Attacking Web application with ffuf: HackTheBox

Name: Daniel Mwendwa Mwithui

ADM NO. CS-SA04-23080

Program: Security Analyst

Date of submission:5th July2023

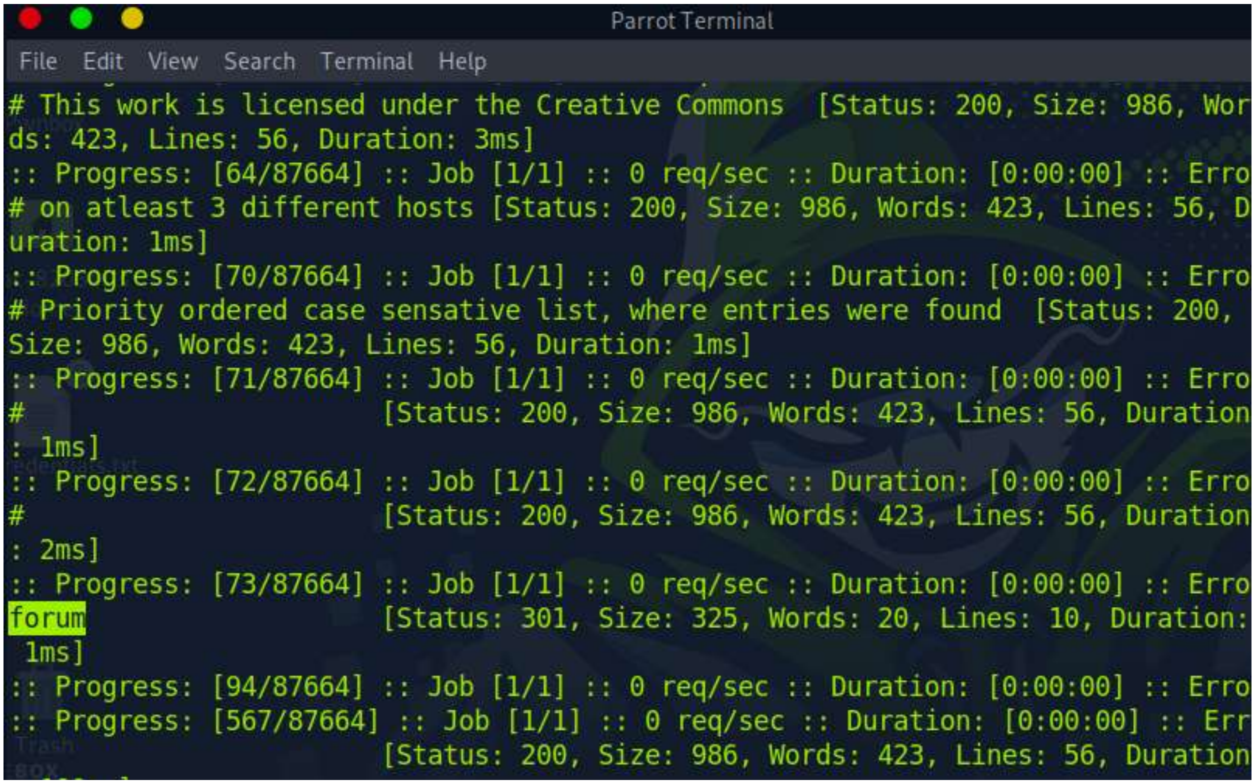
Introduction

In this assignment, we will explore the module "Attacking Web Applications with ffuf" on Hack the Box Academy. Throughout this module, we will discuss the concepts and techniques of using ffuf, a powerful tool for attacking and testing web applications. We will learn about directory fuzzing, page fuzzing, sub-domain fuzzing, parameter fuzzing, value fuzzing and complete by doing a skill assessment that tests the knowledge gained through the module. By understanding these techniques, we will have gained insights on how to identify vulnerabilities and uncover hidden or unknown parts of web applications.

Task 1: Directory Fuzzing

Directory fuzzing using ffuf is a technique used to find hidden or unknown directories or files on a website or web server. It is done by systematically trying different directory and file names to see if they exist on the target website. Think of it like trying different keys to open different doors. In this case, the "doors" are directories or files on a website, and ffuf is the tool that tries different "keys" (directory and file names) to see if any of them work. By fuzzing directories, you can potentially discover sensitive or confidential information that was not meant to be publicly accessible. This technique is often used by security researchers or hackers to identify vulnerabilities in websites or web servers.

Running *ffuf -w /opt/useful/SecLists/Discovery/Web-Content/directory-list-2.3-small.txt:FUZZ -u* *http://SERVER_IP:PORT/FUZZ* helps us find the other hidden directory. See the screenshot below.

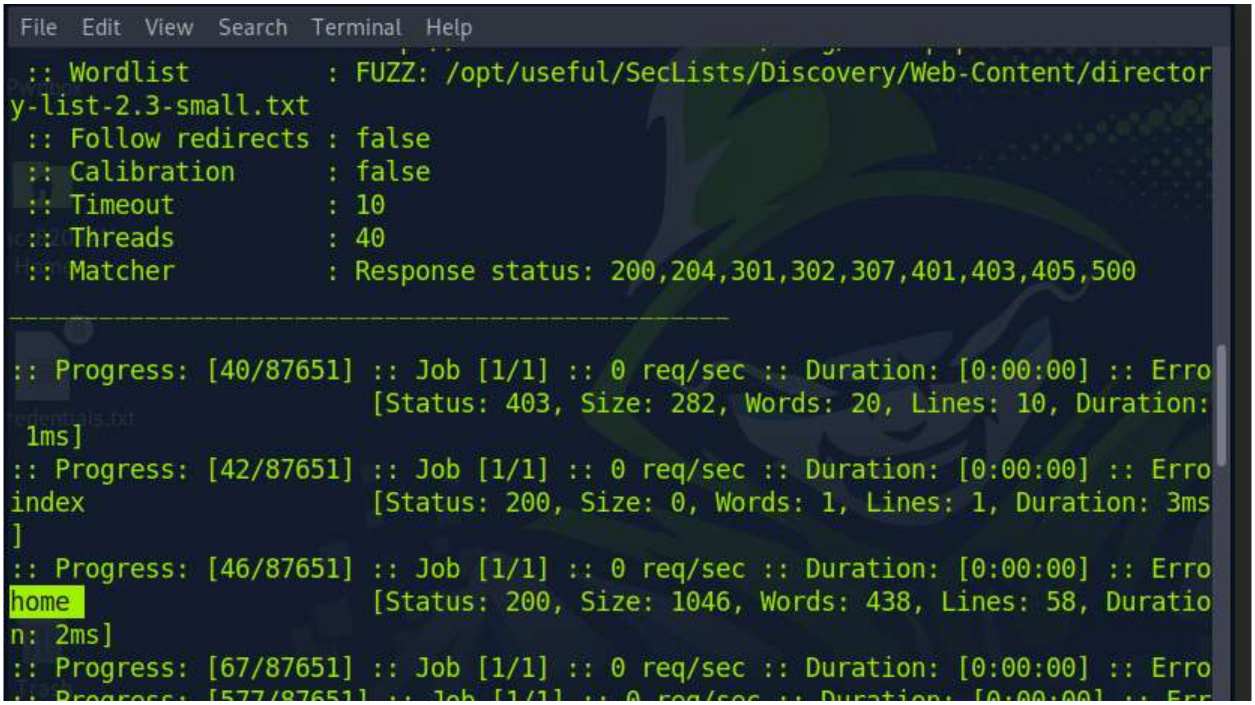


```
# This work is licensed under the Creative Commons  [Status: 200, Size: 986, Wor
ds: 423, Lines: 56, Duration: 3ms]
:: Progress: [64/87664] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
# on atleast 3 different hosts [Status: 200, Size: 986, Words: 423, Lines: 56, D
uration: 1ms]
:: Progress: [70/87664] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
# Priority ordered case sensative list, where entries were found  [Status: 200,
Size: 986, Words: 423, Lines: 56, Duration: 1ms]
:: Progress: [71/87664] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
#                       [Status: 200, Size: 986, Words: 423, Lines: 56, Duration
: 1ms]
:: Progress: [72/87664] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
#                       [Status: 200, Size: 986, Words: 423, Lines: 56, Duration
: 2ms]
:: Progress: [73/87664] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
forum                   [Status: 301, Size: 325, Words: 20, Lines: 10, Duration:
 1ms]
:: Progress: [94/87664] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
:: Progress: [567/87664] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Err
                        [Status: 200, Size: 986, Words: 423, Lines: 56, Duration
```

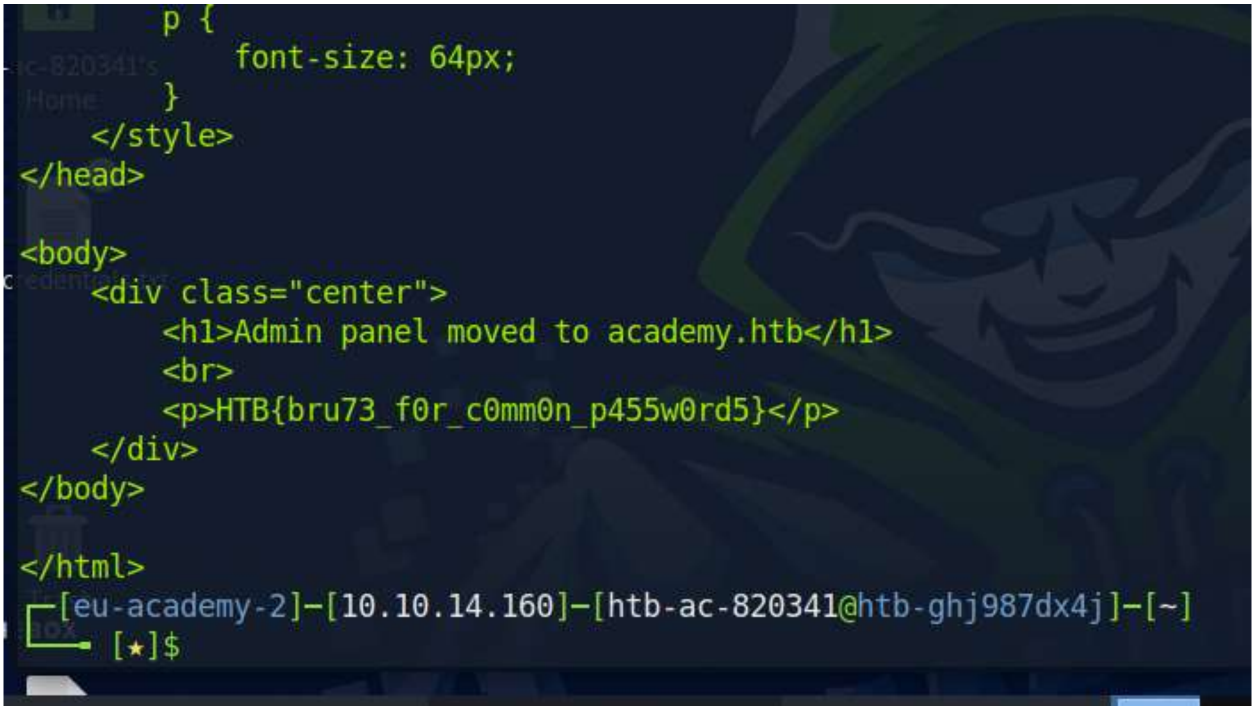
Task 2: Page Fuzzing

Page fuzzing is a technique used to test the behavior of a web application by sending it unexpected or invalid inputs. It involves intentionally sending malformed or unusual data to different fields or parameters of a webpage to see how the application handles them. To understand it better, imagine you're playing a game and you try different combinations of moves or inputs to see if you can discover any unexpected behaviors or glitches. Page fuzzing works in a similar way, where you systematically test different inputs on a webpage to find vulnerabilities or errors in the application's response. By fuzzing pages, security researchers or testers can identify potential weaknesses in the web application, such as input validation flaws or security vulnerabilities. It helps uncover how the application responds to unexpected or malicious input, which can be crucial for identifying and fixing potential issues before they can be exploited by attackers.

Here running this command *ffuf -w /opt/useful/SecLists/Discovery/Web-Content/directory-list-2.3-small.txt:FUZZ -u http://target_ip:port/blog/FUZZ.php -ic* gives us home page which is hidden.-ic ignores the comments in the directory list. Browsing http://target_ip:port/blog/home.php or getting its source code using curl command gives us the flag. See the screenshot below.



```
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/director
y-list-2.3-small.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403,405,500
________________________________________________

:: Progress: [40/87651] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
                        [Status: 403, Size: 282, Words: 20, Lines: 10, Duration:
 1ms]
:: Progress: [42/87651] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
index                   [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 3ms
]
:: Progress: [46/87651] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
home                    [Status: 200, Size: 1046, Words: 438, Lines: 58, Duratio
n: 2ms]
:: Progress: [67/87651] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
:: Progress: [577/87651] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Err
```

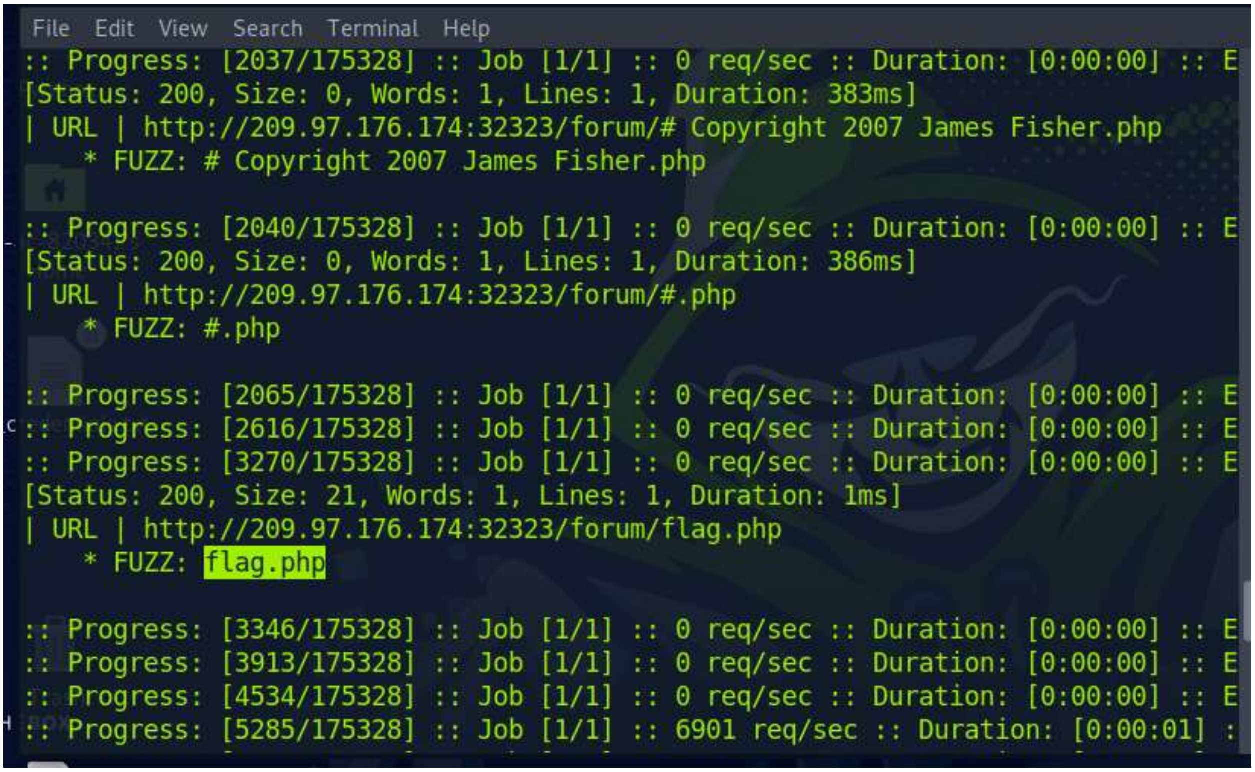


```
        p {
            font-size: 64px;
        }
    </style>
</head>

<body>
    <div class="center">
        <h1>Admin panel moved to academy.htb</h1>
        <br>
        <p>HTB{bru73_f0r_c0mm0n_p455w0rd5}</p>
    </div>
</body>

</html>
┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx4j]─[~]
└──╼ [★]$
```

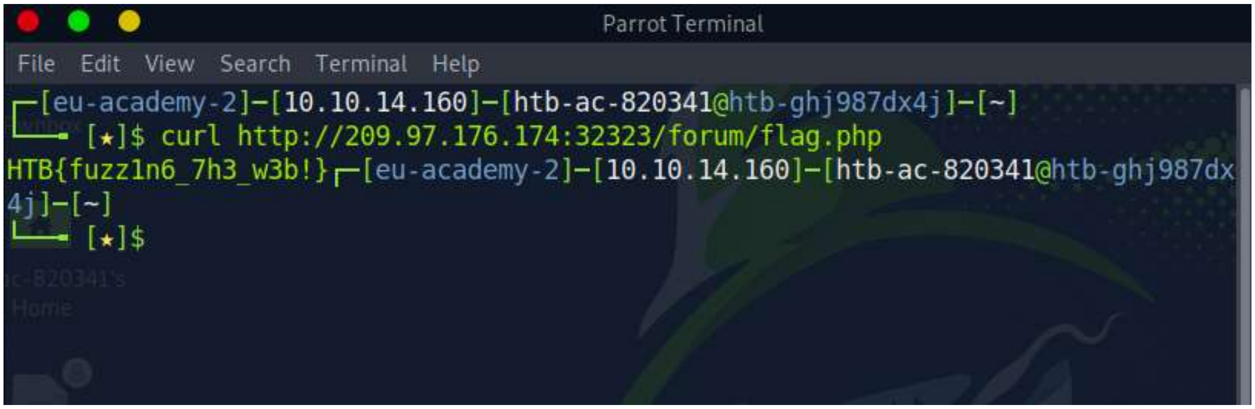
Task 3: Recursive Fuzzing

Recursive fuzzing is a technique used to explore and test multiple levels of a web application or system by automatically applying fuzzing to different parts of it. Instead of fuzzing just one component or parameter, recursive fuzzing extends the fuzzing process to encompass interconnected components, directories, or functionality within the application. To understand it better, imagine you have a map of a city. Instead of exploring just one street, recursive fuzzing allows you to explore all the streets, alleys, and buildings connected to that street. It helps you discover more areas and potential vulnerabilities within the application. We can use -recursion to do recursive fuzzing and -recursion-depth to set how far our fuzzing will go.

This command *ffuf -w /opt/useful/SecLists/Discovery/Web-Content/directory-list-2.3-small.txt:FUZZ -u http://SERVER_IP:PORT/forum/FUZZ -recursion -recursion-depth 1 -e .php -v* gives us the flag.php file. You can curl http://machine_p:port/forum/flag.php to get the flag text. We specified the forum directory to make the scanning faster. This will not scan the other blog directory. See the screenshot below.



```
:: Progress: [2037/175328] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: E
[Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 383ms]
| URL | http://209.97.176.174:32323/forum/# Copyright 2007 James Fisher.php
    * FUZZ: # Copyright 2007 James Fisher.php

:: Progress: [2040/175328] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: E
[Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 386ms]
| URL | http://209.97.176.174:32323/forum/#.php
    * FUZZ: #.php

:: Progress: [2065/175328] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: E
:: Progress: [2616/175328] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: E
:: Progress: [3270/175328] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: E
[Status: 200, Size: 21, Words: 1, Lines: 1, Duration: 1ms]
| URL | http://209.97.176.174:32323/forum/flag.php
    * FUZZ: flag.php

:: Progress: [3346/175328] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: E
:: Progress: [3913/175328] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: E
:: Progress: [4534/175328] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: E
:: Progress: [5285/175328] :: Job [1/1] :: 6901 req/sec :: Duration: [0:00:01] :
```

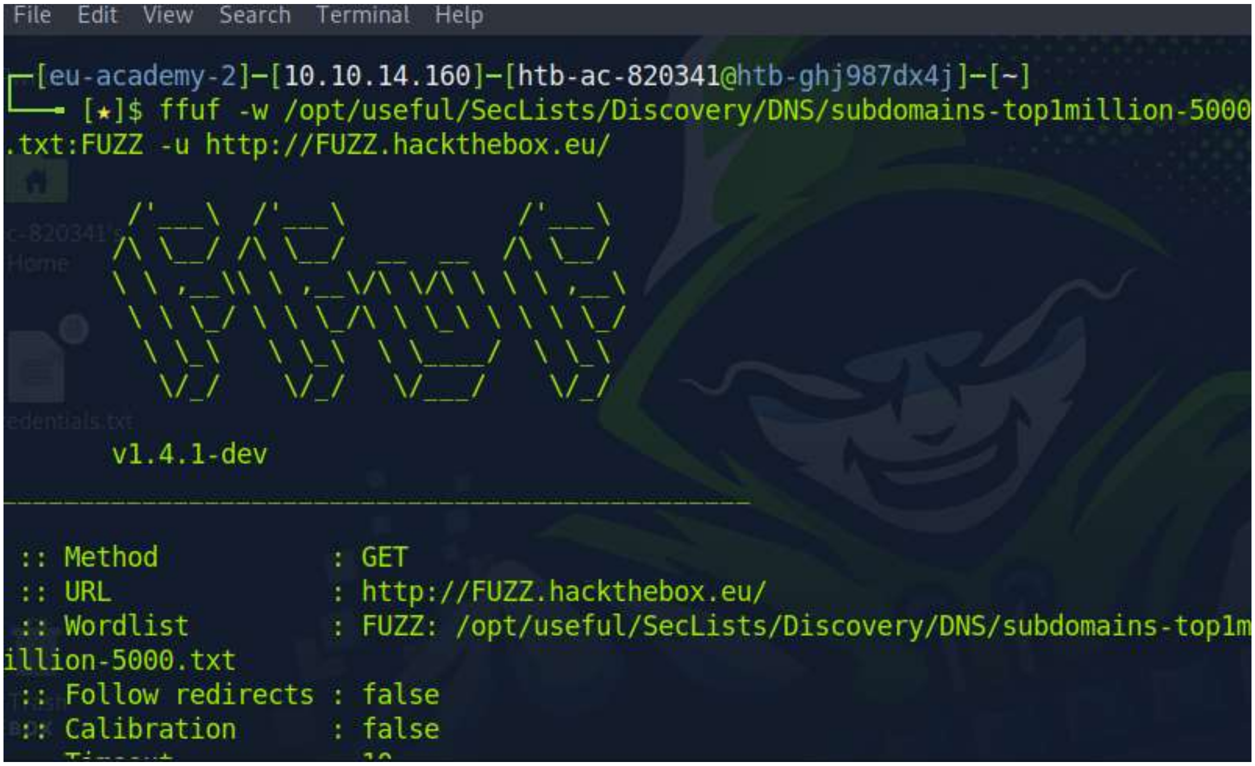


```
┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx4j]─[~]
└──╼ [★]$ curl http://209.97.176.174:32323/forum/flag.php
HTB{fuzz1n6_7h3_w3b!}┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx
4j]─[~]
└──╼ [★]$
```

Task 4: Sub-Domain Fuzzing

Sub-domain fuzzing is a technique used to discover new or hidden sub-domains associated with a target domain. It involves systematically testing different combinations of words, letters, or numbers as sub-domains in order to find valid ones. Imagine you have a big box of puzzle pieces, and you're trying to find all the missing pieces that fit into a specific section. Sub-domain fuzzing is like trying different puzzle pieces to see if they fit as sub-domains of a target domain. By fuzzing sub-domains, security researchers or testers can uncover sub-domains that may not be publicly known or accessible. These sub-domains could potentially contain sensitive information, misconfigurations, or even vulnerable systems.

Here we use command *ffuf -w /opt/useful/SecLists/Discovery/DNS/subdomains-top1million-5000.txt:FUZZ -u https://FUZZ.hackthebox.eu/* to find for any hidden domains.

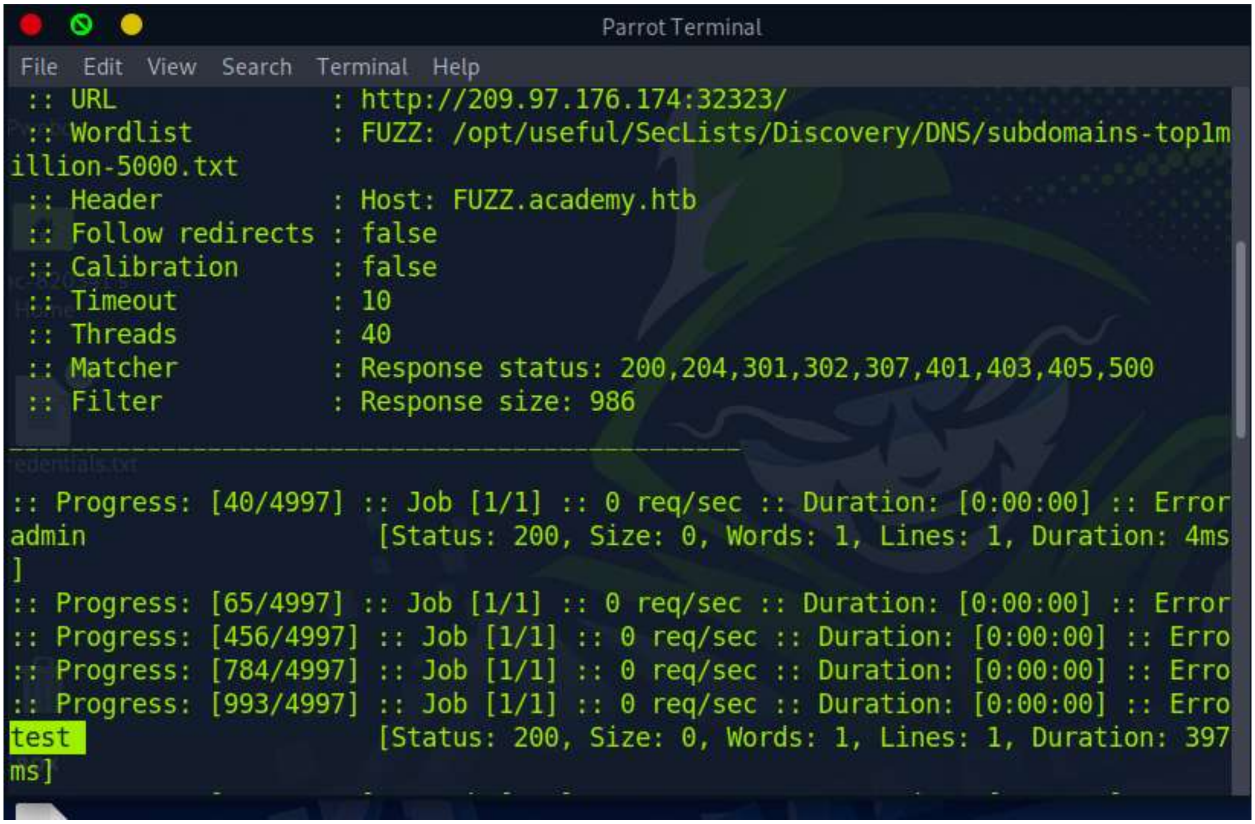


```
┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx4j]─[~]
└──╼ [★]$ ffuf -w /opt/useful/SecLists/Discovery/DNS/subdomains-top1million-5000
.txt:FUZZ -u http://FUZZ.hackthebox.eu/

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://FUZZ.hackthebox.eu/
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/DNS/subdomains-top1m
illion-5000.txt
 :: Follow redirects : false
 :: Calibration      : false
```

Task 5: Vhost Fuzzing

Vhost fuzzing is a technique used to find virtual hosts or web applications hosted on a single IP address or server. It involves systematically testing different hostnames or domain names to identify additional web applications or services running on the same server. By fuzzing virtual hosts, security researchers or testers can uncover hidden or unknown web applications that share the same IP address or server. These virtual hosts might be running different websites, APIs, or services, each with its own unique functionality and potential vulnerabilities.

The command *ffuf -w /opt/useful/SecLists/Discovery/DNS/subdomains-top1million-5000.txt:FUZZ -u http://ip:PORT/ -H 'Host: FUZZ.academy.htb' -fs 986* specify the vhost header using -H and filter content to 986. See the screenshot below.

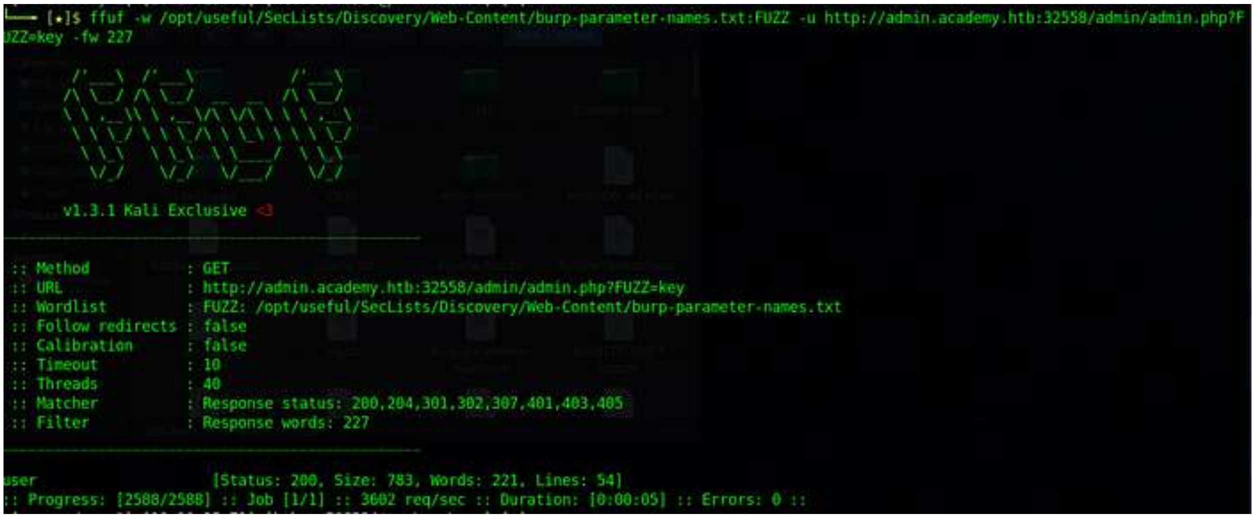


```
 :: URL              : http://209.97.176.174:32323/
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/DNS/subdomains-top1m
illion-5000.txt
 :: Header           : Host: FUZZ.academy.htb
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403,405,500
 :: Filter           : Response size: 986
________________________________________________

:: Progress: [40/4997] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Error
admin                   [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 4ms
]
:: Progress: [65/4997] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Error
:: Progress: [456/4997] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
:: Progress: [784/4997] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
:: Progress: [993/4997] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Erro
test                    [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 397
ms]
```

Task 6: Parameter Fuzzing

Parameter fuzzing is a technique used to test the behavior of a web application by systematically modifying or injecting different values into its parameters. Parameters are variables or fields within a URL, form, or API request that carry user input or control application behavior. Parameters can either be GET or POST. GET parameters are part of the URL itself. They are appended to the end of the URL after a question mark '?' and are formatted as key-value pairs separated by an ampersand '&'. POST parameters, on the other hand, are sent within the body of an HTTP POST request. They are not visible in the URL. POST parameters are commonly used when submitting forms or sending data that is not suitable to be exposed in the URL.

The command *ffuf -w /opt/useful/SecLists/Discovery/Web-Content/burp-parameter-names.txt:FUZZ -u http://admin.academy.htb:PORT/admin/admin.php?FUZZ=key -fs 227* will help us solve this section. See the screenshot below.

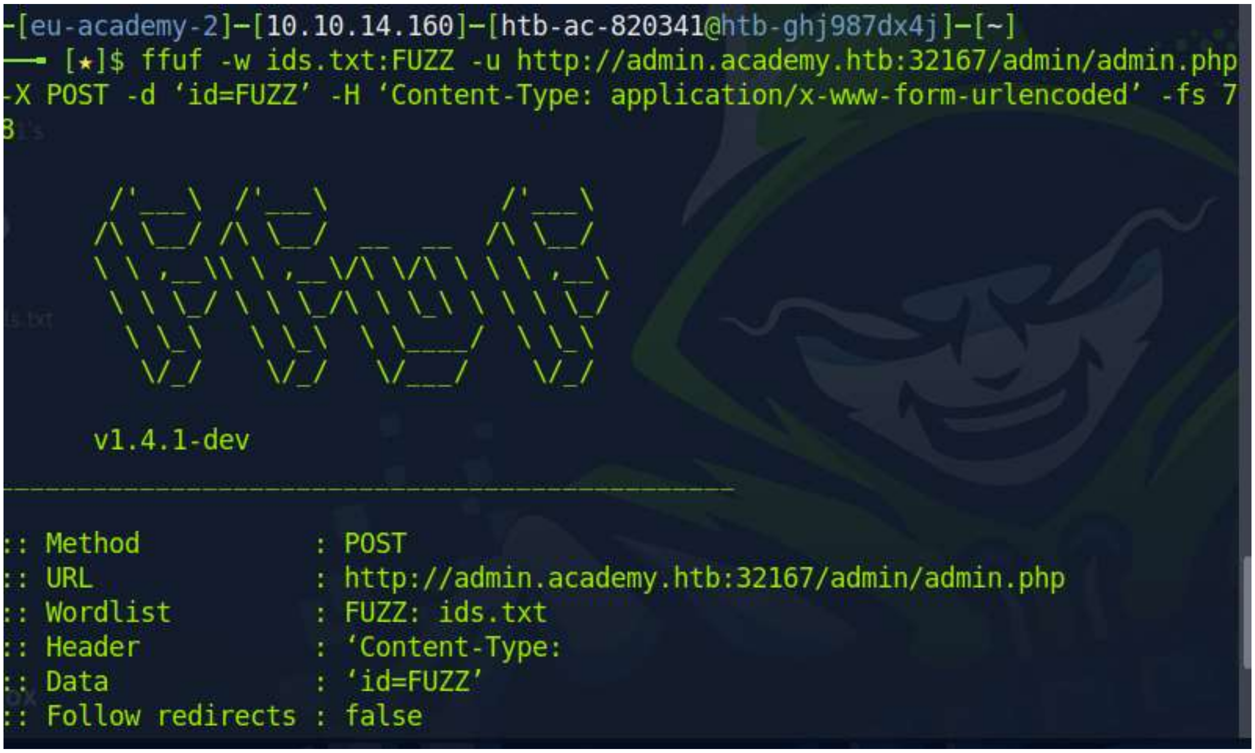


```
└── [★]$ ffuf -w /opt/useful/SecLists/Discovery/Web-Content/burp-parameter-names.txt:FUZZ -u http://admin.academy.htb:32558/admin/admin.php?F
UZZ=key -fw 227

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.3.1 Kali Exclusive <3
________________________________________________

 :: Method           : GET
 :: URL              : http://admin.academy.htb:32558/admin/admin.php?FUZZ=key
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/burp-parameter-names.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403,405
 :: Filter           : Response words: 227
________________________________________________

user                    [Status: 200, Size: 783, Words: 221, Lines: 54]
:: Progress: [2588/2588] :: Job [1/1] :: 3602 req/sec :: Duration: [0:00:05] :: Errors: 0 ::
```

Task 7: Value Fuzzing

Value fuzzing is a technique used to test the behavior of a web application by systematically modifying the values of parameters or inputs. It involves sending different combinations of data, such as invalid, unexpected, or extreme values, to test how the application handles them. The purpose of value fuzzing is to uncover vulnerabilities, security flaws, or unexpected behaviors in the application's response to different input values.

ffuf -w ids.txt:FUZZ -u http://admin.academy.htb:port/admin/admin.php -X POST -d ‘id=FUZZ’ -H ‘Content-Type: application/x-www-form-urlencoded’ -fs 768. Then a curl request with id=73:



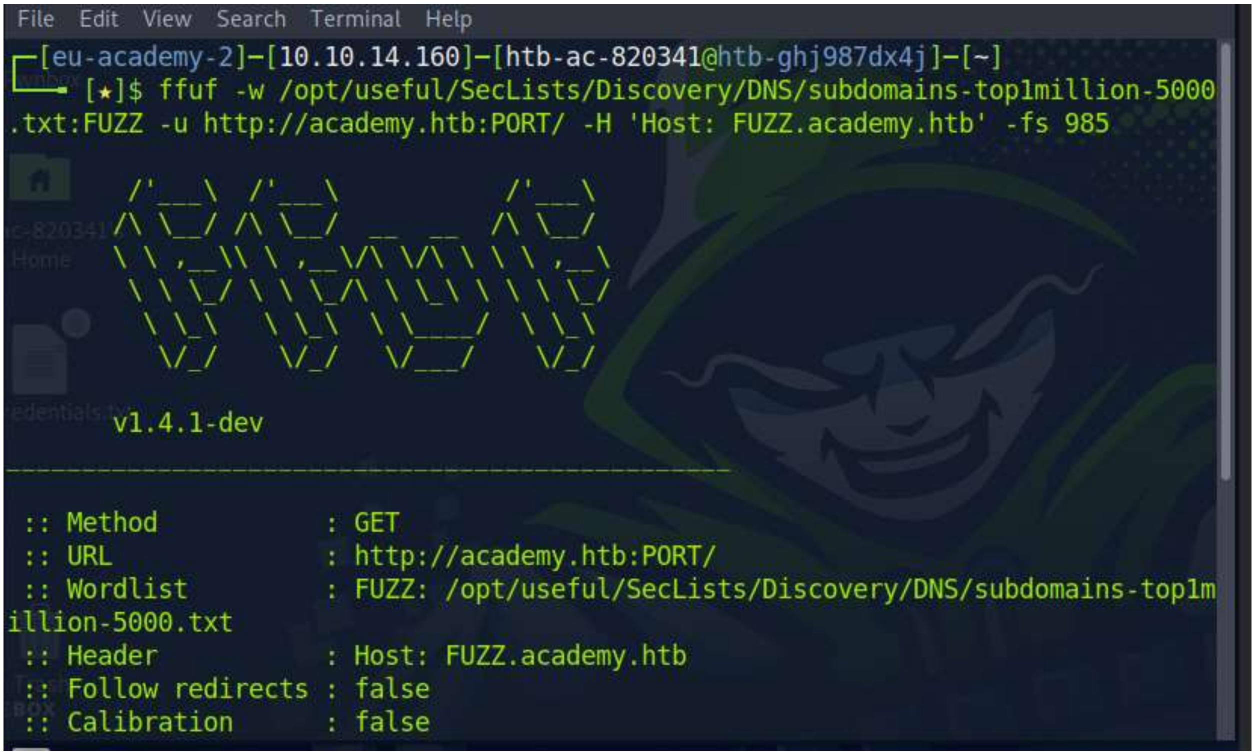
```
-[eu-academy-2]-[10.10.14.160]-[htb-ac-820341@htb-ghj987dx4j]-[~]
—— [★]$ ffuf -w ids.txt:FUZZ -u http://admin.academy.htb:32167/admin/admin.php
-X POST -d ‘id=FUZZ’ -H ‘Content-Type: application/x-www-form-urlencoded’ -fs 7
8

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
________________________________________________

:: Method           : POST
:: URL              : http://admin.academy.htb:32167/admin/admin.php
:: Wordlist         : FUZZ: ids.txt
:: Header           : ‘Content-Type:
:: Data             : ‘id=FUZZ’
:: Follow redirects : false
```

curl http://admin.academy.htb:port/admin/admin.php -X POST -d ‘id=73’ -H ‘Content-Type: application/x-www-form-urlencoded’.

Task 8: Skill Assessment.

In this section, we are going to apply the knowledge gained through the above module to complete the section. In question one, command *ffuf -w*

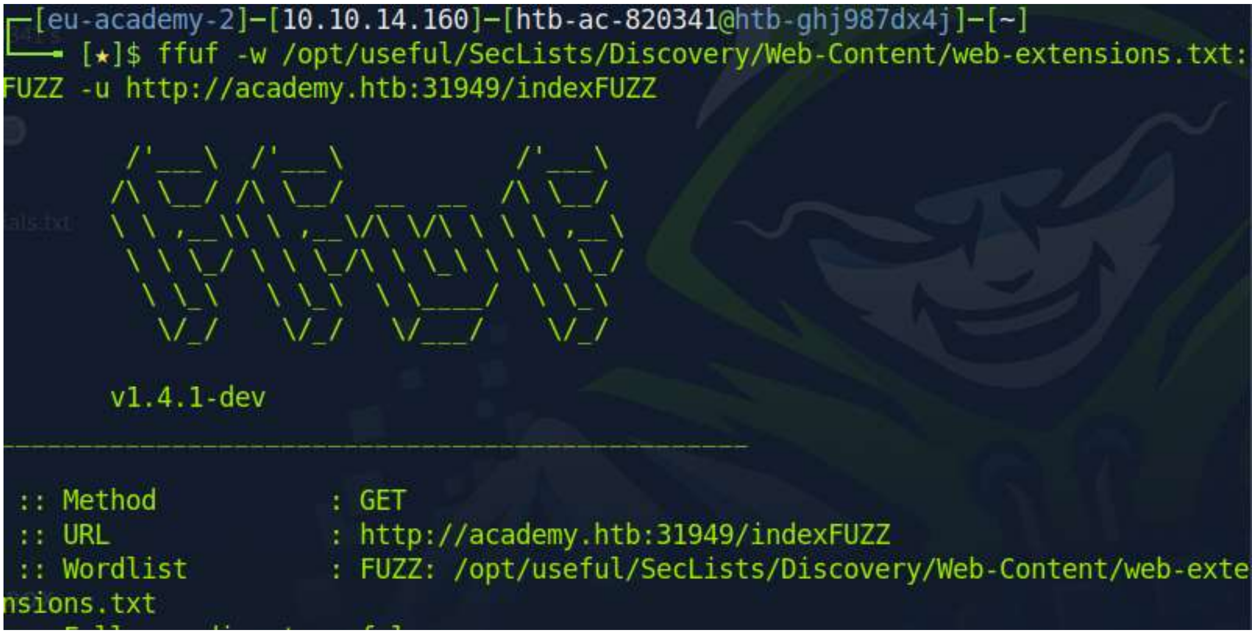
/opt/useful/SecLists/Discovery/DNS/subdomains-top1million-5000.txt:FUZZ -u http://academy.htb:PORT/ -H 'Host: FUZZ.academy.htb' -fs 985 helps us check for the vhosts. See the screenshot below.



```
┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx4j]─[~]
└──╼ [★]$ ffuf -w /opt/useful/SecLists/Discovery/DNS/subdomains-top1million-5000
.txt:FUZZ -u http://academy.htb:PORT/ -H 'Host: FUZZ.academy.htb' -fs 985

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://academy.htb:PORT/
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/DNS/subdomains-top1m
illion-5000.txt
 :: Header           : Host: FUZZ.academy.htb
 :: Follow redirects : false
 :: Calibration      : false
```

In the second question running the following commands will help us get extensions accepted by the domain for all of the above vhosts.

- ffuf -w /opt/useful/SecLists/Discovery/Web-Content/web-extensions.txt:FUZZ -u http://academy.htb:PORT/indexFUZZ



```
┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx4j]─[~]
└──╼ [★]$ ffuf -w /opt/useful/SecLists/Discovery/Web-Content/web-extensions.txt:
FUZZ -u http://academy.htb:31949/indexFUZZ

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://academy.htb:31949/indexFUZZ
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/web-exte
nsions.txt
```

- ffuf -w /opt/useful/SecLists/Discovery/Web-Content/web-extensions.txt:FUZZ -u

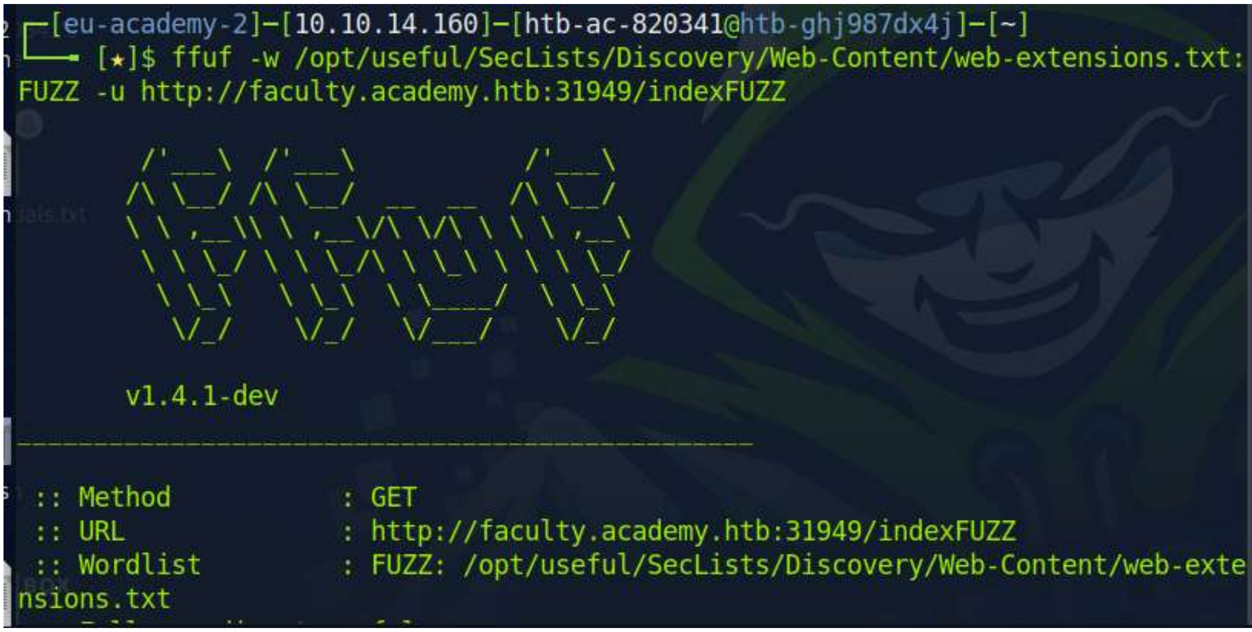
  http://test.academy.htb:PORT/indexFUZZ

- ffuf -w /opt/useful/SecLists/Discovery/Web-Content/web-extensions.txt:FUZZ -u

  http://archive.academy.htb:PORT/indexFUZZ

- ffuf -w /opt/useful/SecLists/Discovery/Web-Content/web-extensions.txt:FUZZ -u

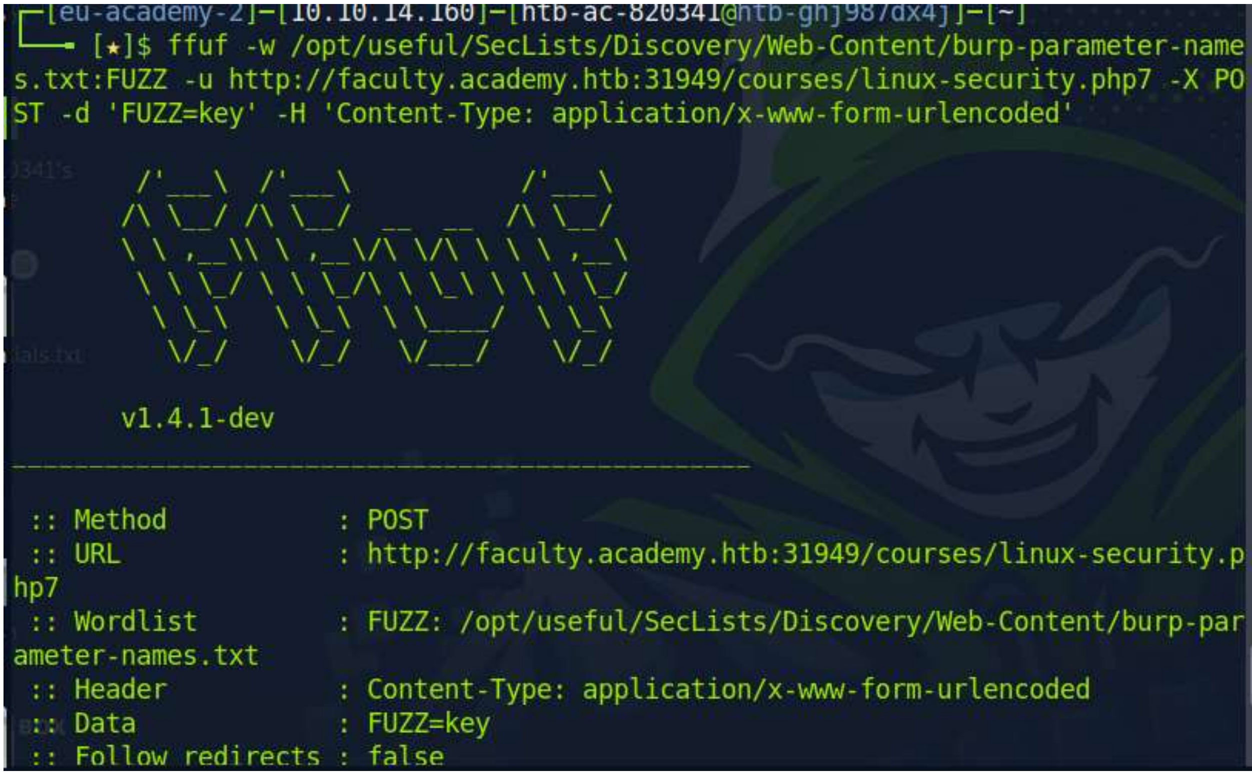
  http://faculty.academy.htb:PORT/indexFUZZ



```
┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx4j]─[~]
└──╼ [★]$ ffuf -w /opt/useful/SecLists/Discovery/Web-Content/web-extensions.txt:
FUZZ -u http://faculty.academy.htb:31949/indexFUZZ

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://faculty.academy.htb:31949/indexFUZZ
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/web-exte
nsions.txt
```

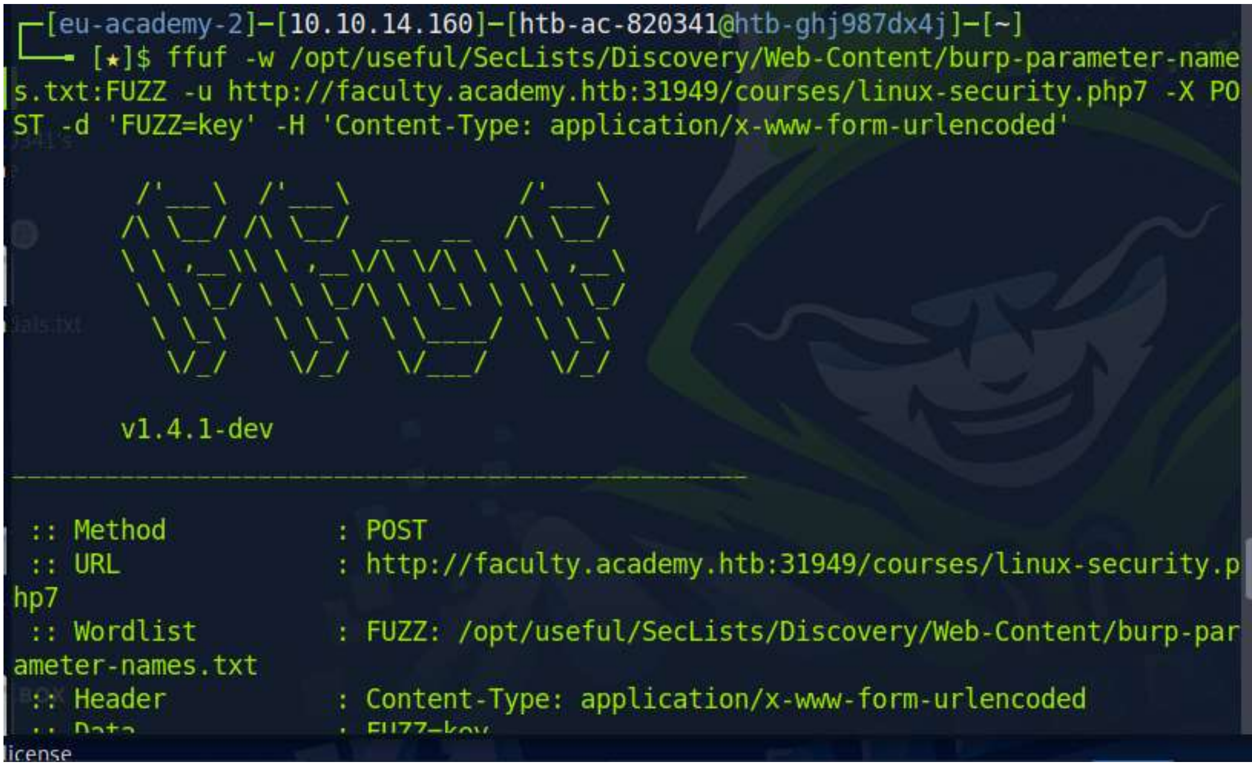
The third question in this section wants us to find parameters accepted by the page. Here we use the following command to perform a parameter fuzzing.

- ffuf -w /opt/useful/SecLists/Discovery/Web-Content/burp-parameter-names.txt:FUZZ -u http://faculty.academy.htb:PORT/courses/linux-security.php7 -X POST -d 'FUZZ=key' -H 'Content-Type: application/x-www-form-urlencoded'



```
┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx4]─[~]
└──╼ [★]$ ffuf -w /opt/useful/SecLists/Discovery/Web-Content/burp-parameter-name
s.txt:FUZZ -u http://faculty.academy.htb:31949/courses/linux-security.php7 -X PO
ST -d 'FUZZ=key' -H 'Content-Type: application/x-www-form-urlencoded'

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
________________________________________________

 :: Method           : POST
 :: URL              : http://faculty.academy.htb:31949/courses/linux-security.p
hp7
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/burp-par
ameter-names.txt
 :: Header           : Content-Type: application/x-www-form-urlencoded
 :: Data             : FUZZ=key
 :: Follow redirects : false
```

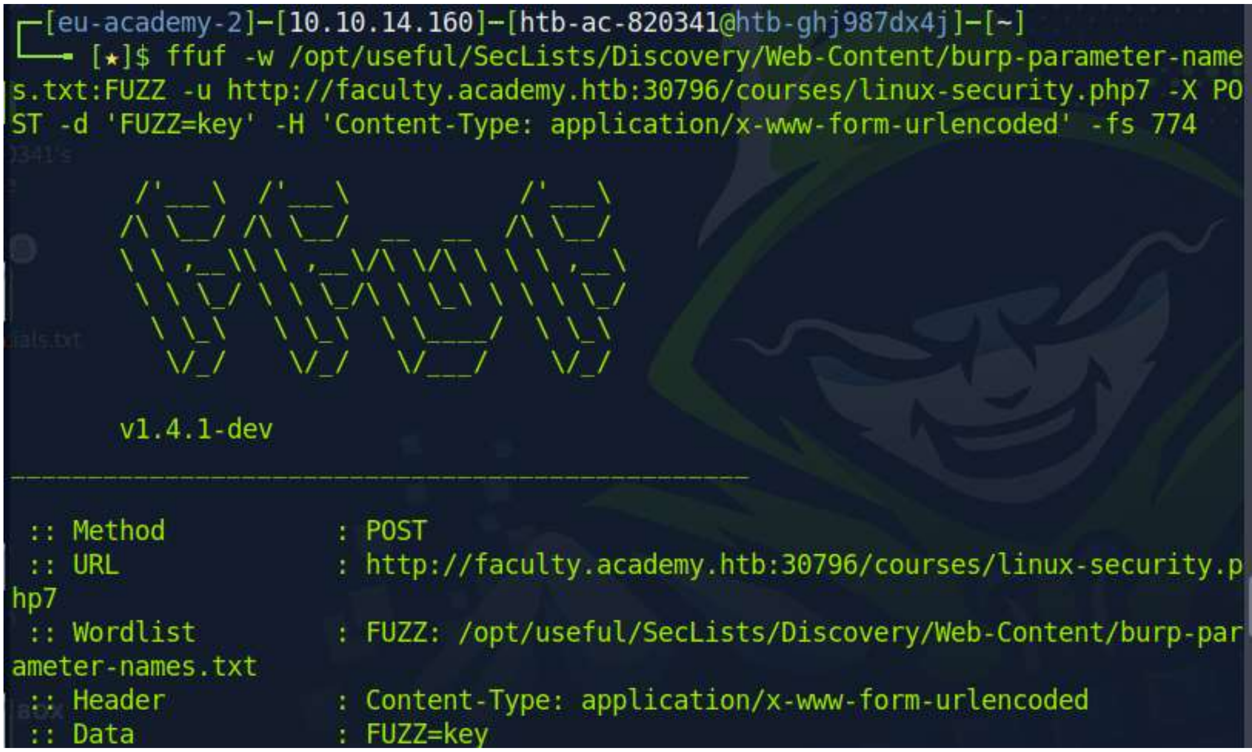
- ffuf -w /opt/useful/SecLists/Discovery/Web-Content/burp-parameter-names.txt:FUZZ -u http://faculty.academy.htb:30796/courses/linux-security.php7 -X POST -d 'FUZZ=key' -H 'Content-Type: application/x-www-form-urlencoded' -fs 774



```
┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx4j]─[~]
└──╼ [★]$ ffuf -w /opt/useful/SecLists/Discovery/Web-Content/burp-parameter-name
s.txt:FUZZ -u http://faculty.academy.htb:31949/courses/linux-security.php7 -X PO
ST -d 'FUZZ=key' -H 'Content-Type: application/x-www-form-urlencoded'

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
________________________________________________

 :: Method           : POST
 :: URL              : http://faculty.academy.htb:31949/courses/linux-security.p
hp7
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/burp-par
ameter-names.txt
 :: Header           : Content-Type: application/x-www-form-urlencoded
 :: Data             : FUZZ=key
```

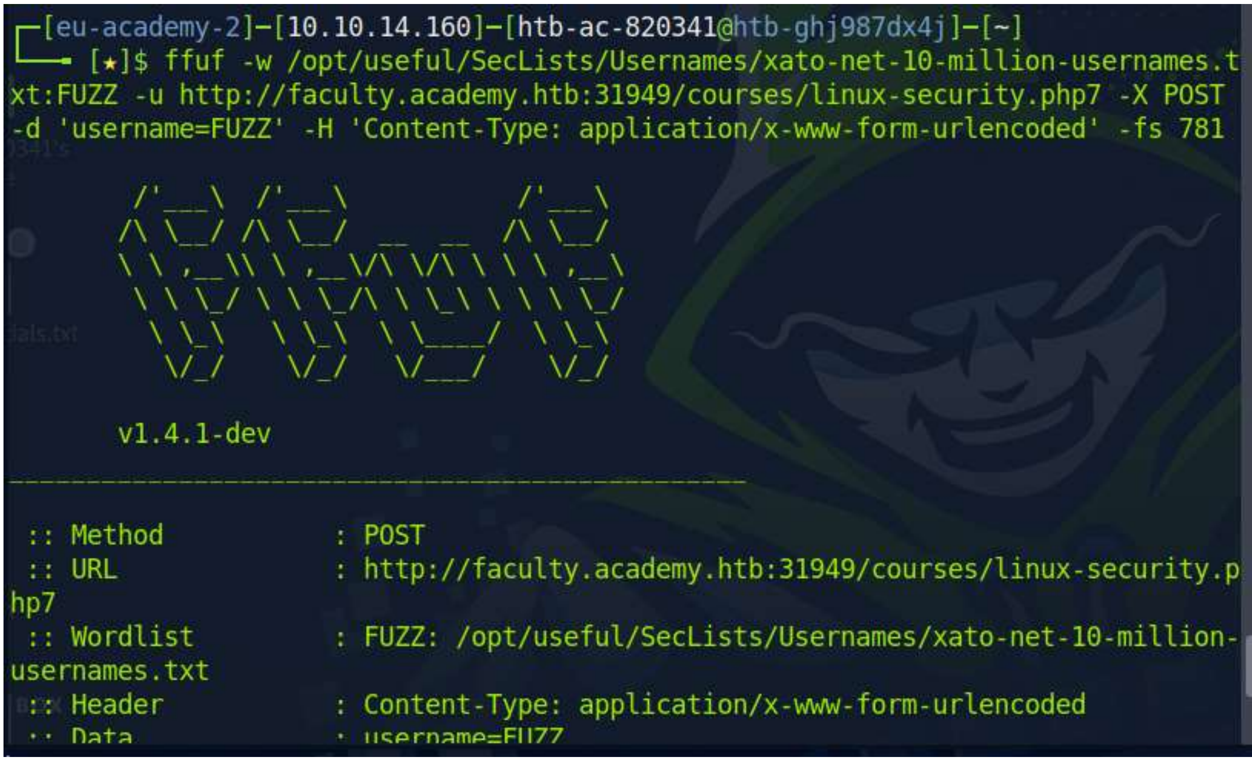
The final question requires value fuzzing for the parameters obtained above. Here we are going to try seclists that help us in brute forcing. The following two commands will help us in solving the question.

- ffuf -w /opt/useful/SecLists/Usernames/xato-net-10-million-usernames.txt:FUZZ -u http://faculty.academy.htb:30401/courses/linux-security.php7 -X POST -d 'username=FUZZ' -H 'Content-Type: application/x-www-form-urlencoded'

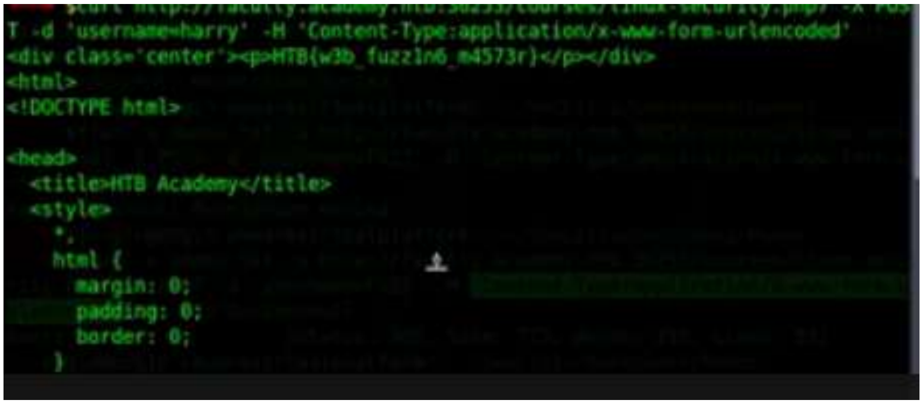


```
┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx4j]─[~]
└──╼ [★]$ ffuf -w /opt/useful/SecLists/Discovery/Web-Content/burp-parameter-name
s.txt:FUZZ -u http://faculty.academy.htb:30796/courses/linux-security.php7 -X PO
ST -d 'FUZZ=key' -H 'Content-Type: application/x-www-form-urlencoded' -fs 774

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
________________________________________________

 :: Method           : POST
 :: URL              : http://faculty.academy.htb:30796/courses/linux-security.p
hp7
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/burp-par
ameter-names.txt
 :: Header           : Content-Type: application/x-www-form-urlencoded
 :: Data             : FUZZ=key
```

- ffuf -w /opt/useful/SecLists/Usernames/xato-net-10-million-usernames.txt:FUZZ -u http://faculty.academy.htb:30401/courses/linux-security.php7 -X POST -d 'username=FUZZ' -H 'Content-Type: application/x-www-form-urlencoded' -fs 781



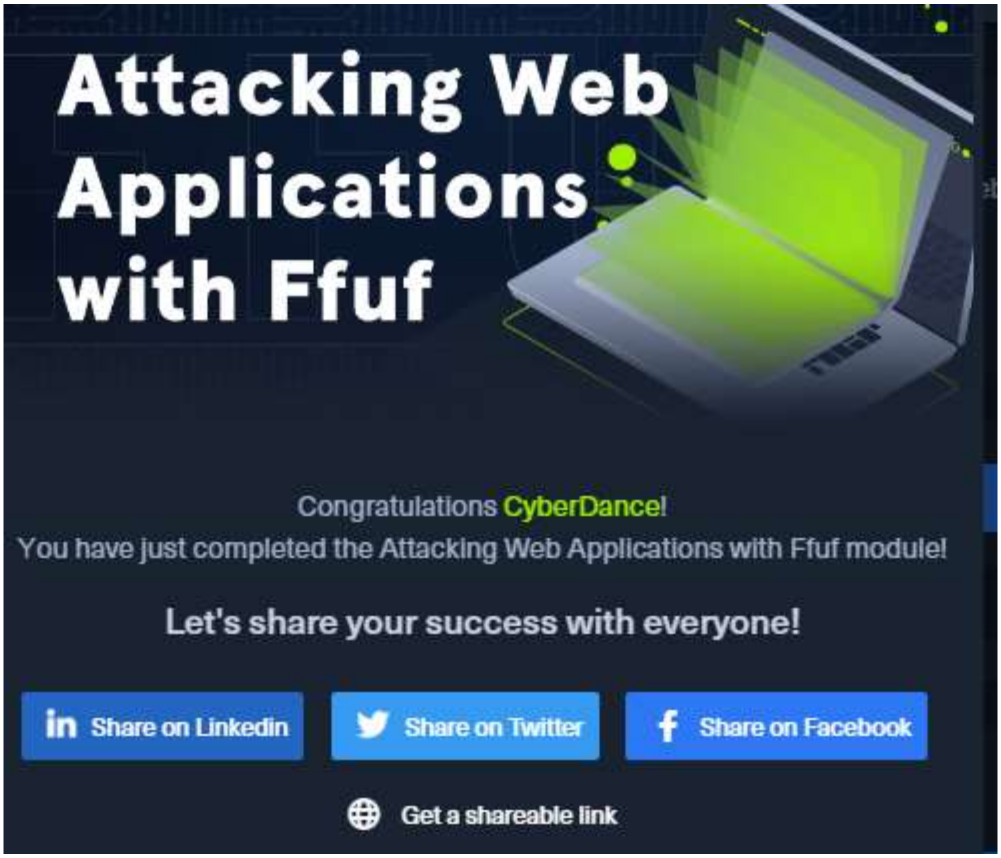
```
┌─[eu-academy-2]─[10.10.14.160]─[htb-ac-820341@htb-ghj987dx4j]─[~]
└──╼ [★]$ ffuf -w /opt/useful/SecLists/Usernames/xato-net-10-million-usernames.t
xt:FUZZ -u http://faculty.academy.htb:31949/courses/linux-security.php7 -X POST
-d 'username=FUZZ' -H 'Content-Type: application/x-www-form-urlencoded' -fs 781

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
________________________________________________

 :: Method           : POST
 :: URL              : http://faculty.academy.htb:31949/courses/linux-security.p
hp7
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Usernames/xato-net-10-million-
usernames.txt
 :: Header           : Content-Type: application/x-www-form-urlencoded
 :: Data             : username=FUZZ
```



```
$curl http://faculty.academy.htb:30233/courses/linux-security.php7 -X POS
T -d 'username=harry' -H 'Content-Type:application/x-www-form-urlencoded'
<div class='center'><p>HTB{w3b_fuzz1n6_m4573r}</p></div>
<html>
<!DOCTYPE html>

<head>
  <title>HTB Academy</title>
  <style>
    *,
    html {
      margin: 0;
      padding: 0;
      border: 0;
    }
```

Here is a completion screenshot for this module and a sharable link:

Link: https://academy.hackthebox.com/achievement/820341/54



Congratulations CyberDance!
You have just completed the Attacking Web Applications with Ffuf module!

Let's share your success with everyone!

Conclusion

In conclusion, this module on attacking web applications with ffuf has provided a valuable learning experience. We have acquired practical knowledge of using ffuf to fuzz directories, pages, sub-domains, parameters, and values. Through this hands-on exploration, we have developed a better understanding of how attackers can identify vulnerabilities and exploit weaknesses in web applications. This module has enhanced our skills in conducting comprehensive security assessments and testing the robustness of web applications. Overall, the knowledge gained and the experience gained from completing this module will undoubtedly contribute to our expertise in web application security.