**SCENARIO**

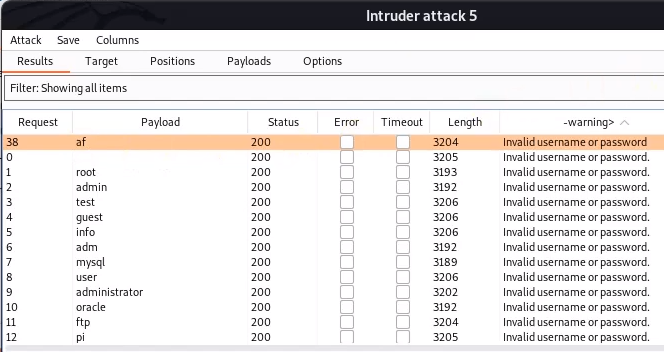
This lab is subtly vulnerable to username enumeration and password brute-force attacks. It has an account with a predictable username and password that can be identified using provided wordlists: Candidate usernames and Candidate passwords. The objective is to enumerate a valid username, brute-force this user's password, and then access their account page.

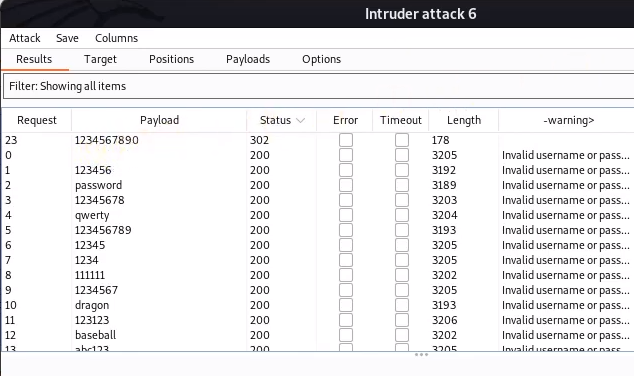
**PROCEDURE**

1. Start Burp and submit an invalid username and password combination in the lab.
2. In the POST /login request, highlight the username parameter and send it to Burp Intruder.
3. Navigate to the Payloads tab. The username parameter should be automatically marked as a payload position. Ensure "Simple list" payload type is selected and load the list of candidate usernames.
4. In the Settings tab, under "Grep - Extract," click "Add". Scroll to find the error message "Invalid username or password." in the response. Highlight the text content of this message and confirm. Start the attack.
5. Once the attack completes, review the results. One error message might slightly differ due to a typo (e.g., a space instead of a full stop). Identify this unique username.
6. Head back to the Positions tab. Insert the discovered username and add a payload marker to the password parameter: username=identified-user&password=§invalid-password§.
7. On the Payloads tab, clear the usernames list and load the candidate passwords list. Initiate the attack.
8. Post-attack, identify the request with a 302 response. This indicates a successful login. Note down this password.
9. Use the discovered username and password to log in to the lab and access the user's account page.

**PAYLOAD**

1. List of candidate usernames for enumeration.
2. List of candidate passwords for brute-force attack.

**PROOF OF CONCEPT  
**

****

**REMEDIATION**

1. **Consistent Error Messages:** Ensure error messages are consistent regardless of the nature of the login error to prevent username enumeration.
2. **Account Lockout:** Implement account lockout policies to prevent brute-force attacks.
3. **Captcha Implementation:** Introduce a CAPTCHA after a set number of failed login attempts to hinder automated attacks.
4. **Rate Limiting:** Apply rate limiting to the login endpoint to slow down brute-force attempts.