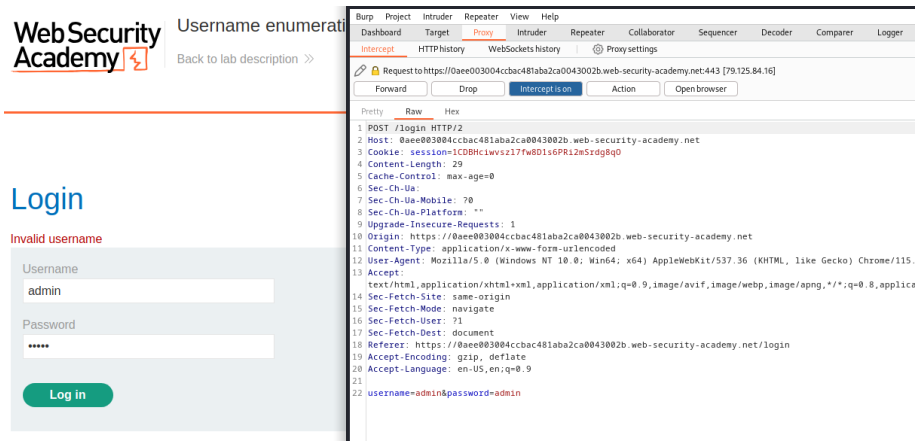
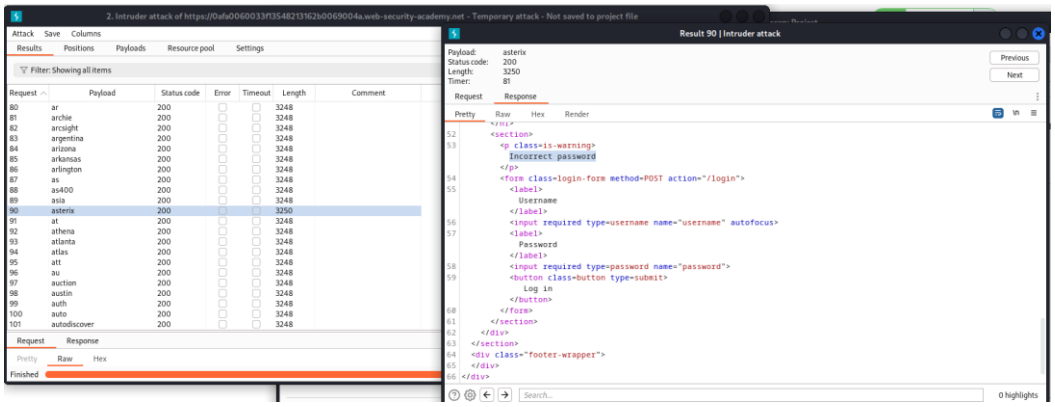
Vulnerabilities in password-based login

LAB 18 [Username enumeration via different responses]

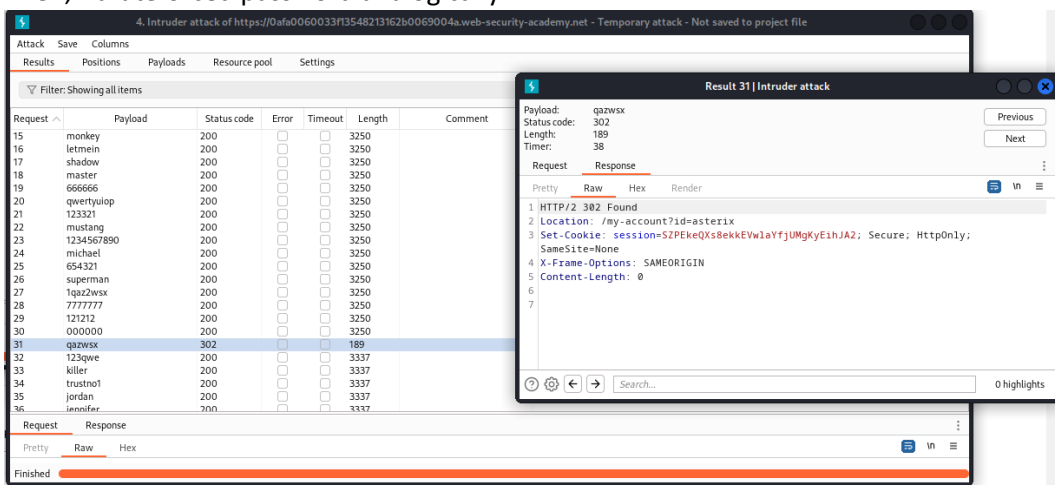
I have noticed, that log in credentials are passed in HTTP POST request in plain text, which is vulnerable to brute forcing:



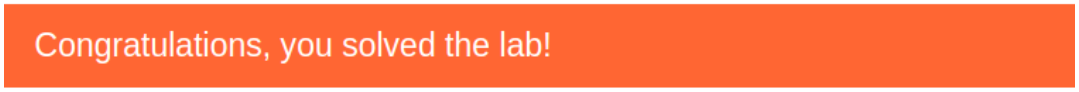
Next, I sent the request to the Intruder and launched a Sniper brute force dictionary attack:



From response length I can see that username ‘asterix’ may be correct, as the erorr in response is ‘Incorrect password’.
Then, I bruteforced password analogically:

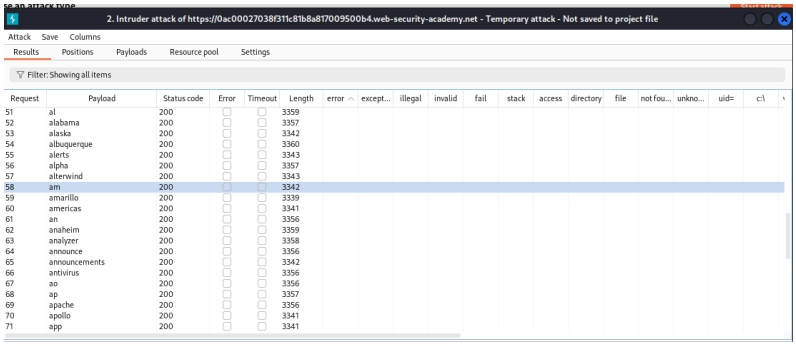


USERNAME: asterix | PASSWORD: qazwsx



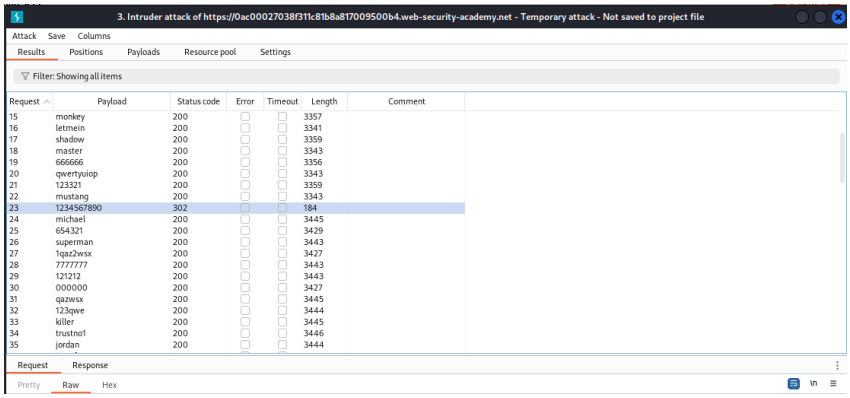
Log in successful. Lab is Done!

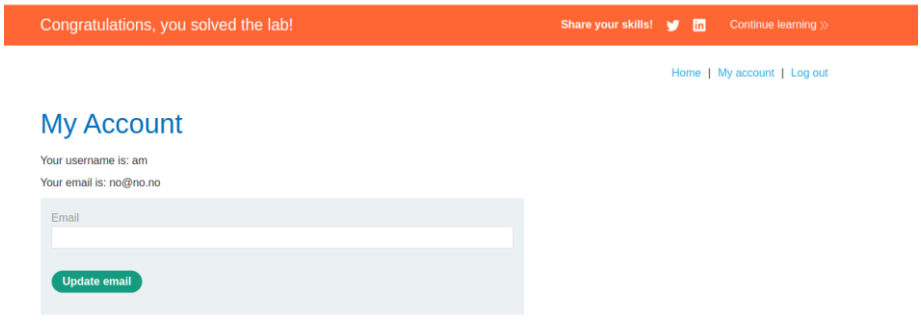
LAB 19 [Username enumeration via subtly different responses]



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:

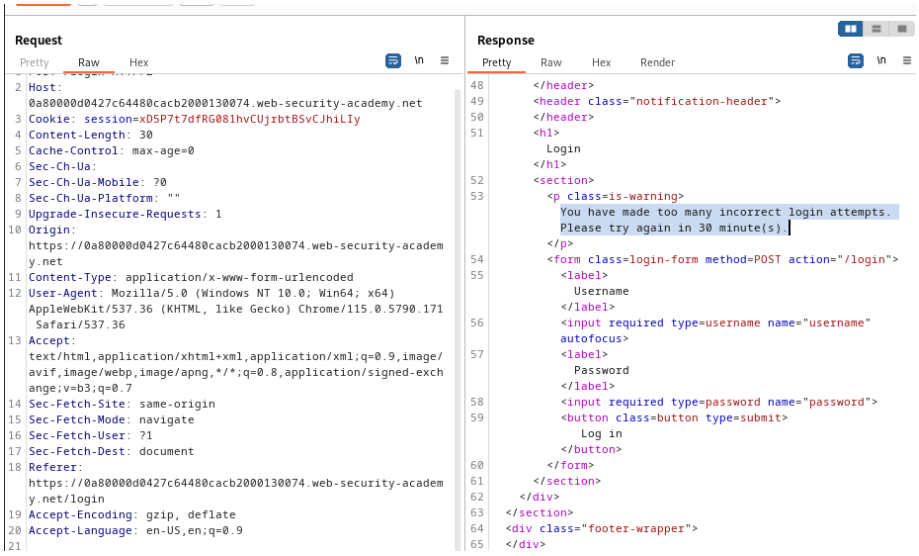




Lab’s done!

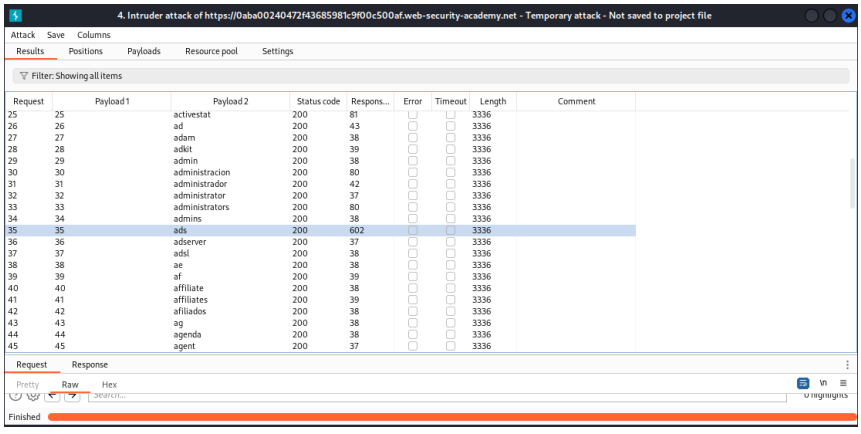
LAB 20 Username enumeration via response timing

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

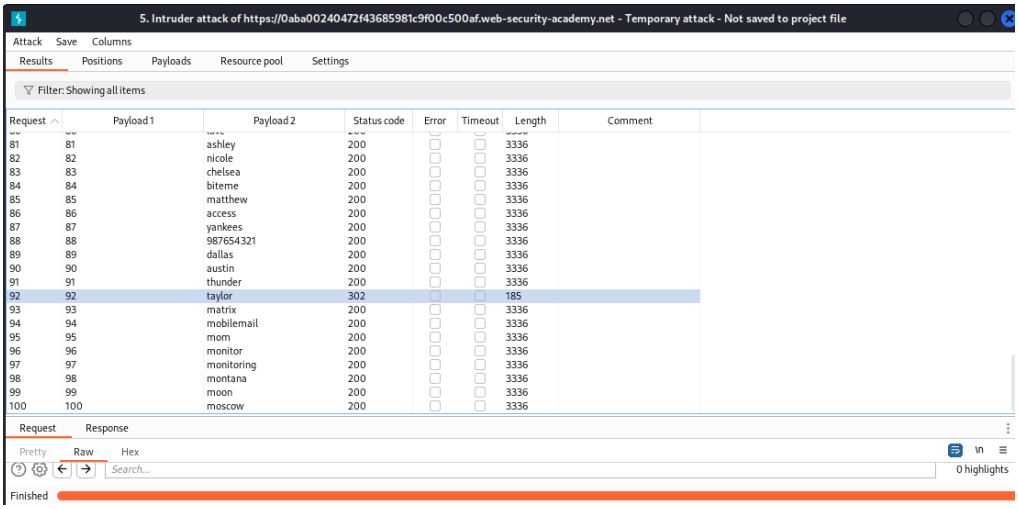


To bypass this, I can try to use HTTP header X-Forwarded-For.

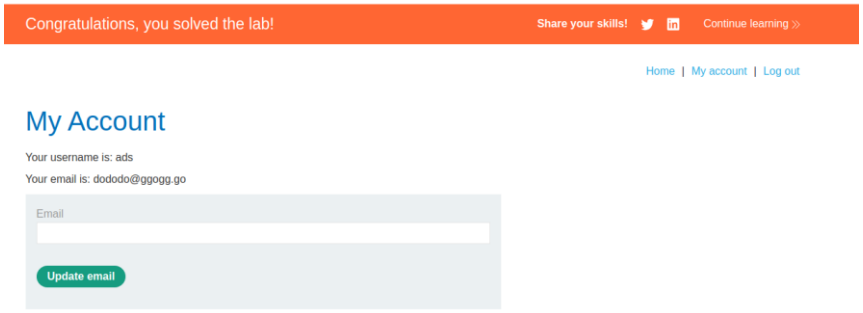
In task description I was given a valid username and password -- wiener:peter. I tried to check how the application behaves with valid credentials. If username is correct, then response time depends on password length. So, idea will be in bruteforcing the password with a very long password, that will 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 esponse time will indicate me at a valid username:



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



Password is ‘taylor’



Lab’s solved!