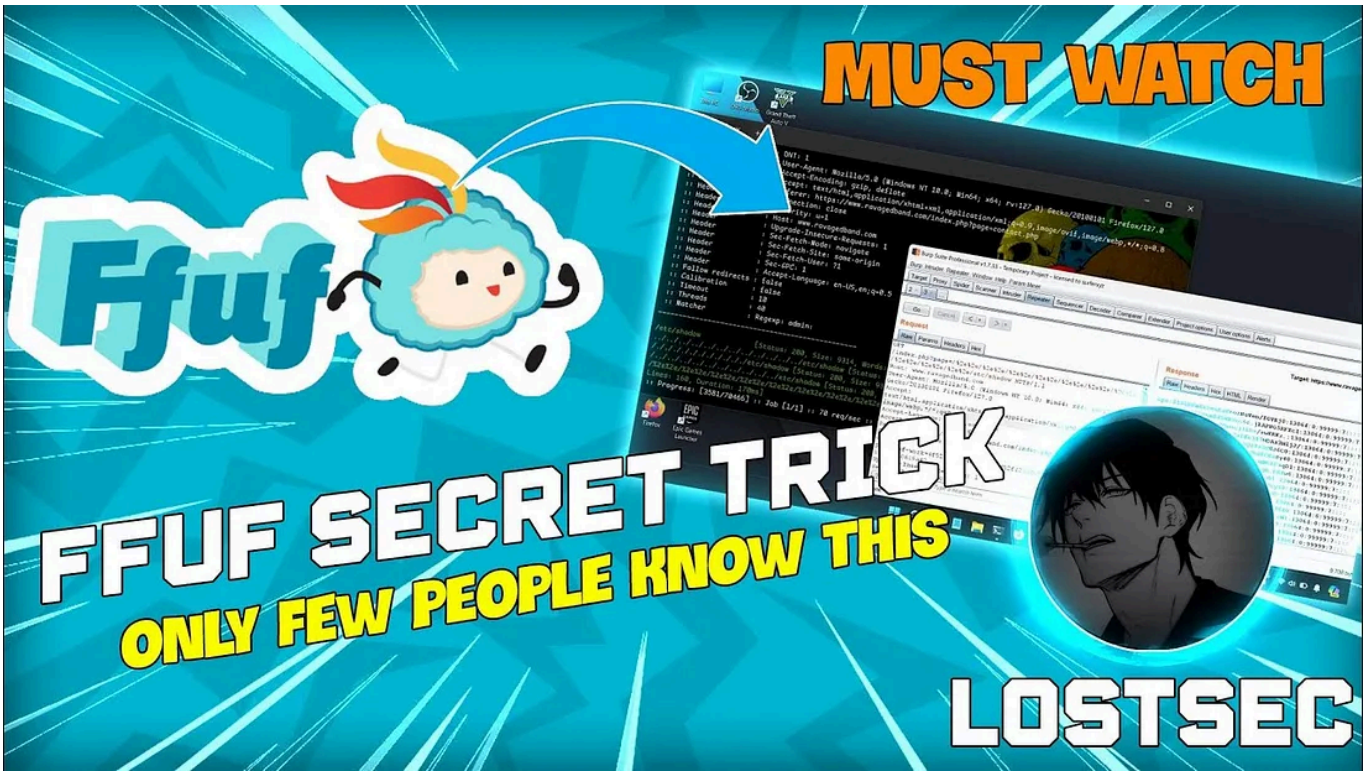
OSINT Team

Member-only story

FFUF Mastery: The Ultimate Web Fuzzing Guide

A Complete Guide to master FFUF tool with Advance methods

coffinxp Follow 6 min read · Feb 3, 2025



Introduction

FUF is a powerful open source fuzzing tool used for web application security testing. It allows users to discover hidden files, directories, subdomains and parameters through its high speed fuzzing. This article will break down FFUF commands and explain how to use them effectively.

> *FFUF Installation*

To install FFUF on your system use the following command:

```
apt install ffuf
```

This will install FFUF on Debian based systems. For other operating systems you can download the binary from the official GitHub repository from given

link

GitHub - ffuf: Fast web fuzzer written in Go

Fast web fuzzer written in Go. Contribute to ffuf development by creating an account on GitHub.

github.com

Basic FFUF Commands

Directory and File Brute force

One of the most common uses of FFUF is finding hidden directories and files on a web server. You can do this by using the -u flag to set the target URL and the -w flag to provide a wordlist

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

Here FUZZ acts as a placeholder that FFUF replaces with words from the wordlist to brute force directories or files

POST Request with Wordlist

```
ffuf -w wordlist.txt -u https://website.com/FUZZ -X POST
```

This command will fuzz the given url using each word in wordlist.txt while sending POST requests aiming to find hidden directories or endpoints

Case Insensitive Matching

Use -ic flag for a case insensitive search especially when you are unsure about the server case sensitivity. This means that it doesn't matter whether the characters in the search term are uppercase or lowercase both will be treated the same during the fuzzing process. You can also add the -c flag for colorful output

```
ffuf -u https://example.com/FUZZ -w wordlist.txt -ic -c
```

File Extension Fuzzing

To search for files with specific extensions use the -e flag to list the extensions you want to check

```
ffuf -u https://example.com/indexFUZZ -w wordlist.txt -e .php,.asp,.bak,.db
```

This command tests different extensions like .php, .asp, .bak and .db by adding them to each word in the wordlist. It helps to find hidden files with common extensions

Recursive Fuzzing

For fuzzing multiple levels of directories use the -recursion flag and specify the recursion depth using -recursion-depth.

```
ffuf -u https://example.com/FUZZ -w wordlist.txt -recursion -recursion-depth 3
```

This will scan up to three directory levels deep useful for finding more hidden directories

Filtering Responses

FFUF allows you to filter responses based on specific criteria like HTTP status codes or response sizes you can use that by following command

```
ffuf -w wordlist.txt -u https://example.com/FUZZ -fc 404,500
```

This command excludes responses with status codes 404 or 500 or your given one

Multi Wordlist Fuzzing

```
ffuf -u https://example.com/W2/W1/ -w dict.txt:W1 -w dns_dict.txt:W2
```

This command fuzzes two parameters (W1 and W2) using separate two wordlists.

Subdomain and Virtual Host Fuzzing

Subdomain Fuzzing

A key part of web security testing is finding hidden subdomains. FFUF can brute force subdomains by replacing the FUZZ keyword in the target URL

```
ffuf -w subdomains.txt -u https://FUZZ.example.com/
```

This command finds subdomains by fuzzing subdomains from subdomains.txt and replacing FUZZ with each one. It helps identify potential subdomains within a target domain

Virtual Host (VHost) Fuzzing

To detect virtual hosts use the -H flag to fuzz the Host header.

```
ffuf -w vhosts.txt -u https://example.com/ -H "Host: FUZZ.example.com"
```

This will search for virtual hosts like admin.example.com, test.example.com, and other subdomains

Fuzzing HTTP Parameters

Fuzzing GET Parameters

To find potential GET parameters fuzz the query string in the URL

```
ffuf -w wordlist.txt -u https://example.com/page.php?FUZZ=value
```

This command replaces FUZZ with words from the wordlist to test different GET parameters

Fuzzing POST Parameters

For POST requests use the -X POST flag to specify the HTTP method and fuzz the POST data.

```
ffuf -w wordlist.txt -u https://example.com/api -X POST -d 'FUZZ=value'
```

This is especially useful for testing APIs or login forms.

POST Request Fuzzing (Login Bypass)

```
t.txt -X POST -d "username=admin&password=FUZZ" -u https://www.example.com/login
```

FFUF sends POST requests to given url fuzzing the password parameter by replacing FUZZ with each entry from passwordlist.txt. This is useful for brute forcing login systems

PUT Request Fuzzing

```
f -w /path/to/wordlist.txt -X PUT -u https://target.com/FUZZ -b 'session=abcdef'
```

This fuzzes with HTTP PUT requests using a session cookie (session=abcdef). Its useful for testing unauthorized file uploads or modifications

Advanced FFUF Method

Clusterbomb Mode

```
xt -request-proto http -mode clusterbomb -w usernames.txt:HFUZZ -w passwords.txt:WFU
```

In clusterbomb mode FFUF uses two wordlists: usernames.txt for HFUZZ and passwords.txt for WFUZZ. It combines each username with every password to test login attempts using the custom request structure from req.txt

```
:PASS -u https://example.com/login?username=USER&password=PASS -mode clusterbomb
```

In clusterbomb mode FFUF tests combinations of usernames from users.txt and passwords from passwords.txt by replacing USER and PASS in the URL

Pitchfork Mode

```
ers.txt:USER -w passwords.txt:PASS -u https://example.com/login?username=USER&passwo
```

In pitchfork mode FFUF takes each word from one list (e.g., usernames) and pairs it with the corresponding entry from another list (e.g., passwords). This mode offers a more controlled brute force test compared to clusterbomb

To include cookies in your requests use the -b flag.

```
ffuf -b "SESSIONID=abcd1234; USER=admin" -w wordlist.txt -u https://example.com/FUZZ
```

This is useful for authenticated fuzzing scenarios.

Using Proxies

You can route FFUF requests through a proxy like Burp Suite for deeper analysis

```
ffuf -x http://127.0.0.1:8080 -w wordlist.txt -u https://example.com/FUZZ
```

Custom Header Fuzzing

```
ffuf -w headers.txt -u https://example.com/ -H "X-Custom-Header: FUZZ"
```

This fuzzes the X-Custom-Header by replacing FUZZ with values from headers.txt helping find issues with custom HTTP headers

Fuzzing with Custom User-Agent

Use the -H flag to modify custom user-agent headers in your requests

```
txt -H "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KH
```

This is especially useful for bypassing restrictions or mimic specific browser behavior.

Rate Limiting Bypass

To avoid overwhelming the target server you can control the request rate using the -rate and -t flag

```
ffuf -w wordlist.txt -u https://example.com/FUZZ -rate 50 -t 50
```

This limits requests to 50 per second and threads to 50 helping prevent overwhelming the target or triggering rate limiting defenses

Output Options

FFUF provides several output formats including HTML, JSON, and CSV for saving the results

```
ffuf -w wordlist.txt -u https://example.com/FUZZ -o results.html -of html
```

This saves the results in an HTML file for easy analysis. You can also use the -of -all flag to save all output files at once

Custom wordlist

A wordlist is important for directory brute forcing. A better wordlist increases your chances of finding valuable files and directories. You can access my custom wordlist on my GitHub repository to improve your testing. it contains all types of wordlist that are useful in bug hunting you can access it from given link

You can also use the SecLists wordlist which contains a variety of wordlists useful for web fuzzing and other system testing. You can download it from the following link

GitHub - danielmiessler/SecLists: SecLists is the security tester's companion. It's a collection of...

SecLists is the security tester's companion. It's a collection of multiple types of lists used during security...

github.com

Conclusion

FFUF is a powerful tool for web security testing with many customization options. Learning its commands helps you find hidden vulnerabilities and secure your applications. Whether you are experienced or just starting mastering it will improve your penetration testing skills

Disclaimer

> The content provided in this article is for educational and informational purposes only. Always ensure you have proper authorization before conducting security assessments. Use this information responsibly