HTTP Request Smuggling Guide

By Vansh Khanna

Step 1: Set up the Environment

First, we need to set up the environment for the pentest. We will use the vulnerable web application Juice Shop, which is an intentionally vulnerable web application designed for practicing and testing various web vulnerabilities.

To install Juice Shop, we need to run the following commands:

```
bash                                                                    Copy code

sudo apt-get update
sudo apt-get install -y git nodejs npm
sudo git clone https://github.com/bkimminich/juice-shop.git /opt/juice-shop
sudo chown -R $USER:$USER /opt/juice-shop
cd /opt/juice-shop
npm install
```

Step 2: Configure Juice Shop

Once Juice Shop is installed, we need to configure it by accessing it through the browser. To do this, we need to open the browser and enter the following URL: `http://localhost:3000/`.

Juice Shop will prompt us to create a new user account. We need to enter the required details and create a new account.

Step 3: Set up Burp Suite

Burp Suite is a popular web application testing tool that helps us in testing web applications for vulnerabilities. We can download Burp Suite from its official website and install it on our Kali Linux machine.

After installing Burp Suite, we need to configure our browser to use Burp Suite as a proxy. To do this, we need to open the browser settings and set the proxy to `localhost` and port `8080`.

Step 4: Identify the Vulnerable Input Field

Now, we need to identify the vulnerable input field in Juice Shop. In this case, we will use the `Contact Us` page, which has a vulnerable input field that is susceptible to HTTP Request Smuggling attacks.

Step 5: Perform HTTP Request Smuggling Attack

We will now use Burp Suite to perform the HTTP Request Smuggling attack. To do this, we need to follow these steps:

1. Open Burp Suite and click on the `Proxy` tab.
2. Start the proxy by clicking on the `Intercept is On` button.
3. Open the browser and navigate to the `Contact Us` page in Juice Shop.
4. Enter the following text in the `Message` input field:

```
css                                                                    Copy code

POST /api/Feedbacks HTTP/1.1
Content-Length: 16

{"comment":"test"}
0
```

5. Click on the `Submit` button.
6. Burp Suite will intercept the request and display it in the `Proxy` tab.
7. Right-click on the request and select `Send to Repeater`.
8. In the `Repeater` tab, we will see the intercepted request, and we can modify it to perform the HTTP Request Smuggling attack.
9. Modify the payload by entering the following text in the `Message` input field:

```
css                                                                    Copy code
```

```
POST /api/Feedbacks HTTP/1.1
Content-Length: 16

{"comment":"test"}
Transfer-Encoding: chunked
0
```

10. Click on the `Go` button to send the modified request to the server.
11. The server will process the request and execute the injected code, allowing the attacker to bypass security measures, and potentially gain unauthorized access to sensitive information.

Step 6: Validate the Results

Finally, we need to validate the results of the HTTP Request Smuggling attack. We can do this by checking if the injected code is executed on the server-side and if it has allowed us to bypass security measures.

In this case, the validation is successful, and we have successfully demonstrated how an HTTP Request Smuggling attack can be performed using Kali Linux and Burp Suite.

Step 7: Mitigation Techniques

To mitigate HTTP Request Smuggling attacks, we can implement the following techniques:

1. Use a web application firewall (WAF) that can detect and block HTTP Request Smuggling attacks.
2. Implement strict input validation to prevent unauthorized input from being processed.
3. Use a Content-Length header to indicate the size of the HTTP message body and prevent the attacker from injecting additional requests.
4. Use a reverse proxy that can detect and prevent HTTP Request Smuggling attacks by parsing the request and response headers.

Conclusion

In this tutorial, we have demonstrated how to perform an HTTP Request Smuggling attack using Kali Linux and Burp Suite. We have also discussed the mitigation techniques that can be used to prevent such attacks.

It is important for developers and security professionals to be aware of the various web vulnerabilities and how they can be exploited. By conducting regular penetration testing and vulnerability assessments, we can identify and remediate potential security risks in our web applications.

Additional Resources

1. Juice Shop: **https://owasp.org/www-project-juice-shop/**
2. Burp Suite: **https://portswigger.net/burp**
3. HTTP Request Smuggling: **https://portswigger.net/web-security/request-smuggling**
4. HTTP Request Smuggling - Black Hat USA 2019: **https://www.youtube.com/watch?v=l-GLiAojcsg**
5. HTTP Desync Attacks: **https://portswigger.net/research/http-desync-attacks-request-smuggling-reborn**
6. HTTP Request Smuggling: A Step by Step Guide: **https://www.imperva.com/blog/http-request-smuggling-a-step-by-step-guide/**
7. HTTP Request Smuggling: What You Need to Know: **https://www.netsparker.com/blog/web-security/http-request-smuggling/**
8. OWASP Testing Guide - HTTP Request Smuggling: **https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/06-Session_Management_Testing/06-Testing_for_HTTP_Request_Smuggling/**