ffuf.

Presentation by:

- Ben ™
- Marty D
- Nick W

This presentation is on the Fuzzing tool called ffuf

Presented by myself Nick, Ben and Marty

Table of contents

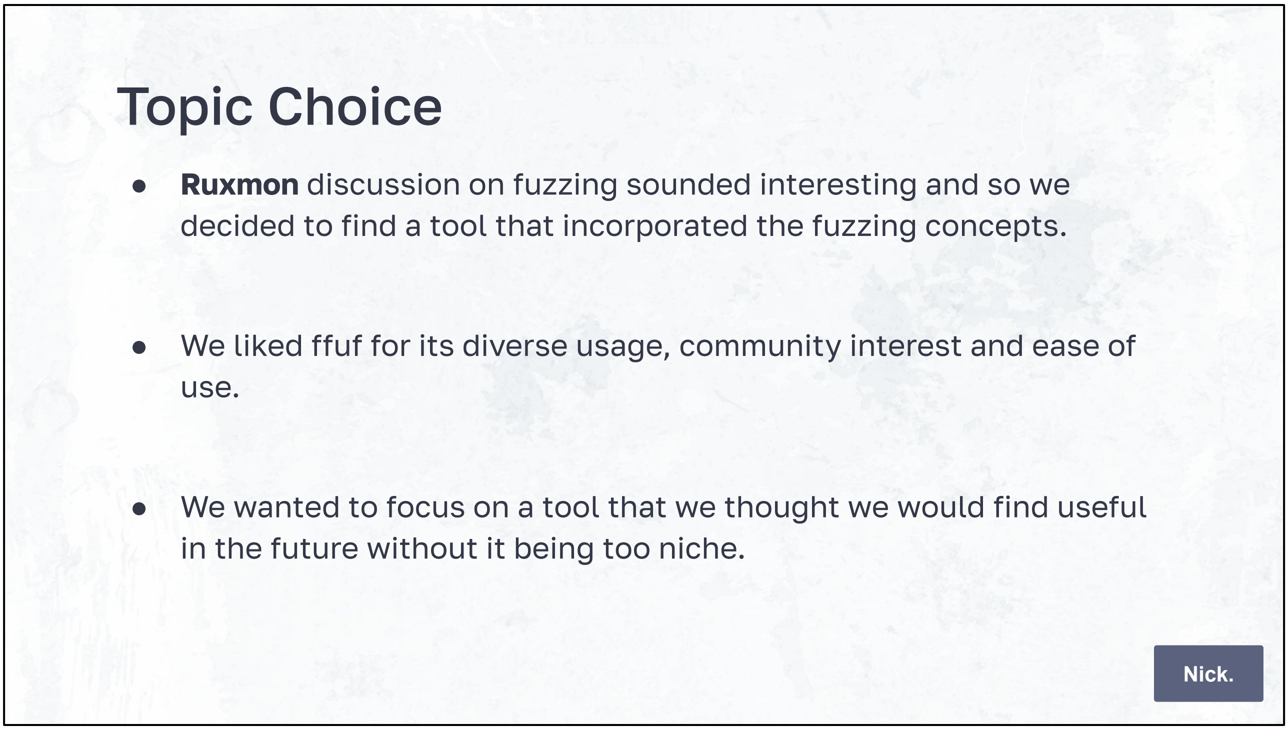
01 What is ffuf?

02 Demonstration

03 Mitigation

04 Appendix

Nick.



Topic Choice

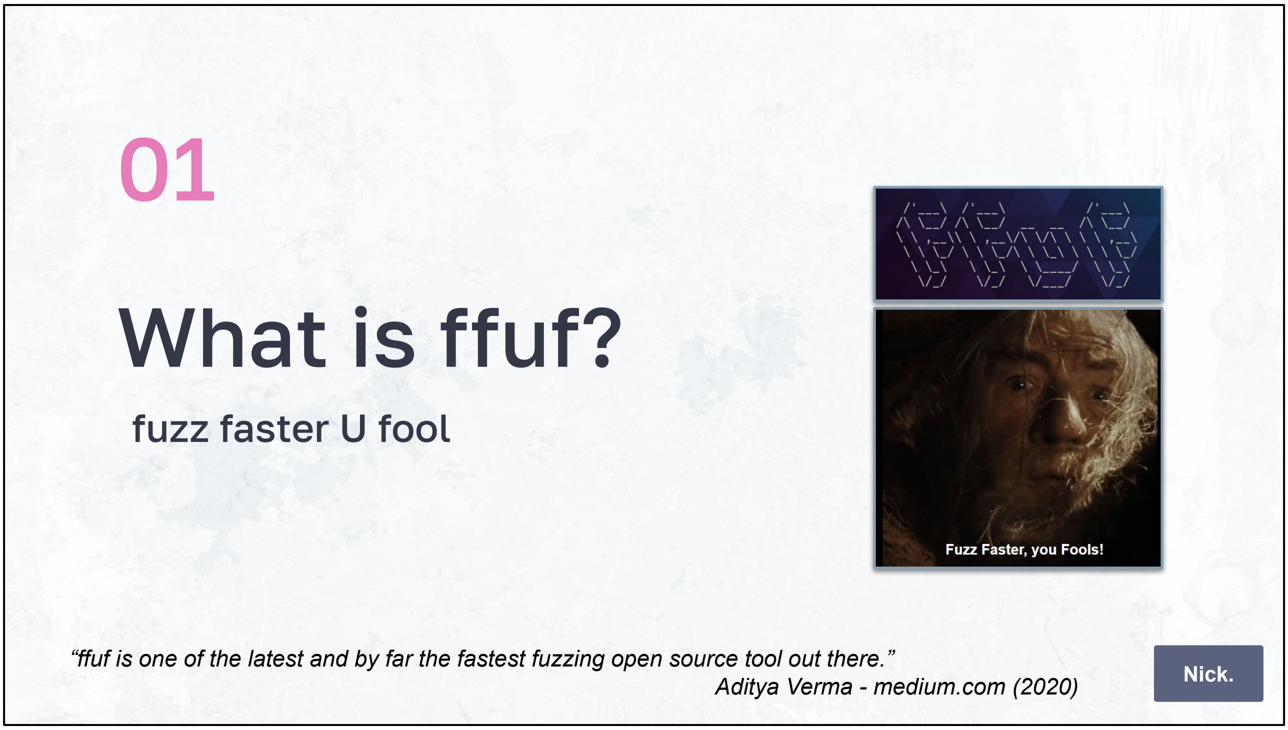
- **Ruxmon** discussion on fuzzing sounded interesting and so we decided to find a tool that incorporated the fuzzing concepts.

- We liked ffuf for its diverse usage, community interest and ease of use.

- We wanted to focus on a tool that we thought we would find useful in the future without it being too niche.

Nick.

Practical tool
Useful for many security assessments for both red or blue teams.



01

What is ffuf?

fuzz faster U fool

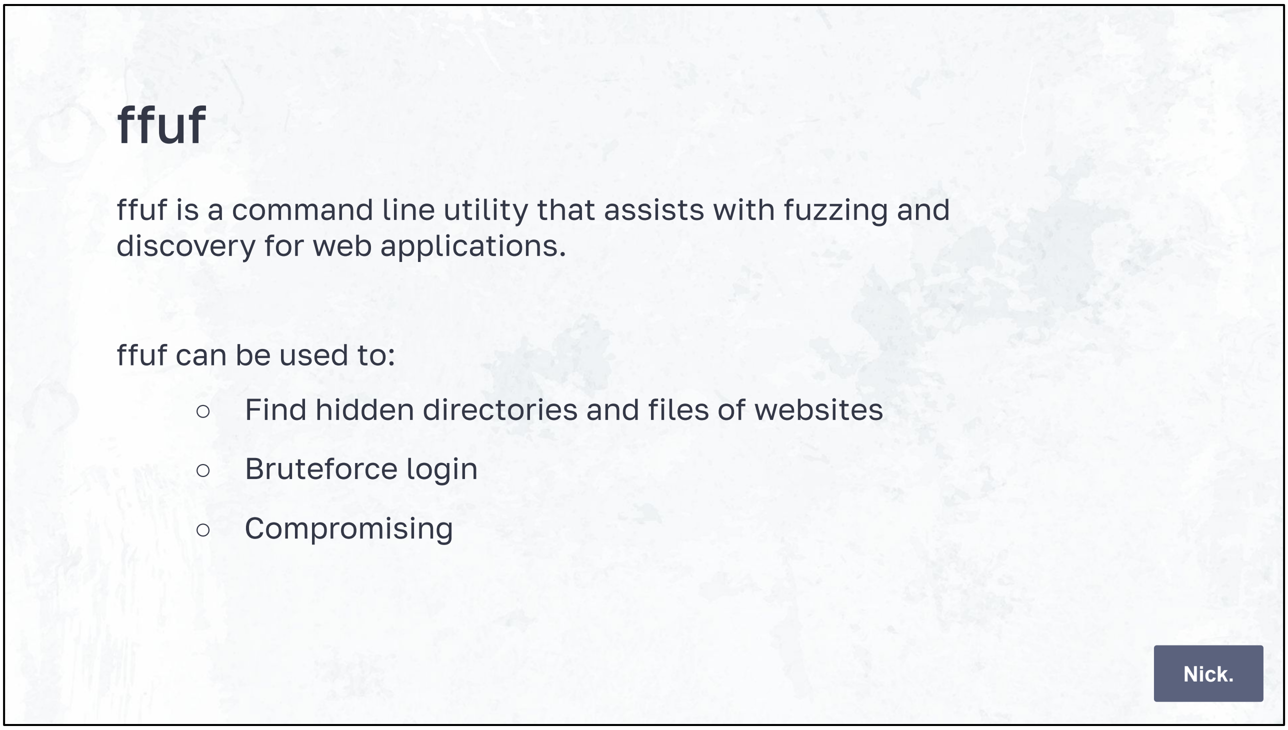
“ffuf is one of the latest and by far the fastest fuzzing open source tool out there.”
Aditya Verma - medium.com (2020)

Nick.

ffuf is a command line utility that assists with fuzzing and discovery for web applications.

ffuf can be used to:

- ○ Find hidden directories and files of websites
- ○ Bruteforce login
- ○ Compromising



ffuf

ffuf is a command line utility that assists with fuzzing and discovery for web applications.

ffuf can be used to:

- ○ Find hidden directories and files of websites
- ○ Bruteforce login
- ○ Compromising

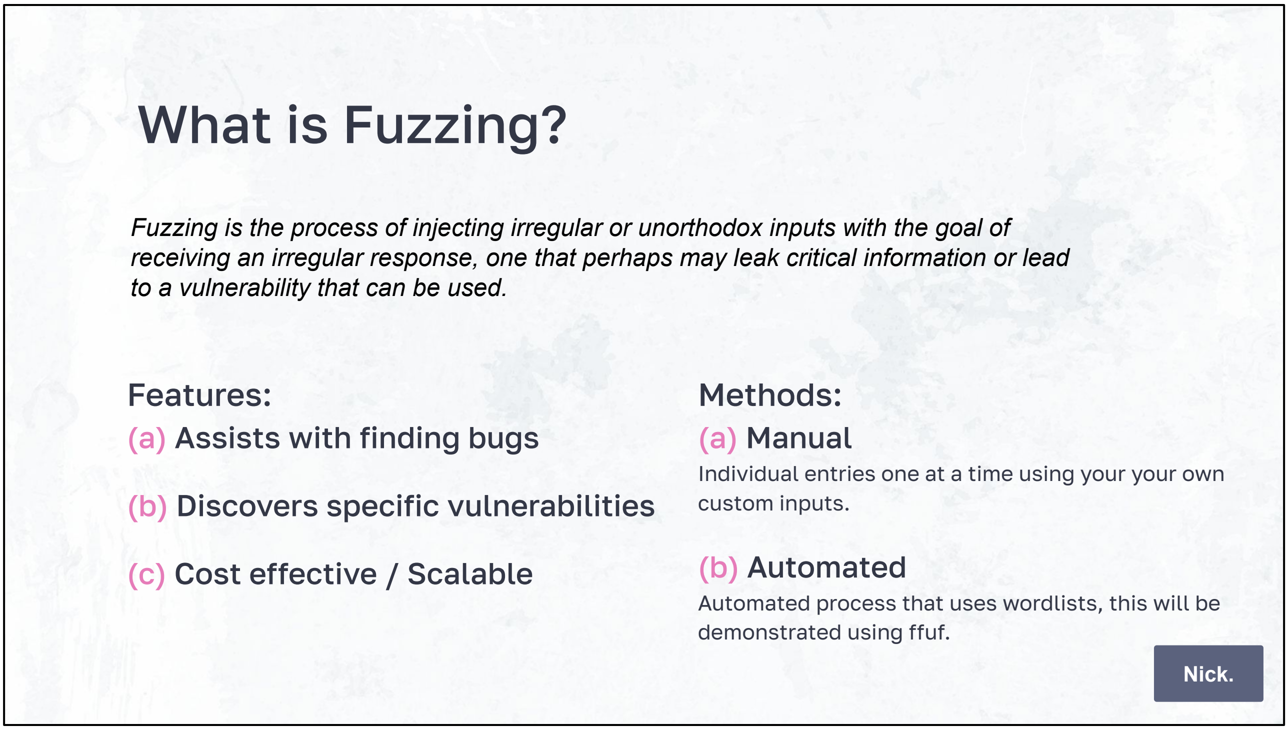
Nick.

Ffuf is a fuzzing tool for attacking web applications.

Ffuf allows you to expose vulnerabilities using wordlists and can also incorporate filters using regular expressions and HTTP response codes which can make your searches much faster.

Some discovered vulnerabilities could Include hidden directories and files, bypassing authentication and compromising sensitive data.

Much like burpsuite, ffuf can be used to bruteforce login pages.



What is Fuzzing?

Fuzzing is the process of injecting irregular or unorthodox inputs with the goal of receiving an irregular response, one that perhaps may leak critical information or lead to a vulnerability that can be used.

Features:

(a) Assists with finding bugs

(b) Discovers specific vulnerabilities

(c) Cost effective / Scalable

Methods:

(a) Manual

Individual entries one at a time using your your own custom inputs.

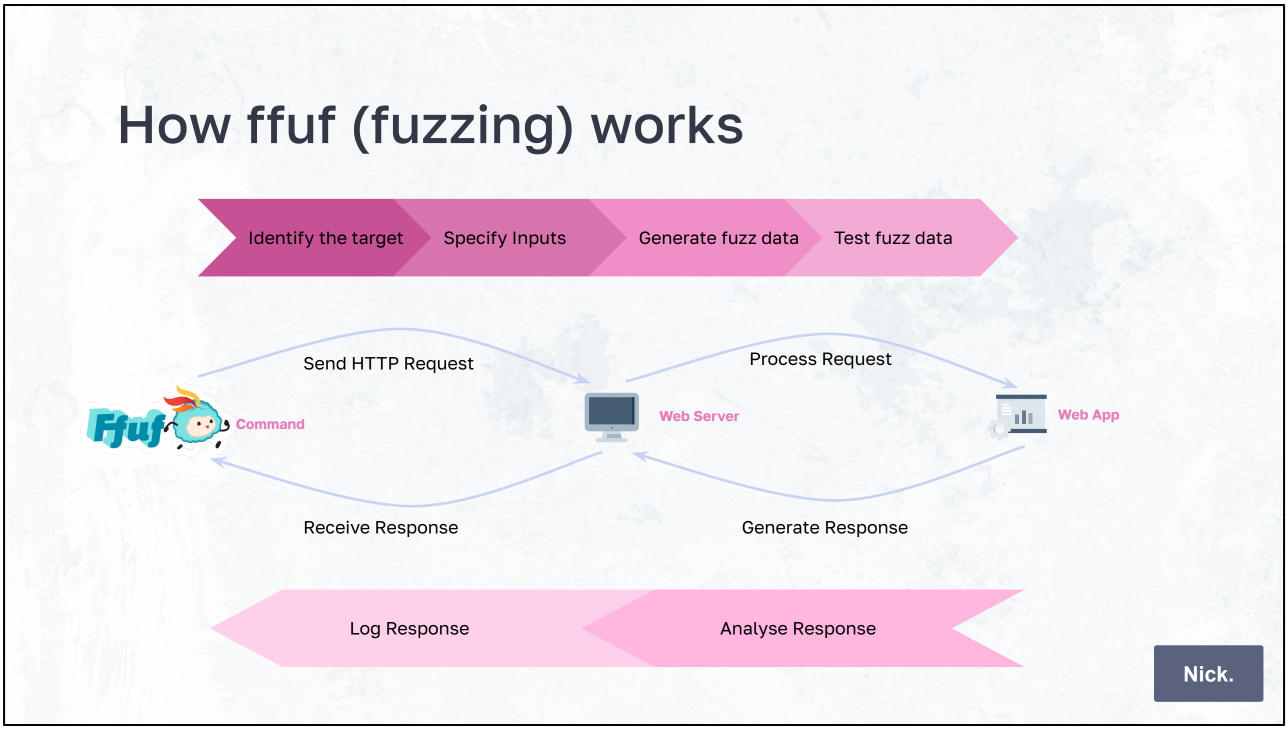
(b) Automated

Automated process that uses wordlists, this will be demonstrated using ffuf.

Nick.

It's important to understand what fuzzing is before we discuss the tool demonstration.

In essence fuzzing is similar to brute forcing except you aren't necessarily trying to input correct credentials; instead use irregular inputs to get irregular results which may yield vulnerabilities.



How ffuf (fuzzing) works

Identify the target

Specify Inputs

Generate fuzz data

Test fuzz data

Ffuf

Command

Send HTTP Request

Web Server

Process Request

Web App

Generate Response

Receive Response

Analyse Response

Log Response

Nick.

We are going to look at a high level view of what ffuf does and how we use it.

Target - Where specifically on the web app to target

Specify Inputs - What input to attack using a keyword such as FUZZ

Generate Fuzz Data - Wordlist

Test fuzz data - Execute Tool

Analyze - Read the results, log what is relevant and then possibly choose a new target and input to start the process again.

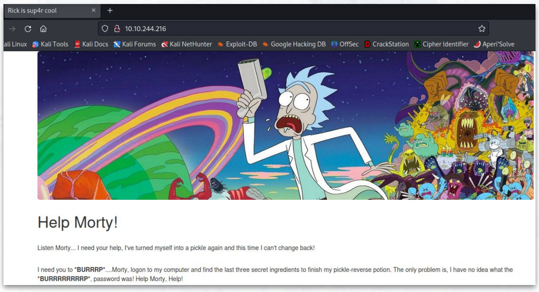
02

Demonstration

TryHackMe: Pickle Rick

Challenge Overview

- Challenge: web based CTF
- Objective: capture 3 flags (ingredients)
- Information Provided: IP Address (10.10.244.216)



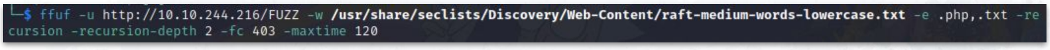
Help Morty!

Listen Morty... I need your help, I've turned myself into a pickle again and this time I can't change back!

I need you to *BURRRP*....Morty, logon to my computer and find the last three secret ingredients to finish my pickle-reverse potion. The only problem is, I have no idea what the *BURRRRRRRRP*, password was! Help Morty, Help!

Ben.

Command: Directory and File enumeration



```
└─$ ffuf -u http://10.10.244.216/FUZZ -w /usr/share/seclists/Discovery/Web-Content/raft-medium-words-lowercase.txt -e .php,.txt -re
cursion -recursion-depth 2 -fc 403 -maxtime 120
```

-u http://10.10.244.216

-w [wordlist path]

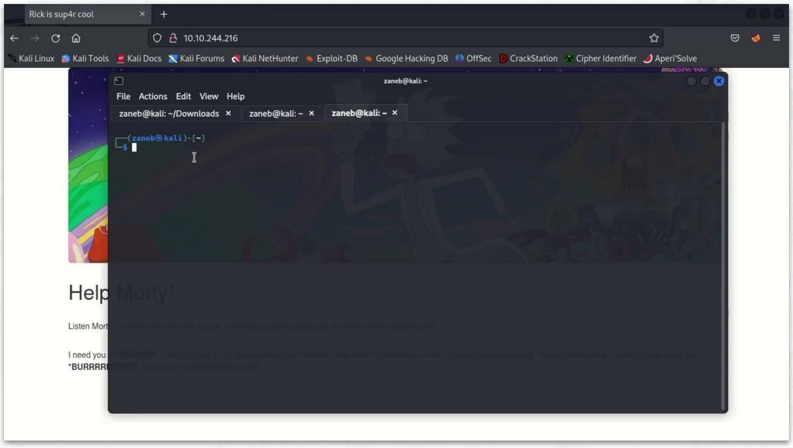
-e [required file extensions]

-recursion

-fc 403

Ben.

Walkthrough - Directory and File Enumeration



```
┌──(zaneb㉿kali)-[~]
└─$ 
```

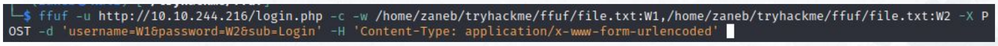
Help Morty!

Listen Mort

I need you
*BURRRR

Ben.

Command: Bruteforce Login



```
└─$ ffuf -u http://10.10.244.216/login.php -c -w /home/zaneb/tryhackme/ffuf/file.txt:W1,/home/zaneb/tryhackme/ffuf/file.txt:W2 -X P
OST -d 'username=W1&password=W2&sub=Login' -H 'Content-Type: application/x-www-form-urlencoded'
```

-u http://10.10.244.216/login.php

-c [colorize output]

-w [wordlist path 1, wordlist path 2]

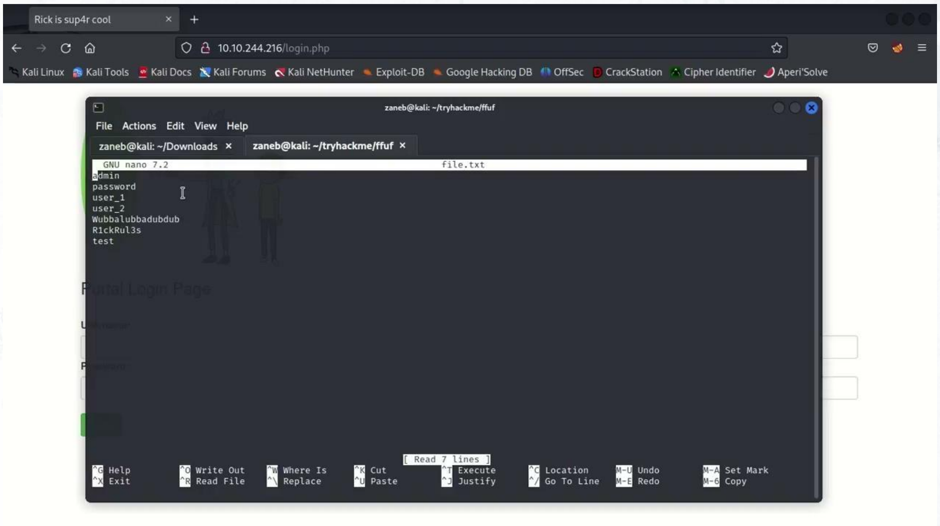
-X [http method]

-d [post the data]

-H [header content type]

Ben.

Walkthrough - Bruteforce Login



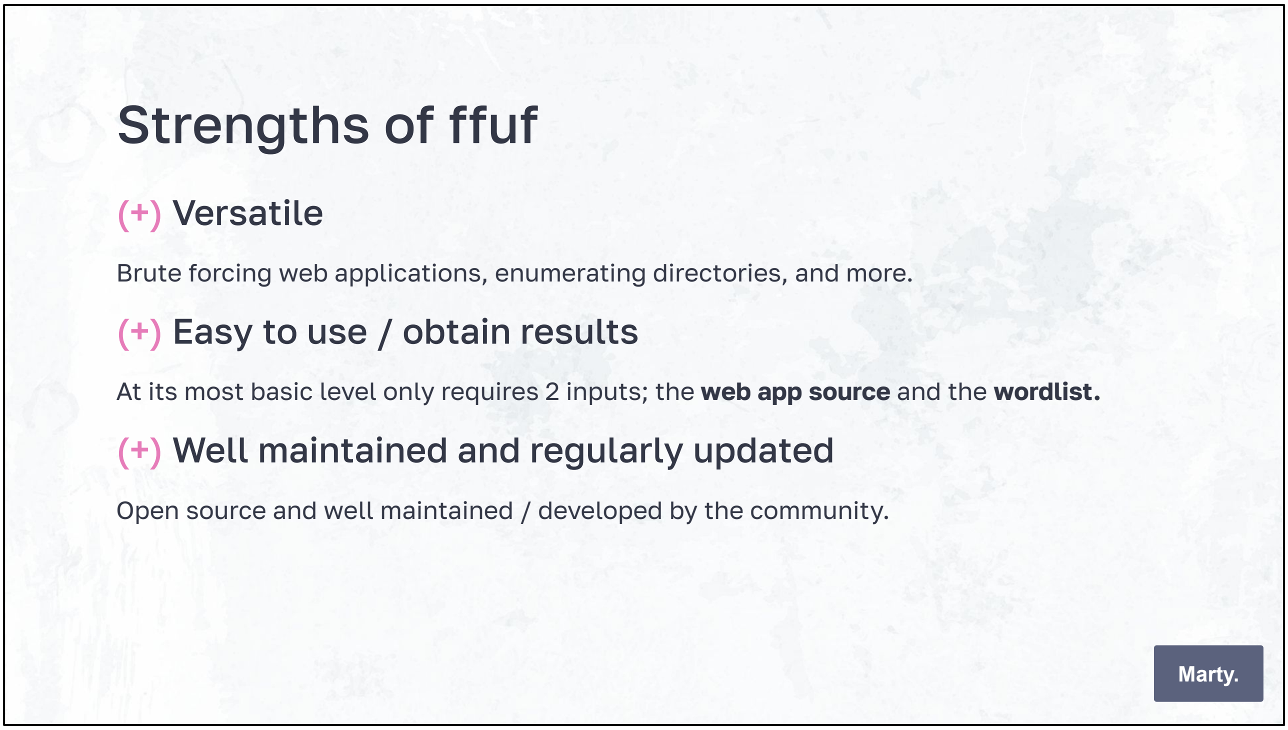
```
  GNU nano 7.2                                file.txt
admin
password
user_1
user_2
Wubbalubbadubdub
R1ckRul3s
test




                                   [ Read 7 lines ]
^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy
```

Ben.

Demonstration Summary

- Through using ffuf, we were able to find a starting point for our attack (i.e. login page) and find the file extension contain a password
- ffuf gave us the ability to quickly search and find the first 2 flags
- The flags have been discovered, but fall out of the scope of this demonstration

Ben.



Strengths of ffuf

(+) Versatile

Brute forcing web applications, enumerating directories, and more.

(+) Easy to use / obtain results

At its most basic level only requires 2 inputs; the **web app source** and the **wordlist.**

(+) Well maintained and regularly updated

Open source and well maintained / developed by the community.

Marty.

Versatile

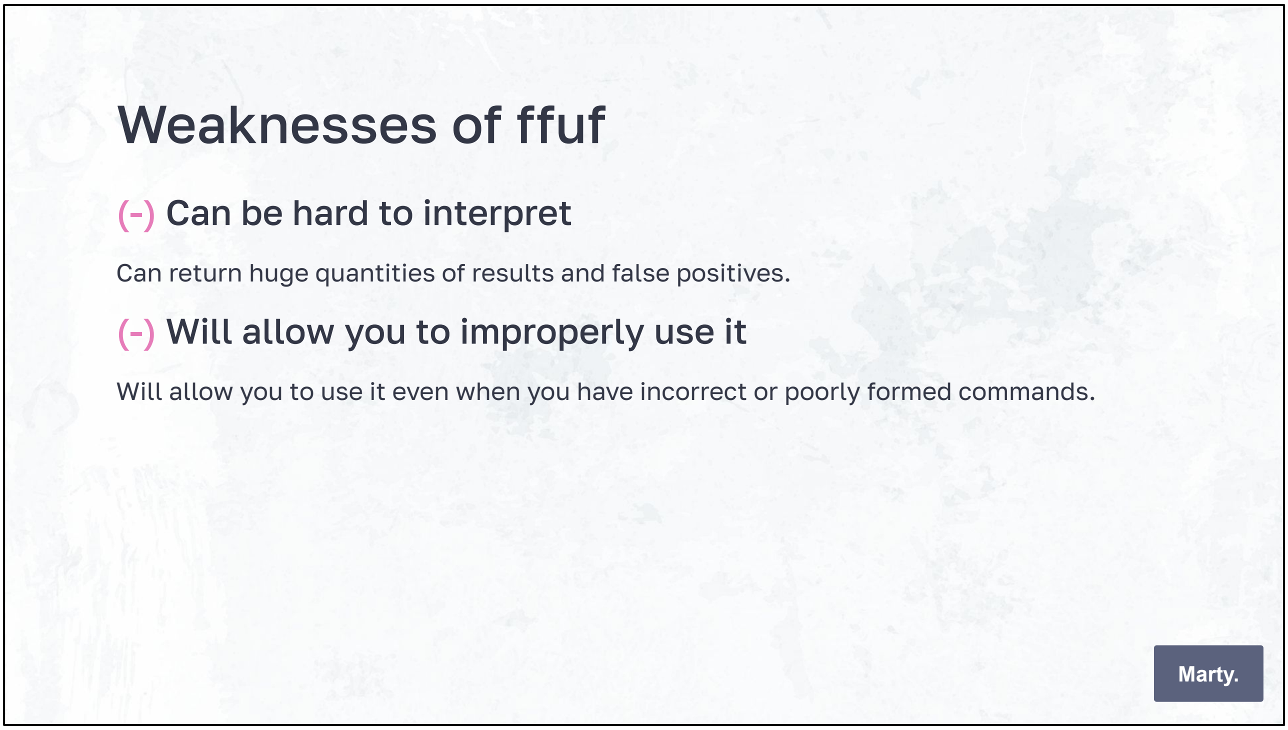
- Ffuf is designed to assist with the discovery of key data on web applications. It can do this through brute forcing web applications and enumerating directories, and more.

Easy to use / obtain results

- Ffuf is incredibly easy to use and provides easy to view/interpret results and at its most basic level only requires 2 inputs; the **web app source** and the **wordlist.**

Well maintained and regularly updated

- The developer/creator of ffuf (Joohoi) continues to work on this tool additionally, as it is open source it is also maintained and developed by the community ensuring that it is up to date and effective.



Weaknesses of ffuf

(-) Can be hard to interpret

Can return huge quantities of results and false positives.

(-) Will allow you to improperly use it

Will allow you to use it even when you have incorrect or poorly formed commands.

Can be hard to interpret

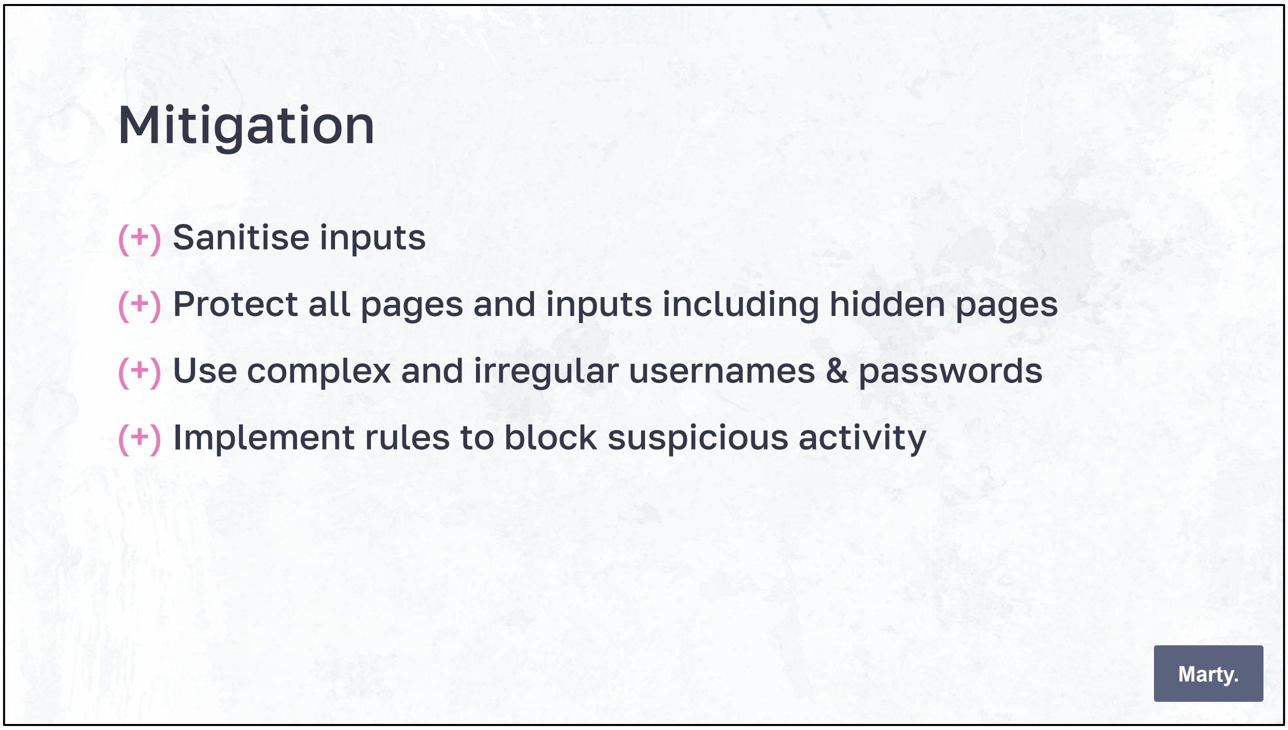
- Due to the nature of ffuf it can return huge quantities of results and can also provide you with false positives based on hard to control variables such as redirects.

Will allow you to improperly use it

- Ffuf as a tool will allow you to use it even if what you are aiming to do is not ideal or even incorrect and won't explicitly notify you unless the error prevents it from functioning.

03

Mitigation



Mitigation

(+) Sanitise inputs

(+) Protect all pages and inputs including hidden pages

(+) Use complex and irregular usernames & passwords

(+) Implement rules to block suspicious activity

Marty.

Sanitise Inputs:

By controlling the specific accepted inputs so that only those specific inputs function and all unrecognised inputs yield a 403 error you prevent input fields being used in a manner other than intended.

Protect all pages and inputs including hidden:

Treating all pages as if they are readily accessible can help with beefing up security on otherwise unsuspecting pages and inputs that typically would be hidden from standard users.

Use complex and irregular usernames & passwords:

To reduce the likelihood of ffuf being used to brute force your web app use irregular and complex usernames and passwords that are unlikely to occur on a wordlist.

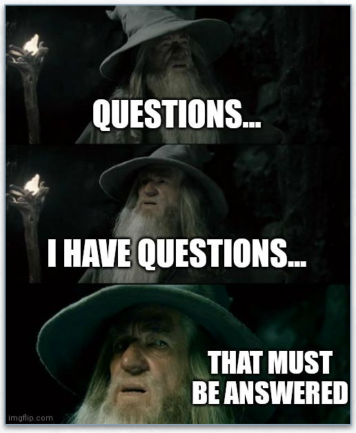
Implement rules to block suspicious activity

In order to mitigate against excessive HTTP requests case of access to an IPS, IPS rules preventing X amount of HTTP requests in X amount of time.

Presentation by:
Ben TM
Marty D
Nick W

Thanks!

Any questions?



04

Appendix

Details

Assets

- Kali Linux Machine
- Vulnerable web app - DVWA, OWASP, Tryhackme CTFs
- Internet connection

Resources

- Ruxmon Melbourne April 2023
- https://tryhackme.com/
- http://ffuf.me/
- PickleRick - BYPASSING Blacklists, John Hammond, Aug 20, 2020
- https://tryhackme.com/room/ffuf
- https://github.com/ffuf/ffuf
- https://codingo.io/tools/ffuf/bounty/2020/09/17/everything-you-need-to-know-about-ffuf.html
- https://owasp.org/www-community/Fuzzing
- https://medium.com/quiknapp/fuzz-faster-with-ffuf-c18c031fc480
- https://allabouttesting.org/top-25-example-usage-of-ffuf-web-fuzzer/
- https://www.freecodecamp.org/news/web-security-fuzz-web-applications-using-ffuf/

Marty.

Demonstration 2

TryHackMe: ffuf

Challenge Overview

- Challenge: web based CTF
- Objective: utilise ffuf to discover, access and exploit vulnerabilities
- Information Provided: IP Addresses (multiple)

ffuf: Enumeration and Brute Forcing

LOGIN ATTEMPT FAILED

ffuf.