

Fuzzers CLI + GUI Suite

Modern Security Testing with Integrated UI/UX

Hardish Singh



Project Overview: Unifying Fuzzing

What is Fuzzing?

Fuzzing is an automated software testing technique that involves providing invalid, unexpected, or random data as inputs to a computer program. The goal is to discover software bugs, security vulnerabilities, or crashes by monitoring the program's behavior.

Why Combine CLI + GUI?

A unified interface offers the best of both worlds: the power and scripting capabilities of CLI for advanced users, and the ease-of-use and visual feedback of GUI for beginners. This integration streamlines workflows, enhances accessibility, and accelerates vulnerability discovery.

Benefits for Bug Bounty Hunters & Pentesters

- Accelerated vulnerability discovery
- Streamlined workflow for diverse fuzzers
- Improved collaboration and reporting
- Reduced learning curve for new tools

Architecture Diagram: Next.js ⇄ Node.js ⇄ Fuzzers

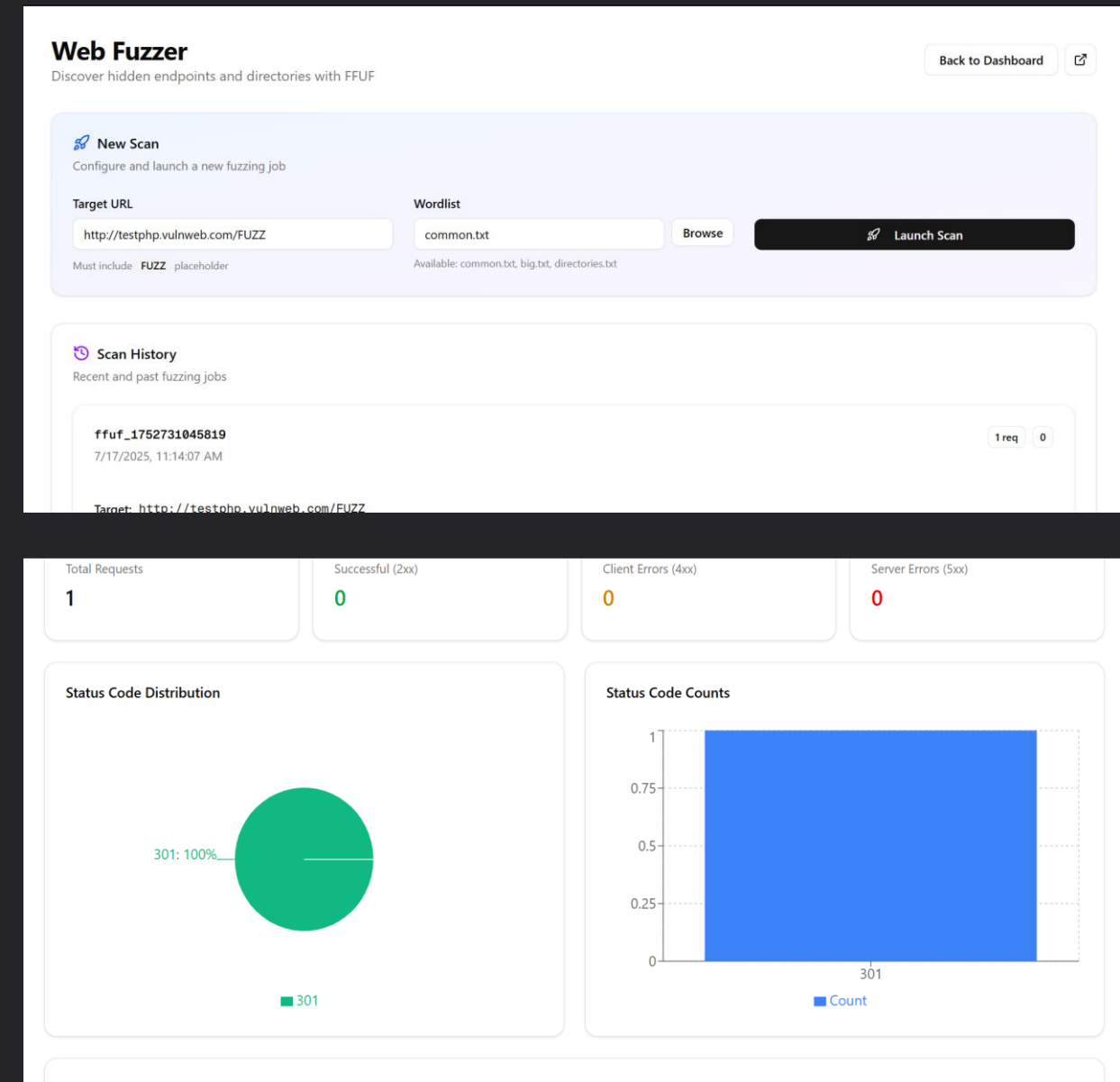
FFUF Integration: Fast Web Fuzzing

FFUF (Fuzz Faster U Fool) is a fast web fuzzer often used for directory and file enumeration, virtual host discovery, and GET/POST parameter fuzzing. Its speed and versatility make it a staple in web application security testing.

CLI Sample:

```
ffuf -u https://target/FUZZ -w wordlist.txt
```

The GUI integrates FFUF's core functionalities, offering intuitive controls for target input, wordlist selection, and HTTP response analysis. Users can easily visualize findings and filter results based on status codes or content length.



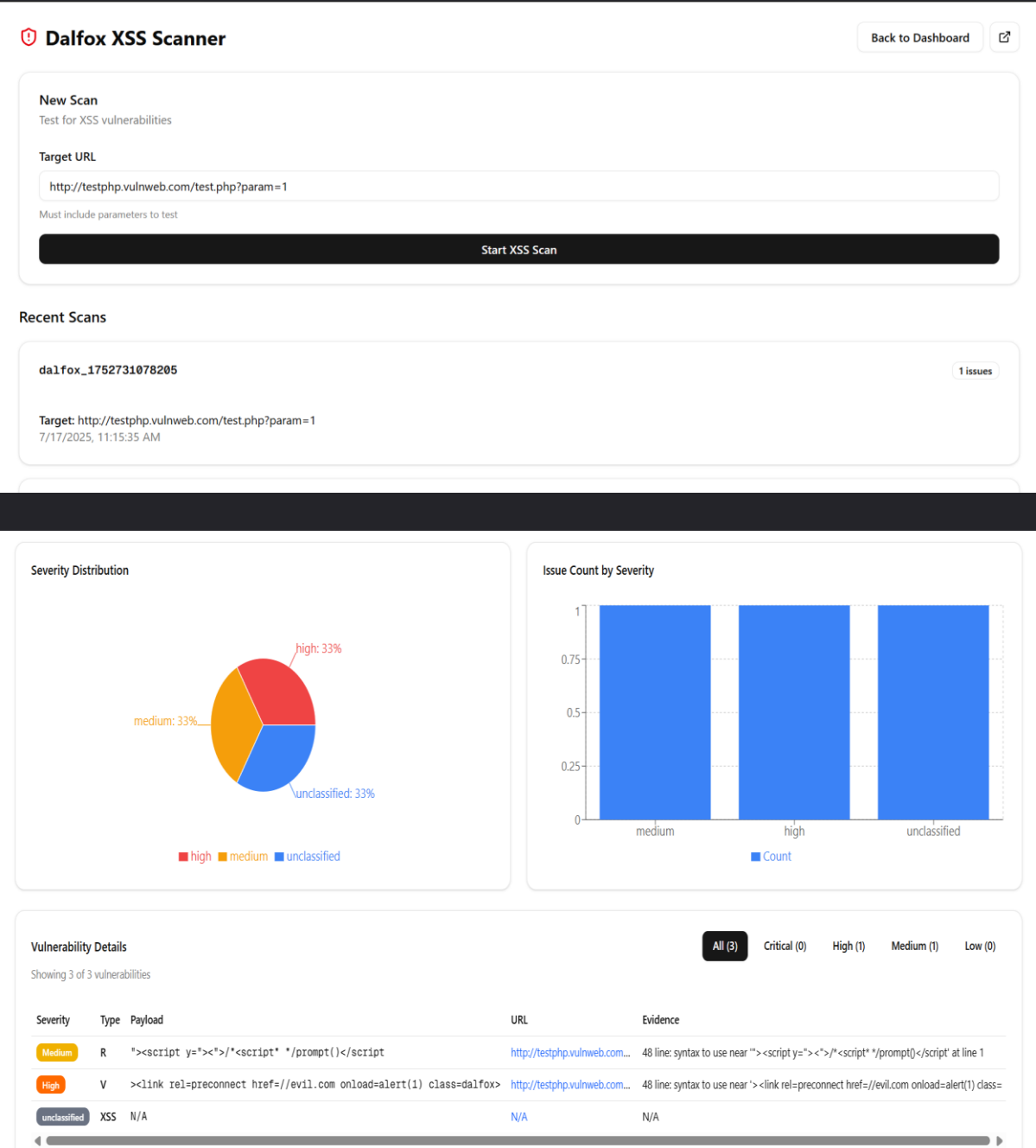
Dalfox Integration: XSS Scanning

Dalfox is a powerful tool designed for detecting Cross-Site Scripting (XSS) vulnerabilities. It supports various payload injection points and provides detailed reports, making it an essential asset for web penetration testers.

CLI Sample:

```
dalfox url https://target
```

Our GUI provides a clear overview of Dalfox scan results, allowing for easy filtering and sorting of identified vulnerabilities. Interactive elements enable users to drill down into specific findings and analyze the impact.



WFuzz Integration: Advanced Fuzzing

WFuzz is a highly flexible and powerful web application fuzzer. It offers extensive customization options, including payload encoding, different fuzzer types, and filters, enabling deep and precise vulnerability hunting.

CLI Sample:

```
wfuzz -c -z file,wordlist.txt -u https://target/FUZZ
```

The GUI for WFuzz simplifies complex configurations, providing a visual builder for fuzzing parameters and real-time display of response times. This helps in identifying slow responses or potential denial-of-service vulnerabilities efficiently.

Back to Dashboard

WFuzz Scanner

Target URL

http://testphp.vulnweb.com/artists.php?FUZZ=1

Must include FUZZ placeholder

Wordlist

common.txt

File must exist in wordlists directory

Start Scan

Recent Scans

wfuzz_1752731193846

10 requests 10 2xx 0 3xx 0 4xx 0 5xx

Target: http://testphp.vulnweb.com/artists.php?FUZZ=1
7/17/2025, 11:16:55 AM

wfuzz_1752405442785

10 requests 10 2xx 0 3xx 0 4xx 0 5xx

Target: http://testphp.vulnweb.com/artists.php?FUZZ=1
7/13/2025, 4:47:30 PM

wfuzz_1752404687568

10 requests 10 2xx 0 3xx 0 4xx 0 5xx

Target: http://testphp.vulnweb.com/artists.php?FUZZ=1

Back to Scans

Scan Results

Job ID: wfuzz_1752731193846

Total Requests

10

Successful (2xx)

10

Client Errors (4xx)

0

Server Errors (5xx)

0

All Results (10) Successful (10) Errors (0)

Request Details

Showing 10 of 10 requests

ID	Status	Payload	Response Size	URL
00003	200	test	5328 chars / 104 lines	http://testphp.vulnweb.com/artists.php?test=1
00001	200	admin	5328 chars / 104 lines	http://testphp.vulnweb.com/artists.php?admin=1
00010	200	config	5328 chars / 104 lines	http://testphp.vulnweb.com/artists.php?config=1
00008	200	portal	5328 chars / 104 lines	http://testphp.vulnweb.com/artists.php?portal=1
00009	200	console	5328 chars / 104 lines	http://testphp.vulnweb.com/artists.php?console=1
00006	200	backend	5328 chars / 104 lines	http://testphp.vulnweb.com/artists.php?backend=1
00002	200	login	5328 chars / 104 lines	http://testphp.vulnweb.com/artists.php?login=1
00004	200	api	5328 chars / 104 lines	http://testphp.vulnweb.com/artists.php?api=1
00005	200	user	5328 chars / 104 lines	http://testphp.vulnweb.com/artists.php?user=1

Made with GAMMA

Radamsa Integration: Smart Fuzzing

Radamsa is a general-purpose fuzzer that generates test cases based on a given input. It's known for its ability to produce highly effective and "interesting" inputs, making it valuable for discovering unexpected edge cases and crashes in various applications.



CLI Sample:

```
echo "input" | radamsa
```

The Radamsa GUI provides an interactive interface for input mutation, allowing users to preview fuzzed samples in real-time. This visual feedback helps in understanding how inputs are transformed and refining the fuzzing strategy.

Radamsa Fuzzer

Back to Dashboard

New Fuzzing Job

Upload a file to generate fuzzed variants

Input File

Choose File No file chosen

Iterations

10

Seed (optional)

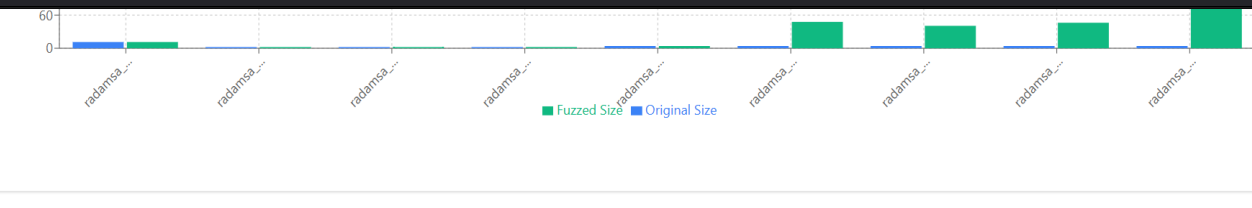
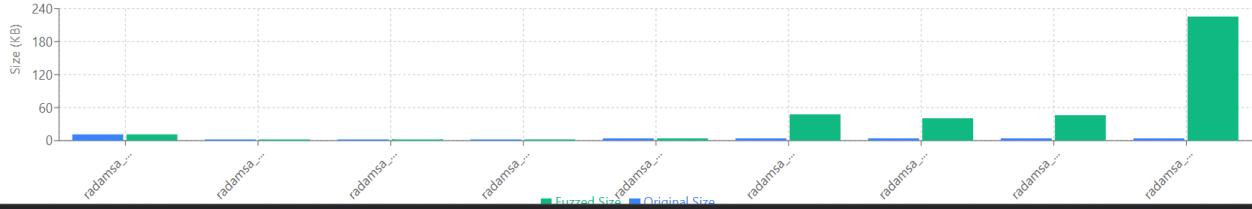
Random seed for reproducibility

Start Fuzzing

Fuzzing Statistics

File Size Comparison

Original vs Fuzzed File Sizes



Recent Jobs

Job ID	Original File	Fuzzed File	Size Change	Iterations	Date	Actions
radamsa_...	needto fuzz.jpeg	11.31 KB	0.04%	10	7/17/2025, 1:07:33 PM	<div>Download</div> <div>Download All (10)</div>
radamsa_...	test.json	2.13 KB	0.37%	10	7/17/2025, 12:30:12 PM	<div>Download</div> <div>Download All (10)</div>
radamsa_...	test.json	2.12 KB	-0.28%	10	7/17/2025, 12:25:56 PM	<div>Download</div> <div>Download All (10)</div>
radamsa_...	test.json	2.19 KB	3.25%	10	7/17/2025, 11:52:01 AM	<div>Download</div> <div>Download All (10)</div>
radamsa_...	picture.jpeg	4.04 KB	-1.95%	10	7/17/2025, 11:50:57 AM	<div>Download</div> <div>Download All (10)</div>
radamsa_...	picture.jpeg	47.8 KB	1060.66%	10	7/13/2025, 6:50:19 PM	<div>Download</div>
radamsa_...	picture.jpeg	40.66 KB	887.22%	10	7/13/2025, 6:48:06 PM	<div>Download</div>
radamsa_...	picture.jpeg	46.25 KB	1022.98%	10	7/13/2025, 6:47:52 PM	<div>Download</div>
radamsa_...	picture.jpeg	225.68 KB	5380.03%	10	7/13/2025, 6:44:21 PM	<div>Download</div>

Made with GAMMA

ZFuzz Integration: API Fuzzing

ZFuzz is tailored for efficient API fuzzing, focusing on web APIs and their parameters. It can quickly uncover vulnerabilities like injection flaws, improper authentication, or data exposure by systematically testing API endpoints.

CLI Sample:

```
zfuzz -u https://target --wordlist wordlist.txt
```

The ZFuzz GUI features a live results stream, providing immediate feedback on API responses and identified issues. This real-time monitoring allows security professionals to react quickly to discovered vulnerabilities and adjust their testing strategy on the fly.

zzuf Binary Fuzzer

Back to Dashboard

New Job

Job History

Analytics

New Fuzzing Job

Fuzz binary files with zzuf

Input File

Choose File No file chosen

Mutation Ratio (0-1)

Seed (optional)

Timeout (seconds)

0.4

Random seed for reproducibility

30

Start Fuzzing

zzuf Binary Fuzzer

Back to Dashboard

New Job

Job History

Analytics

Recent Jobs

Previous fuzzing jobs

Job ID	File	Ratio	Size Change	Time	Status	Actions
zzuf_1752732265945	test.json	0.4	0.00%	35ms	completed	Details
zzuf_1752732231768	picture.jpeg	0.4	0.00%	68ms	completed	Details
zzuf_1752731436759	sample_binary_file.bin	0.4	0.00%	58ms	completed	Details
zzuf_1752415981475	sample_binary_file.bin	0.4	0.00%	101ms	completed	Details

zzuf Binary Fuzzer

Back to Dashboard

New Job

Job History

Analytics

Fuzzing Analytics

Performance metrics and trends

Job ID	Original Size (KB)	Fuzzed Size (KB)	Mutation Ratio (%)
zzuf_1752732265945	2	2	40
zzuf_1752732231768	4	2	40
zzuf_1752731436759	5	2	40
zzuf_1752415981475	5	2	40

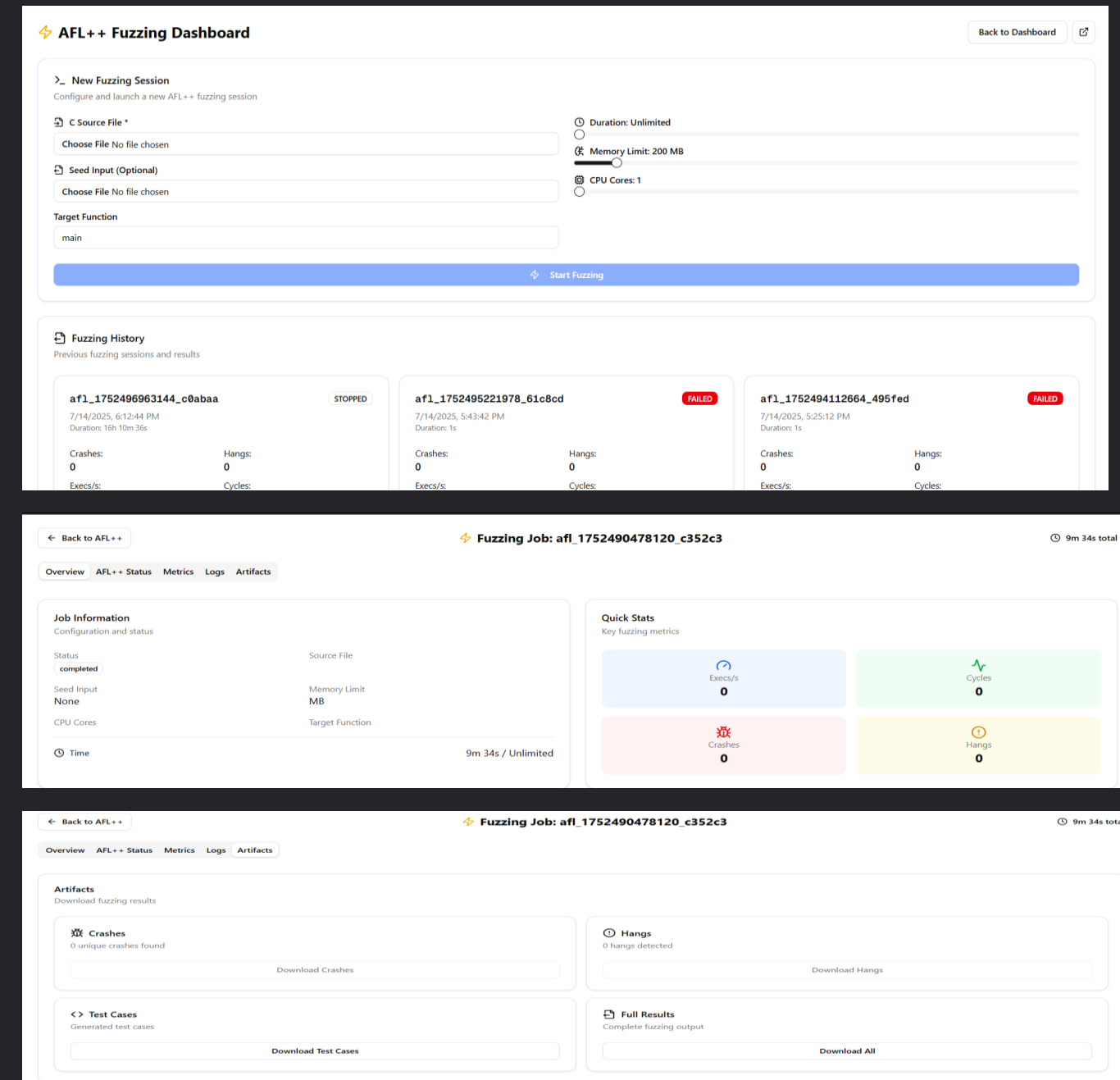
AFL++ Integration: Advanced Coverage Fuzzing

AFL++ (American Fuzzy Lop Plus Plus) is a highly optimized, coverage-guided fuzzer known for its effectiveness in finding deep-seated bugs and vulnerabilities in complex binaries. It uses advanced techniques to explore program execution paths.

CLI Sample:

```
afl-fuzz -i inputs -o outputs -- ./target
```

Our GUI for AFL++ is not ready yet I am working on it . One's ready This visual representation helps users track fuzzing progress, identify bottlenecks, and understand the code coverage achieved, optimizing the fuzzing campaign .



Project Tech Stack

Our Fuzzers CLI + GUI Suite is built on a robust and modern technology stack designed for performance, scalability, and an intuitive user experience.



Next.js

Frontend framework for interactive and responsive UI/UX.



Node.js

Backend for CLI commands and fuzzer orchestration.



Firebase

Database for configuration persistence and CLI management.

This combination ensures a seamless blend of powerful backend operations with a modern, dynamic frontend, providing a superior fuzzing experience.

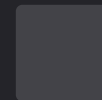
Conclusion & Future Plans

The Fuzzers CLI + GUI Suite provides a unified, efficient, and user-friendly platform for cybersecurity professionals. By combining the power of various fuzzers with a modern interface .



Summary of Features

- Integrated CLI + GUI for flexibility
- Modern tech stack (Next.js, Node.js, Firebase)
- Enhanced usability and workflow



Future Plans

- Real-time logging and advanced analytics
- Custom plugin architecture for extensibility

Follow me on GitHub if you feel that something more needed in this let's contribute to the future of integrated security testing!

GitHub Repository:

github.com/Hardish-singh

Contact:

hardishrajpu35@gmail.com



THANK YOU!