

Forum:[Hack Forums](#)

Thread: [Password Reset Poisoning: Intercepting and Manipulating Email Links](#)

#AfterDark - August 27, 2024, 4:28 pm

Password Reset Poisoning: Intercepting, Manipulating Email Links

This tutorial demonstrates how to exploit vulnerabilities in the password reset process by intercepting and manipulating email links using BurpSuite. This can allow an attacker to reset a user's password and gain unauthorized access to their account. This guide is intended for educational purposes to enhance your penetration testing skills as part of OSCP training

Step-by-Step Guide

Prerequisites

A vulnerable web application with a password reset feature.

BurpSuite installed and configured to intercept HTTP/HTTPS traffic.

An email client to receive password reset emails.

Configuring BurpSuiteLaunch BurpSuite and navigate to the "Proxy" tab.

Click on "Intercept" to ensure it is turned on.

Configuring Browser to interact with BurpSuiteOpen your browser settings and configure the proxy settings to use 127.0.0.1 and port 8080 (default BurpSuite settings).

Ensure you have installed BurpSuite's CA certificate in your browser to avoid SSL/TLS issues. This can be done by going to BurpSuite's "Proxy" > "Options" > "Import / Export CA Certificate"

Initiating the Password Reset Process

Trigger Password Reset

Navigate to the Target Web Application's Password Reset Page:

Locate the "Forgot Password" or equivalent link on the login page.

Enter the Email Address:

Use the email address of the account you want to reset the password for.

Submit the Form:

Click on the "Reset Password" or equivalent button to submit the request.

Capture the Request

Intercept the Request in BurpSuite:Ensure interception is turned on in BurpSuite. When you submit the form, BurpSuite will capture the HTTP request.

Analyze the Request:

Examine the request headers and parameters to understand how the password reset process works. For example:

```
Code `POST /reset_password HTTP/1.1 Host: example.com Content-Type: application/x-www-form-urlencoded Content-Length: 40`
```

email=user@example.com&action=reset`

Look for parameters like email or user_id.

Forward the request to allow the email to be sent to the target's email address.

Intercepting the Password Reset Email

Receive the Email

Log in to the email account used for the password reset request.

Look for the password reset email. It typically contains a link or token for resetting the password, such as:

Copy the URL or token provided in the email.

Code `https://example.com/reset_password?token=abcd1234`

Capture the Reset Link

Intercepting and Modifying the Reset Link Request
Paste the copied URL into your browser but do not press Enter yet.

Turn On Interception in BurpSuite:

Ensure BurpSuite's interception is enabled to capture the request.

Press Enter to navigate to the reset link.

BurpSuite will capture the HTTP request containing the reset token or parameters, such as:

Code `GET /reset_password?token=abcd1234 HTTP/1.1 Host: example.com`

Look for parameters like token, reset_token, email, or user_id.

Identify which parameter controls the reset functionality.

Modify the Parameters as Needed:

Example 1: Change the token to another valid token if you have multiple accounts. Let's assume you have another valid token efgh5678:

Code `GET /reset_password?token=efgh5678 HTTP/1.1 Host: example.com`

Example 2: Change the email parameter to another valid email. Suppose the original request was:

Code `GET /reset_password?token=abcd1234&email=user@example.com HTTP/1.1 Host: example.com`

You could change the email to an attacker-controlled email:

Code `GET /reset_password?token=abcd1234&email=attacker@example.com HTTP/1.1 Host: example.com`

Understand the Application's Logic:

Analyze how the application validates the reset token. This can involve replaying the same request with slight modifications to see how the server responds.

Check if the token is validated against the user's session or if it's independent. For example, some applications may bind the token to a specific IP address or session ID

Forward the Modified Request: Send the modified request to the server by clicking the "Forward" button in BurpSuite.

Observe the server's response to confirm if the modification was successful.
Follow the Redirect
If the modified request is accepted, you will typically be redirected to a page where you can set a new password.
Enter and Confirm the New Password:
Enter a new password and confirm it to complete the reset process.
Log in to the account using the new password to verify that the reset was successful.

Additional Examples and Techniques

Example: Exploiting Parameter Manipulation

Suppose the reset link contains multiple parameters including a user identifier:

https://example.com/reset_password?token=abcd1234&user_id=123

You could change the user_id parameter to another valid user ID to see if the application allows resetting another user's password:

https://example.com/reset_password?token=abcd1234&user_id=456

Example: Using BurpSuite Repeater for Token Manipulation Use BurpSuite's Repeater tool to resend and modify the reset token:

Intercept the reset link request.

Send the request to Repeater by right-clicking on the request and selecting "Send to Repeater".

In the Repeater tab, modify the token or other parameters as needed.

Send the modified request repeatedly with different values to test how the application responds.

Mitigation Strategies

To prevent password reset poisoning attacks, consider implementing the following security measures:

Token Expiration: Ensure that password reset tokens have a short expiration time to limit the window of opportunity for attacks.

Secure Token Generation Use cryptographically secure methods to generate unique and random tokens to prevent prediction or reuse.

Rate Limiting: Implement rate limiting on password reset requests to prevent brute force attacks and token enumeration.

Email Confirmation: Require users to confirm their email address or other personal information before allowing a password reset to ensure the request's legitimacy.

Logging and Monitoring: Monitor and log password reset attempts and alert administrators of suspicious activity to detect and respond to potential attacks.

Use HTTPS: Ensure all communication between the client and server is encrypted to prevent interception of sensitive data.

Bind Tokens to User Sessions:

Bind password reset tokens to the user's session or IP address to prevent misuse by unauthorized parties.

Hope you enjoy this tutorial by #AfterDark

? HackthePlanet!

DramaKing - August 27, 2024, 7:02 pm

In the first example, how would changing the token for one password reset request for one attacker-controlled account to a token for another attacker-controlled account do anything useful? In the second example, are you saying that a token for one email could be valid for another? That would be pretty crazy and an easy bounty.

#AfterDark - August 27, 2024, 7:54 pm

(Aug 27, 2024 07:02 PM)DramaKing Wrote: [\[CITE\]](#) In the first example, how would changing the token for one password reset request for one attacker-controlled account to a token for another attacker-controlled account do anything useful? In the second example, are you saying that a token for one email could be valid for another? That would be pretty crazy and an easy bounty.

Example one shows the enumeration steps of identifying the parameters for the password reset. By creating both accounts you are able to intercept and manipulate cookies and token to test the backend permission of the application.

We utilize waybackurls as well to further help with this scenario.

Example two is a perfect example of broken access controls by manipulating the account id. And yes very easy bounties indeed. Got a 7.9 high bounty on Airbnb with this same exact method. ;)

Always think of bad habits and lazy developers. They make it easy for us ?