**SCENARIO**

This lab allows users to stay logged in even after they close their browser session. The cookie used for this functionality is vulnerable to brute-forcing. The goal is to brute-force Carlos's cookie to access his "My account" page.

Your credentials: wiener:peter

Victim's username: carlos

Candidate passwords: [Not Provided]

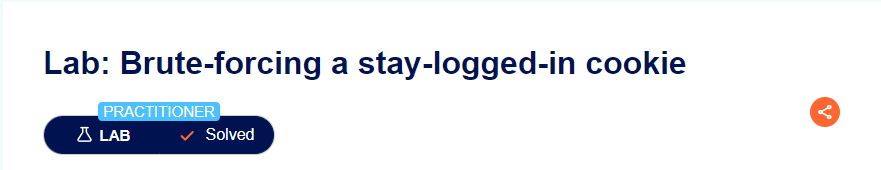
**PROCEDURE**

1. Start with Burp running and log into your own account with the "Stay logged in" option checked. This action sets a 'stay-logged-in' cookie.
2. Examine the cookie in the Inspector panel, noticing it's Base64-encoded. Decode to find the format wiener:51dc30ddc473d43a6011e9ebba6ca770, indicating a possible MD5 hash of the password.
3. Confirm the MD5 hash assumption by hashing your password using MD5. Discover the cookie format as base64(username+':'+md5HashOfPassword).
4. Log out from your account.
5. For the most recent GET /my-account, highlight the 'stay-logged-in' cookie parameter and forward the request to Burp Intruder.
6. In Burp Intruder, find that the cookie is a payload position. Add your password as the payload.
7. Under Payload processing, sequentially apply the following rules:
8. Hash: MD5
9. Add prefix: wiener:
10. Encode: Base64-encode
11. Use the presence of the "Update email" button, only shown in an authenticated state, as an indicator for successful brute-forcing. Add a grep match rule for this string.
12. Start the attack. Note that one payload successfully loads your account page, verifying that the rules created a valid cookie for your account.
13. Modify for the next attack: replace your password in the payload list with the candidate passwords, and adjust the prefix rule to carlos:.
14. Start the attack again. When the attack completes, the lab should be solved. Only one response should contain "Update email", indicating the correct 'stay-logged-in' cookie for Carlos's account.

**PAYLOAD**

Base64 encoding of the format: username:MD5HashOfPassword.

**PROOF OF CONCEPT**



**REMEDIATION**

1. **Enhance Cookie Security:** Utilize stronger and more complex algorithms instead of MD5 for hashing.
2. **Rate Limiting:** Implement rate limiting on login and cookie verification attempts to prevent brute force attacks.
3. **Cookie Expiration:** Ensure session cookies expire after a certain period of inactivity, reducing the window of opportunity for attacks.
4. **Multi-Factor Authentication:** Introduce additional layers of authentication for critical account actions.