Network Security

Web Exploitation and Security

1 Introduction

In this assignment we will explore how web applications can be exploited and defended. We will be studying the 4 major aspects of web security: SQL Injection, Cross-Origin resource Sharing, Cross-Origin Request Forgery and Cross-site Scripting. We will be using labs from Portswigger.

2 Setting up a Virtual Machine and signing up for the Labs

2.1 Step 1 - Setting up the Kali VM

You can use Burp Suite outside your VM. Setting up a VM is not mandatory. But if you feel the need to isolate your use of Burp Suite then you can set up Kali as explained below.

Download VirtualBox from here. Install it with the additional VirtualBox Extension Pack which will help you resize your VMs according to the Window size. Make sure the Virtual Machine is using a NAT network. You can find the Kali VM ISO here. The Kali virtual machine creates a Type 2 virtualized system for users to use without having to install additional resources. It is one of the best offensive security operating systems used by security researchers.

2.2 Step 2 - Signing up for the Lab

Inside the VM browse to Portswigger Web Academy and sign up using your NYU account. Once the account is created browse to the detailed page view.

2.3 Format of the report expected

Students are expected to submit the outcome of each web exploitation detailed in this lab, using the format as shown below.

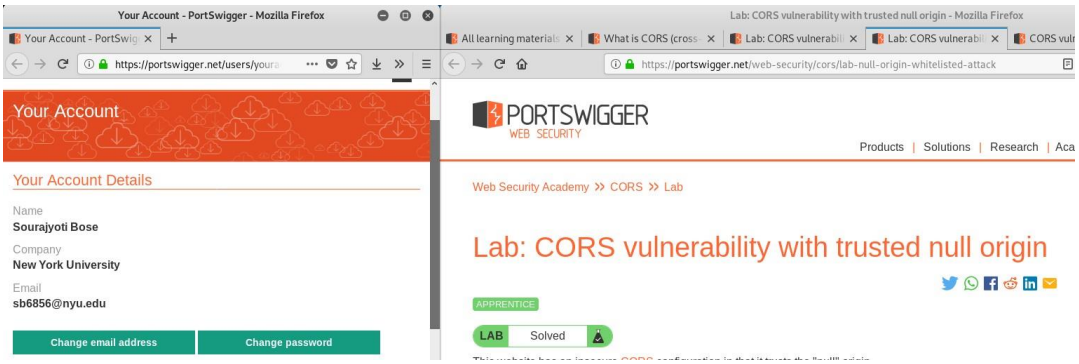

Figure 1: Screenshot format

The screenshots should always have the account window present in it along with the Lab name and the solved green tag.

3 Burp Suite

Burp Suite is an integrated platform for performing security testing of web applications. Its various tools work seamlessly together to support the entire testing process, from initial mapping and analysis of an application's attack surface, through finding and exploiting security vulnerabilities.

Burp gives you full control, letting you combine advanced manual techniques with state-of-the-art automation, to make your work faster, more effective, and more fun.

Here is the link to learn more about Burp Suite. All our assignments depend on Burp Suite as it is a major tool used for finding flaws in web applications.

We focus on these features of Burp Suite for the labs being performed in this Lab assignment.

- Burp Intercept
- Burp Intruder
- Burp Repeater

Please familiarize yourself with these functionalities for Burp Suite before starting the labs. Use the Burp Suite present in the Kali Virtual Machine.

4 Cross-origin Resource Sharing (CORS)

All major web applications have 2 major parts, the backend and the frontend. Both are undoubtedly hosted and served on 2 different servers which will lead to requesting of resources from one server to another whose origins are different. To allow resources to be used across servers, HTTP headers are used as indicators. You can read about CORS here.

Most of the times, poor configuration leads to CORS attacks and that is explained in detail here.

4.1 Topics to be focused on

- **Same-origin policy** - You should be able to distinguish between same origin and cross origin resources.
- **Access Control Allowed Origins (ACAO)** - CORS configuration depends on ACAO HTTP header.

4.2 Labs to be performed

- CORS vulnerability with basic origin reflection **(5 points)**
- CORS vulnerability with trusted null origin **(8 points)**

4.3 Additional Reading

- https://looker.com/blog/iframe-sandbox-tutorial

5 SQL Injection

SQL injection is a code injection technique, used to attack data-driven applications, in which nefarious SQL statements are inserted into an entry field for execution. We will be performing this attack on the Metasploitable server. SQL injection is the top security vulnerability listed in the OWASP top ten.

5.1 Topics to be focused on

- Retrieving data that is being hidden in the front end
- Union attacks - Figuring out how to execute additional SQL queries
- Second-order SQL Injection - Developers hiding known SQL injection vulnerabilities.

5.2 Labs to be performed

- SQL injection vulnerability in WHERE clause allowing retrieval of hidden data (**5 points**)
- SQL injection vulnerability allowing login bypass (**5 points**)
- SQL injection UNION attack, determining the number of columns returned by the query (**8 points**)
- SQL injection UNION attack, retrieving multiple values in a single column (**8 points**)
- Blind SQL injection with time delays and information retrieval (**12 points**)

5.3 Extra Credit

Here the given solution asks the user to use Burp Suite Collaborator which is not available for Community Edition. So, you will have to use a workaround.

Hint: **Use the domain name www.boxangular.com**

- Blind SQL injection with out-of-band interaction (**10 points**)

For the extra credit write a small explanation (2-3 lines will suffice) as to how your attack was modified.

5.4 Additional Reading

- Magento SQL Injection Attack
- Starbucks SQL Injection Attack

6 Cross-site Scripting

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user' s browser has no way to know that the script should not be trusted and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.

6.1 Topics to be focused on

- **Reflected XSS** - The JavaScript script is included in the current HTTP request and response.
- **Stored XSS** - The script is executed from the storage.
- **DOM-based XSS**
- **Cross-site scripting context**

6.2 Labs to be performed

- Reflected XSS into HTML context with nothing encoded (**2.5 points**)
- Stored XSS into HTML context with nothing encoded (**2.5 points**)
- Stored XSS into anchor href attribute with double quotes HTML-encoded (**4 points**)
- Reflected XSS into a JavaScript string with angle brackets and double quotes HTML-encoded and single quotes escaped (**5 points**)
- Reflected XSS into a template literal with angle brackets, single, double quotes, backslash and backticks Unicode-escaped (**5 points**)
- DOM XSS in document. Write sink using source location. Search inside a select element (**8 points**)
- DOM XSS in innerHTML sink using source location.search (**5 points**)

6.3 Extra Credit

- Angular JS XSS attack (**10 points**)

6.4 Additional Reading

- Sinks used for Dom Based XSS attacks
- Testing for XSS
- Sink GitHub Readme - This resource is old, but it highlights common ways to defining sources of attacker-controlled inputs and sinks which potentially could introduce DOM Based XSS issues.

7 Cross-site Resource Forgery

7.1 Topics to be focused on

We will examine the multiple ways to exploit a web application using Cross-Site Request Forgery. CSRF is a web security vulnerability that allows an attacker to induce users to perform actions that they do not intend to perform.

7.2 Labs to be performed

- CSRF where token validation depends on request method (**5 points**)
- CSRF where token validation depends on token being present (**5 points**)
- CSRF where token is tied to non-session cookie (**7 points**)

7.3 Additional Reading

- CSRF token with express
- CSRF support with Spring Boot
- https://www.youtube.com/watch?v=vRBihr41JTo