rss: 216Mb
ALARM: working on the last Unit for 7 seconds
       and the timeout value is 5 (use -timeout=N to change)
MS: 5 InsertRepeatedBytes-ManualDict-InsertRepeatedBytes-CrossOver-ChangeByte- DE: "__proto__"-; base unit: adc83b19e793491b1c6ea0fd8b46cd9f32e592fc
0xcd,0xcd,0xcd,0xcd,0xcd,0x5f,0x5f,0x70,0x72,0x6f,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8
d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0xa,0x8d,0x8d,0x8d,0x8d,0x8d,0x44,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x
8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,0x8d,
0x74,0x6f,0x5f,0x5f,0xcd,0xcd,0xa,
\315\315\315\315\315__pro\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\2
15\215\215\215\215\215\215\215\012\215\215\215\215\215D\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215\215
\215\215\215\215\215\215\215\215\215\215\215\215to__\315\315\012
artifact_prefix='./'; Test unit written to ./timeout-e7844097aec01beee0651295dfab04373bee3636
Base64: zc3Nzc1fX3Byb42NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjQqNjY2NjUSNjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY10b19fzc0K
==273907== ERROR: libFuzzer: timeout after 7 seconds
SUMMARY: libFuzzer: timeout
```

Exercise 2

GitHub Secure Open Source Fund
powered by GitHub Sponsors

```
67  67    // LOCALES
68  68
69      - var MONTHS_IN_FORMAT = /D[oD]?(\[[^\[\]]*\]|\s+)+MMMM?/;
    69  + var MONTHS_IN_FORMAT = /D[oD]?(\[[^\[\]]*\]|\s)+MMMM?/;
70  70    export var defaultLocaleMonths = 'January_February_March_April_May_June_July_August_Sep
71  71    export function localeMonths (m, format) {
72  72        if (!m) {
```

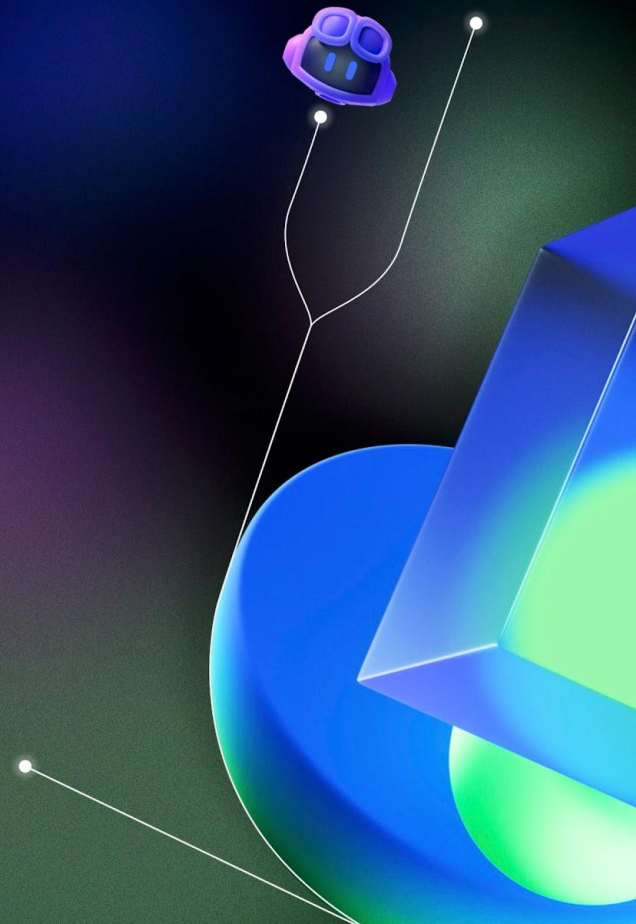
src/locale/lt.js

```
    @@ -50,7 +50,7 @@ export default moment.defineLocale('lt', {
50  50        months : {
51  51            format: 'sausio_vasario_kovo_balandžio_gegužės_birželio_liepos_rugpjūčio_rugsėj
52  52            standalone: 'sausis_vasaris_kovas_balandis_gegužė_birželis_liepa_rugpjūtis_rugs
53      -         isFormat: /D[oD]?(\[[^\[\]]*\]|\s+)+MMMM?|MMMM?(\[[^\[\]]*\]|\s+)+D[oD]?/
    53  +         isFormat: /D[oD]?(\[[^\[\]]*\]|\s)+MMMM?|MMMM?(\[[^\[\]]*\]|\s)+D[oD]?/
```

Exercise 3

GitHub Secure Open Source Fund
powered by GitHub Sponsors

Atheris



GitHub Secure Open Source Fund
powered by GitHub Sponsors

Atheris

- Atheris is a coverage-guided fuzzer for Python and C extension modules
- Developed by Google, it integrates with libFuzzer for feedback-driven fuzzing
- It supports fuzzing both:
  - Pure Python code
  - Native Python extensions (ASAN, UBSAN, etc)
- Available at: https://github.com/google/atheris

GitHub Secure
Open Source Fund
powered by GitHub Sponsors

Mistune

- “A fast yet powerful Python Markdown parser with renderers and plugins”
- Convert Markdown to HTML with ease:

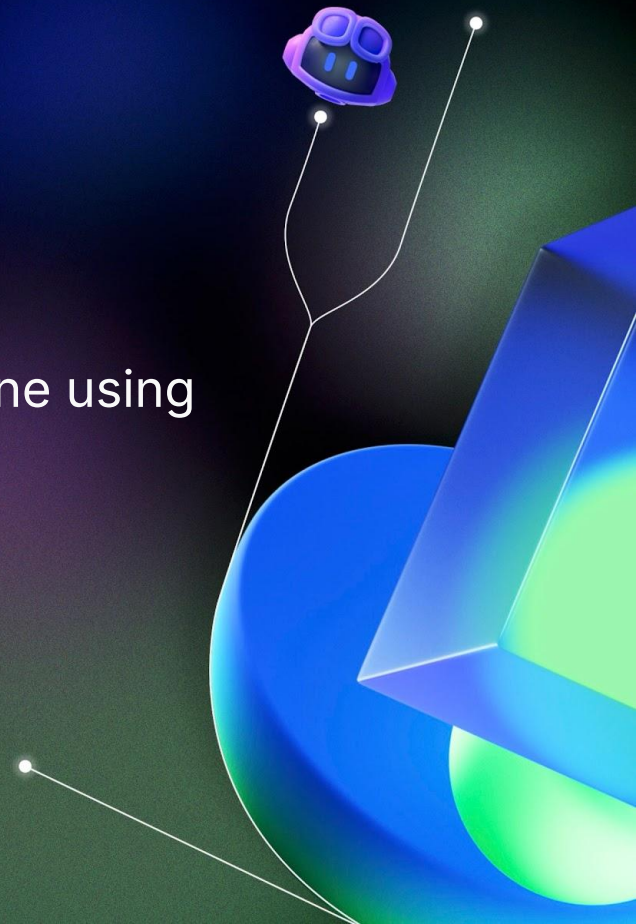
```
import mistune
mistune.html(your_markdown_text)
```

- Available at https://github.com/lepture/mistune

Exercise 3

GitHub Secure Open Source Fund
powered by GitHub Sponsors

The goal is to find a XSS vulnerability in Mistune using Atheris fuzzer



Exercise 3

GitHub Secure Open Source Fund
powered by GitHub Sponsors

In order to run Atheris, we need Python 3.11 (Thanks to Quentin):

```
curl -LsSf https://astral.sh/uv/install.sh | sh
uv tool update-shell

uv python install 3.11
```

Exercise 3

GitHub Secure Open Source Fund
powered by GitHub Sponsors

```
uv venv
source .venv/bin/activate

uv pip install atheris

uv pip install mistune==0.7.4

uv pip install beautifulsoup4 html5lib

python3 fuzz_mistune.py
```

Exercise 3

GitHub Secure Open Source Fund
powered by GitHub Sponsors

```
#155204 REDUCE cov: 209 ft: 1047 corp: 246/13150b lim: 357 exec/s: 2675 rss: 54Mb L: 129/334 MS: 3 ChangeBinInt-ChangeBinInt-EraseBytes-
#155505 REDUCE cov: 209 ft: 1047 corp: 246/13131b lim: 357 exec/s: 2681 rss: 54Mb L: 82/334 MS: 1 EraseBytes-
#155576 REDUCE cov: 209 ft: 1047 corp: 246/13112b lim: 357 exec/s: 2682 rss: 54Mb L: 125/334 MS: 1 EraseBytes-
#155933 REDUCE cov: 209 ft: 1047 corp: 246/13075b lim: 357 exec/s: 2642 rss: 54Mb L: 153/334 MS: 2 InsertByte-EraseBytes-
#156809 REDUCE cov: 209 ft: 1047 corp: 246/13071b lim: 365 exec/s: 2657 rss: 54Mb L: 25/334 MS: 1 EraseBytes-
#156863 REDUCE cov: 209 ft: 1057 corp: 247/13390b lim: 365 exec/s: 2658 rss: 54Mb L: 319/334 MS: 4 ChangeByte-CMP-InsertByte-CopyPart- DE

 === Uncaught Python exception: ===
AssertionError: Unescaped quote in <a href="$ ]#$"[$>"> for payload '[]($)\t\x00#][]($\t]#$"[$>):'
Traceback (most recent call last):
  File "/home/antonio/FuzzingWorkshop/exercise3/mistune/fuzz_mistune.py", line 25, in TestOneInput
    raise AssertionError(
AssertionError: Unescaped quote in <a href="$ ]#$"[$>"> for payload '[]($)\t\x00#][]($\t]#$"[$>):'

==323328== ERROR: libFuzzer: fuzz target exited
SUMMARY: libFuzzer: fuzz target exited
MS: 3 ShuffleBytes-ChangeBinInt-EraseBytes-; base unit: e54ea44c25638e563a5183b02ac98f1c6dd2acd7
0x9,0xdb,0xdd,0x28,0x24,0x29,0x9,0x0,0x23,0xdd,0xdb,0xdd,0x28,0x24,0x9,0xdd,0x23,0x24,0x22,0xdb,0x24,0x3e,0x29,0x3a,
\011\333\335($)\011\000#\335\333\335($\011\335#$\"\333$>):
artifact_prefix='./'; Test unit written to ./crash-eefa1b68470b3f5af236af3465b992549f572d2c
Base64: CdvdKCQpCQAj3dvdKCQJ3SMkItskPik6
```

Exercise 3

GitHub Secure Open Source Fund
powered by GitHub Sponsors

python3 fuzz_mistune.py -seed=3923903506

```
#155204 REDUCE cov: 209 ft: 1047 corp: 246/13150b lim: 357 exec/s: 2675 rss: 54Mb L: 129/334 MS: 3 ChangeBinInt-ChangeBinInt-EraseBytes-
#155505 REDUCE cov: 209 ft: 1047 corp: 246/13131b lim: 357 exec/s: 2681 rss: 54Mb L: 82/334 MS: 1 EraseBytes-
#155576 REDUCE cov: 209 ft: 1047 corp: 246/13112b lim: 357 exec/s: 2682 rss: 54Mb L: 125/334 MS: 1 EraseBytes-
#155933 REDUCE cov: 209 ft: 1047 corp: 246/13075b lim: 357 exec/s: 2642 rss: 54Mb L: 153/334 MS: 2 InsertByte-EraseBytes-
#156809 REDUCE cov: 209 ft: 1047 corp: 246/13071b lim: 365 exec/s: 2657 rss: 54Mb L: 25/334 MS: 1 EraseBytes-
#156863 REDUCE cov: 209 ft: 1057 corp: 247/13390b lim: 365 exec/s: 2658 rss: 54Mb L: 319/334 MS: 4 ChangeByte-CMP-InsertByte-CopyPart- DE

 === Uncaught Python exception: ===
AssertionError: Unescaped quote in <a href="$ ]#$"[$>"> for payload '[]($)\t\x00#][]($\t]#$"[$>):'
Traceback (most recent call last):
  File "/home/antonio/FuzzingWorkshop/exercise3/mistune/fuzz_mistune.py", line 25, in TestOneInput
    raise AssertionError(
AssertionError: Unescaped quote in <a href="$ ]#$"[$>"> for payload '[]($)\t\x00#][]($\t]#$"[$>):'

==323328== ERROR: libFuzzer: fuzz target exited
SUMMARY: libFuzzer: fuzz target exited
MS: 3 ShuffleBytes-ChangeBinInt-EraseBytes-; base unit: e54ea44c25638e563a5183b02ac98f1c6dd2acd7
0x9,0xdb,0xdd,0x28,0x24,0x29,0x9,0x0,0x23,0xdd,0xdb,0xdd,0x28,0x24,0x9,0xdd,0x23,0x24,0x22,0xdb,0x24,0x3e,0x29,0x3a,
\011\333\335($)\011\000#\335\333\335($\011\335#$\"\333$>):
artifact_prefix='./'; Test unit written to ./crash-eefa1b68470b3f5af236af3465b992549f572d2c
Base64: CdvdKCQpCQAj3dvdKCQJ3SMkItskPik6
```

Exercise 3

GitHub Secure Open Source Fund
powered by GitHub Sponsors

Fix bypassing XSS vulnerability.

```
Bypasing with newline.

Old:
>>> markdown('''[aa](java
... script:alert`1`;)''')
'<p><a href="java\nscript:alert`1`;">aa</a></p>\n'

New:
>>> markdown('''[aa](java
... script:alert`1`;)''')
'<p><a href="">aa</a></p>\n'

Bypassing with malicious mail address.

Old:
>>> markdown('<junorouse@gmail.com"\nonclick="alert(1);>')
'<p><a href="mailto:junorouse@gmail.com"\nonclick="alert(1);">junorouse@gmail.com"\nonclick="alert(1);</a></p>\n'

New:
>>> markdown('<junorouse@gmail.com"\nonclick="alert(1);>')
'<p><a href="mailto:junorouse@gmail.com&amp;quot;\nonclick=&amp;quot;alert(1);">junorouse@gmail.com&amp;quot;\nonclick=&amp;quot;alert(1);</a></p>\n'
```

GitHub Secure Open Source Fund
powered by GitHub Sponsors

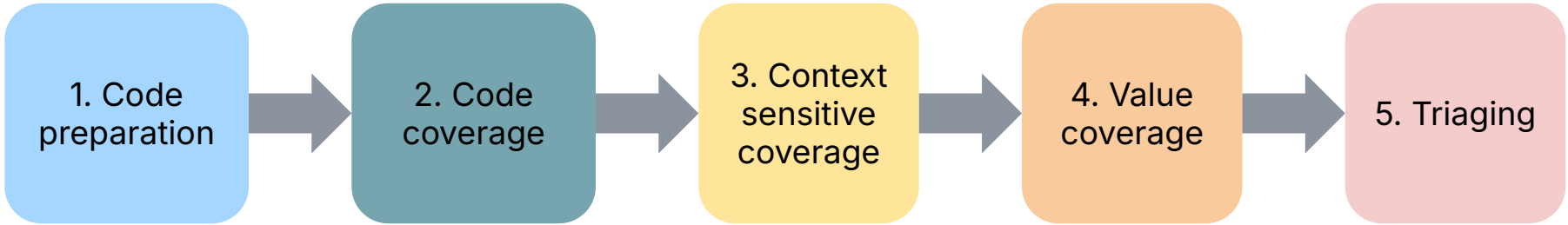
6. More fuzzing

- C/C++: AFL++ fuzzer
  https://github.com/AFLplusplus/AFLplusplus
- Rust: LibFuzzer + cargo-fuzz
  https://github.com/rust-fuzz/cargo-fuzz
- Go:
  - Go <= 1.17: go-fuzz
    https://github.com/dvyukov/go-fuzz
  - Go >= 1.18: native fuzzing support

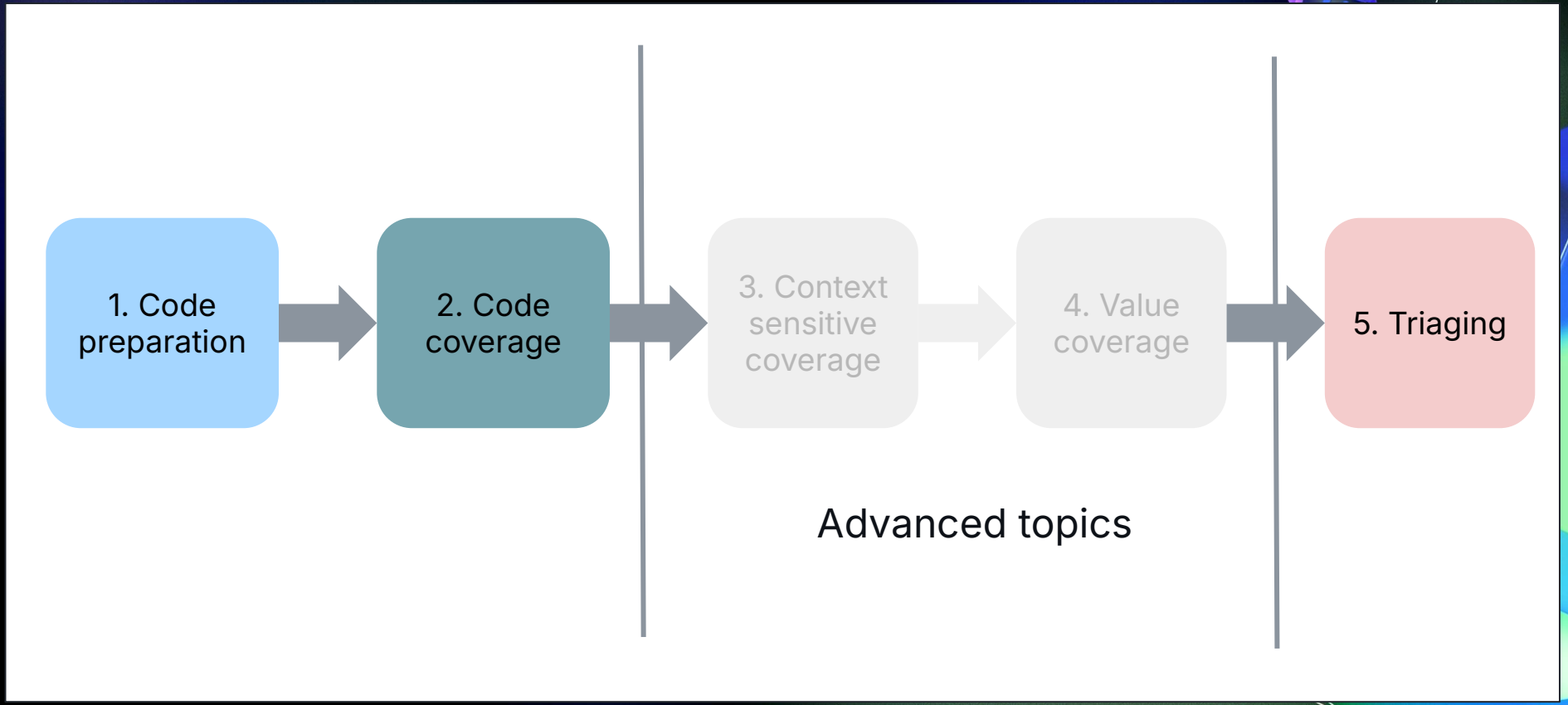
7. The fuzzing workflow

- Fuzzing is an iterative process, similar to any *“agile development methodology”*
- Thinking of fuzzing as a single-iteration process is one of the biggest mistakes most beginners make
- Next, I will show what I believe is the most efficient “fuzzing process” for obtaining good results in fuzzing

7. The fuzzing workflow

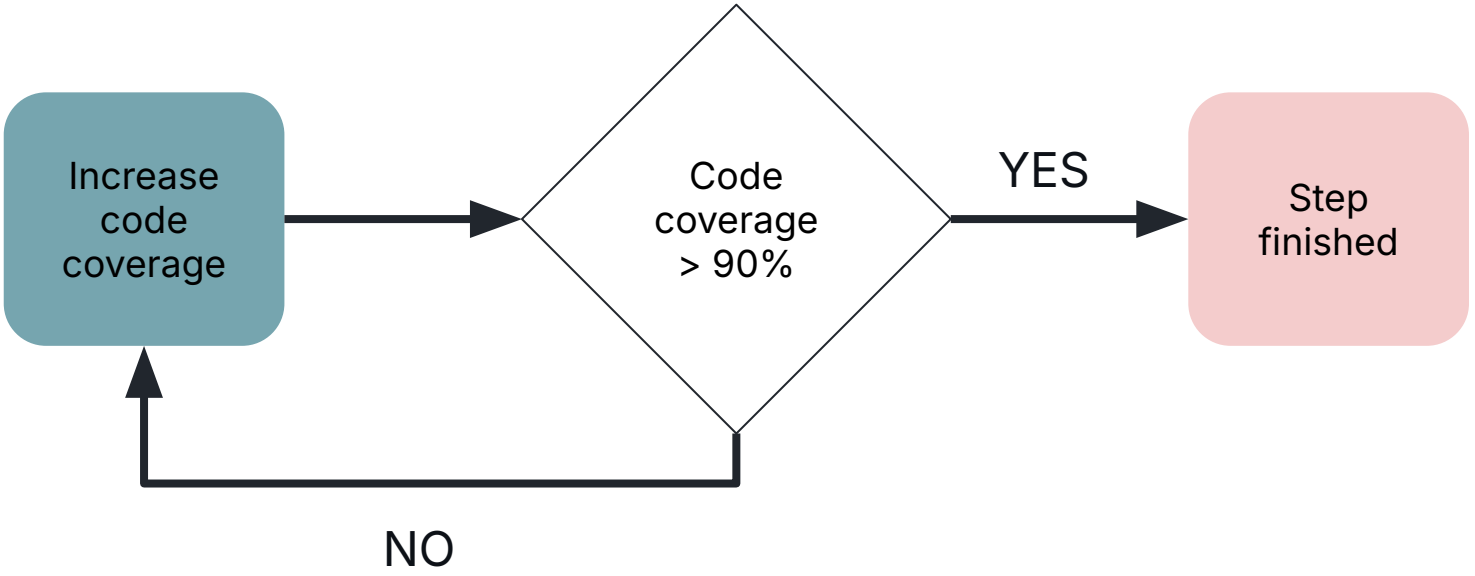


7. The fuzzing workflow



7. The fuzzing workflow

7. The fuzzing workflow



7. The fuzzing workflow

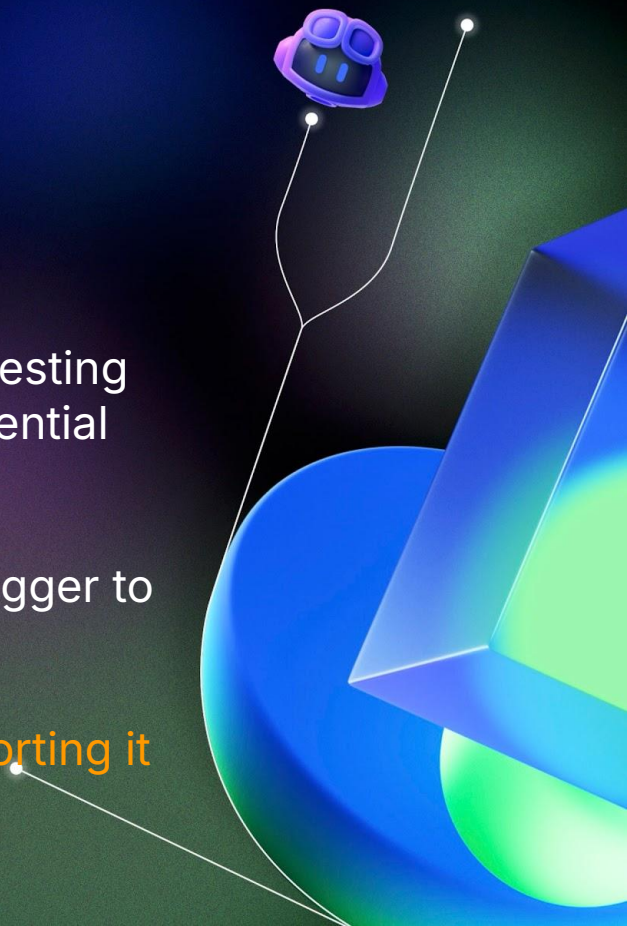
How to increase the code coverage?

- Add new example files
- Use dictionaries
- Use custom mutators [Advanced]
- Use more advanced instrumentation [Advanced]:
  - Compiler transformations (ex. laf-intel)
  - Input-to-State fuzzing (ex. Redqueen)
  - Context-aware fuzzing

7. The fuzzing workflow

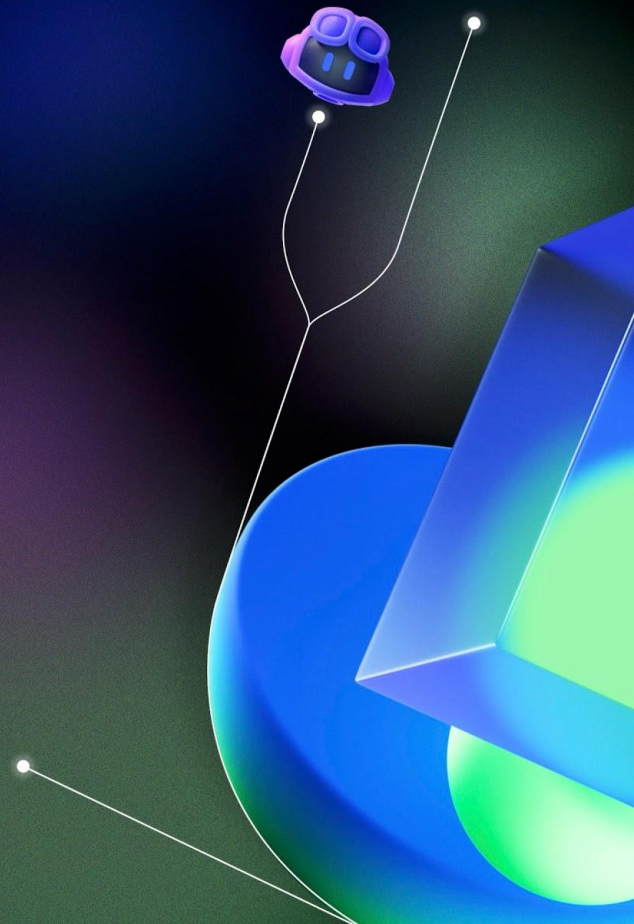
5. Triaging:

- Triaging is the process of categorizing bugs
- It involves analyzing crashes or bugs found during testing to determine their root cause, exploitability, and potential risk
- The most straightforward approach is to use a debugger to examine the crash details
- Once we confirm a vulnerability, the last step is reporting it to developer/s



GitHub Secure
Open Source Fund
powered by GitHub Sponsors

CHALLENGE



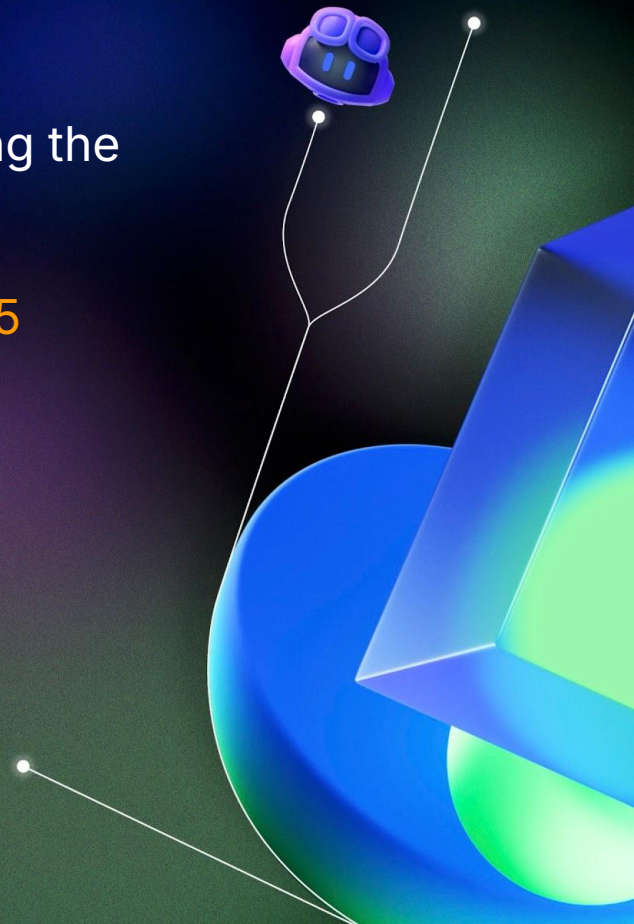
8. Challenge

- The goal is to find a vulnerability in the provided code using Atheris
- The code is a modified version of PyPng: a pure Python library for parsing PNG images
- To win, you must send me:
  - The fuzzer code
  - A proof-of-concept (PoC) file
  - Any additional files used during fuzzing

GitHub Secure Open Source Fund
powered by GitHub Sponsors

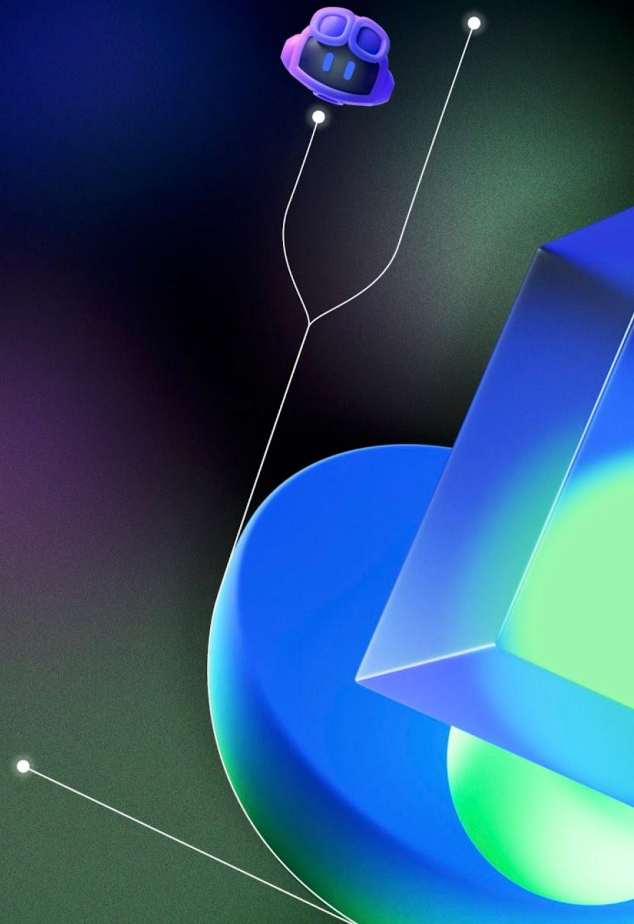
8. Challenge

- I will provide hints and tips both before and during the challenge
- A new hint will be released approximately every 5 minutes
- So, don’t give up and keep trying!
- COPILOT use is allowed



GitHub Secure
Open Source Fund
powered by GitHub Sponsors

Let’s go!!!



8. Challenge

Hint 1

Starting point:

```
$ python3 priinfo ../asset/g229.png
{
  "alpha": false,
  "bitdepth": 8,
  "greyscale": true,
  "planes": 1,
  "size": [
    16,
    16
  ]
}
```

8. Challenge

Hint 2

Provide the fuzzer with an input corpus

“asset” folder

8. Challenge

Hint 3

Write the fuzzer based on **priinfo**

8. Challenge

Hint 4

fuzzer.py

How to run the fuzzer:

```
uv venv
source .venv/bin/activate

uv pip install atheris

python3 fuzzer.py
```

8. Challenge

GitHub Secure Open Source Fund
powered by GitHub Sponsors

Hint 5

Check your code coverage and look for any uncovered code:

```
uv pip install coverage
```

8. Challenge

Hint 6

Measure your coverage:

```
runs=($(ls -1 ../asset | wc -l))

python3 -m coverage run fuzzer.py ../asset/* -atheris_runs="$runs"

python3 -m coverage html
```

Then, you can browse to htmlcov/index.html

8. Challenge

Hint 7

Create a fuzzing dictionary

8. Challenge

Hint 8

python3 fuzzer.py ../asset/ -- -dict=dict.txt

Dictionary example:

"IDAT" "IEND" "pCAL" "IHDR" "PLTE" "CgBI" "bKGD" "cHRM" "fRAc"
"gAMA" "gIFg" "gIFt" "gIFx" "hIST" "iCCP" "iTXt" "oFFs" "IFUZZ"
"pHYs"

Next Steps for Learning

GitHub Secure Open Source Fund
powered by GitHub Sponsors

Fuzzing101
https://github.com/antonio-morales/Fuzzing101

A GitHub Security Lab initiative for learning fuzzing and finding vulnerabilities in real software

Todo el material que he mostrado en este workshop pasará a formar parte del repositorio

Exercise No.	Target	CVEs to find	Time estimated	Main topics
Exercise 1	Xpdf	CVE-2019-13288	120 mins	Afl-clang-fast, Afl-fuzz, GDB
Exercise 2	libexif	CVE-2009-3895, CVE-2012-2836	6 hours	Afl-clang-lto, Fuzz libraries, Eclipse IDE
Exercise 3	TCPdump	CVE-2017-13028	4 hours	ASan, Sanitizers
Exercise 4	LibTIFF	CVE-2016-9297	3 hours	Code coverage, LCOV
Exercise 5	Libxml2	CVE-2017-9048	3 hours	Dictionaries, Basic parallelization, Fuzzing command-line arguments
Exercise 6	GIMP	CVE-2016-4994, Bonus bugs	7 hours	Persistent fuzzing, Fuzzing interactive applications
Exercise 7	VLC media player	CVE-2019-14776	6 hours	Partial instrumentation, Fuzzing harness
Exercise 8	Adobe Reader		8 hours	Fuzzing closed-source applications, QEMU instrumentation
Exercise 9	7-Zip	CVE-2016-2334	8 hours	WinAFL, Fuzzing Windows Applications
Exercise 10 (Final Challenge)	**Google Chrome / V8**	**CVE-2019-5847**	**8 hours**	**Fuzzilli, Fuzzing Javascript engines**

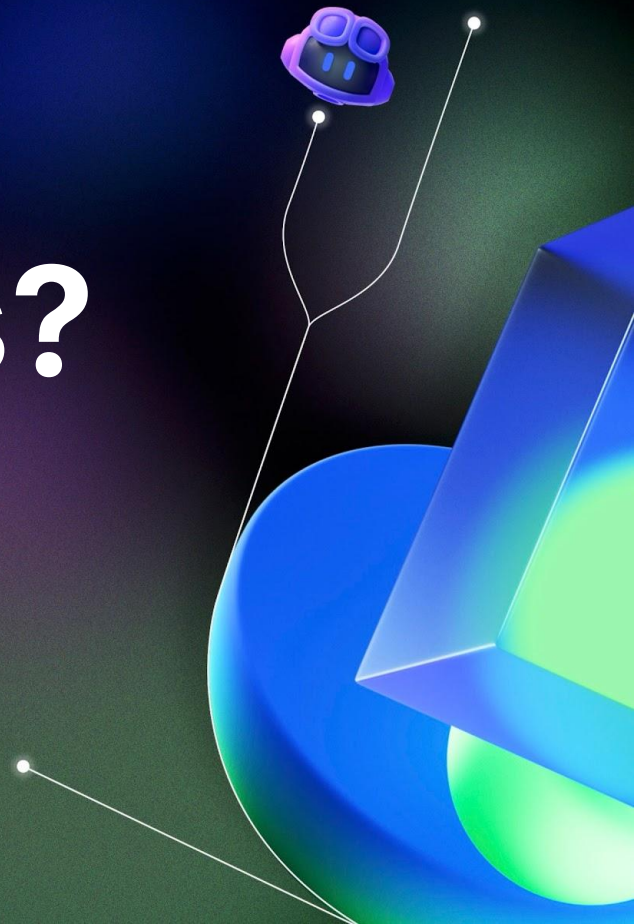
GitHub Secure
Open Source Fund
powered by GitHub Sponsors

10. Homework

How many of you have a ‘fuzzer’ folder in your GitHub repository?

GitHub Secure
Open Source Fund
powered by GitHub Sponsors

Questions?



GitHub Secure
Open Source Fund
powered by GitHub Sponsors