Advent of Cyber 2025 Day 11 Walkthrough XSS Merry XSSMas

Learn about types of XSS vulnerabilities and how to prevent them.

Learning objective

- Understand how XSS works
- Learn to prevent XSS attacks

Introduction

Cross site scripting (XSS) is a web application vulnerability which attacker use to inject malicious code into input field that reflect content viewed by other users. The main aim of this attack is to steal credentials, deface pages or impersonate users.

There are various type of XSS, we will focus on reflected XSS and Stored XSS.

Reflexted XSS

we see reflected variant when the injection is immediately projected in a response.

Example : Imagine search function in a online toy store
`https://trygiftme.thm/search?term=gift`

but when we send to someone
`https://trygiftme.thm/search?term=<script>alert( atob("VEhNe0V2aWxfQnVubnl9") )</script>`

If someone click the link, it will execute code instead.
we could act, view informations or modify information that user could do, view or access it.

Stored XSS

This occur when malicious script is saved on the server and then loaded for every users who view the affected page. It becomes a "set and forget" attack, anyone who loads the page runs the attacks script.

To make it more clear we will take a example of a submit comment post.

Normal comment submission

```
OST /post/comment HTTP/1.1 Host: tgm.review-your-gifts.thm postId=3 name=Tony Baritone email=tony@normal-person-i-swear.net comment=This gift set my carpet on fire but my kid loved it!
```

The server stored this information and display it when someone visit.

Malicious Comment Submission (Stored XSS Example)

if the application doesn't sanitise or filter input, an attacker can submit javascript instead of comment

```
POST /post/comment HTTP/1.1 Host: tgm.review-your-gifts.thm postId=3 name=Tony Baritone email=tony@normal-person-i-swear.net comment=<script>alert(atob("VEhNe0V2aWxfU3RvcmVkX0VnZ30="))</script> + "This gift set my carpet on fire but my kid loved it!"
```

since this comment is saved in database, every user who open that blog will automatically trigger the script. This lets attacker run code as if they were the victim in order to perform malicious actions like

- steal session cookies
- Trigger fake login popups
- deface the page

Protecting against XSS

some of key practice are :

- Disable dangerous rendering raths : rather than using innerHTML property which is vulnerable to code injection use textContent because it treat input as text and parse it for HTML
- Make cookies inaccessible to JS : set session cookies with HTTPOnly , secure and SameSite attributes to reduce the impact of XSS attacks
- Sanitise input/output and encode

Exploiting Reflected XSS

We can use any text input section like search, form section for exploiting XSS vulnerabilities.

we can use test payload for checking if app run injected code . we can use cheatsheet
https://portswigger.net/web-security/cross-site-scripting/cheat-sheet
to use more advance payloads.

we will use this payload

```
<script>alert('Reflected Meow Meow')</script>
```

we can inject the code by adding the payload in search bar and search message . If output shows alert text it confirm reflected XSS.

what happen here ?

- the search input is reflected directly in the result without encoding
- the browser interprets javascript as executable code

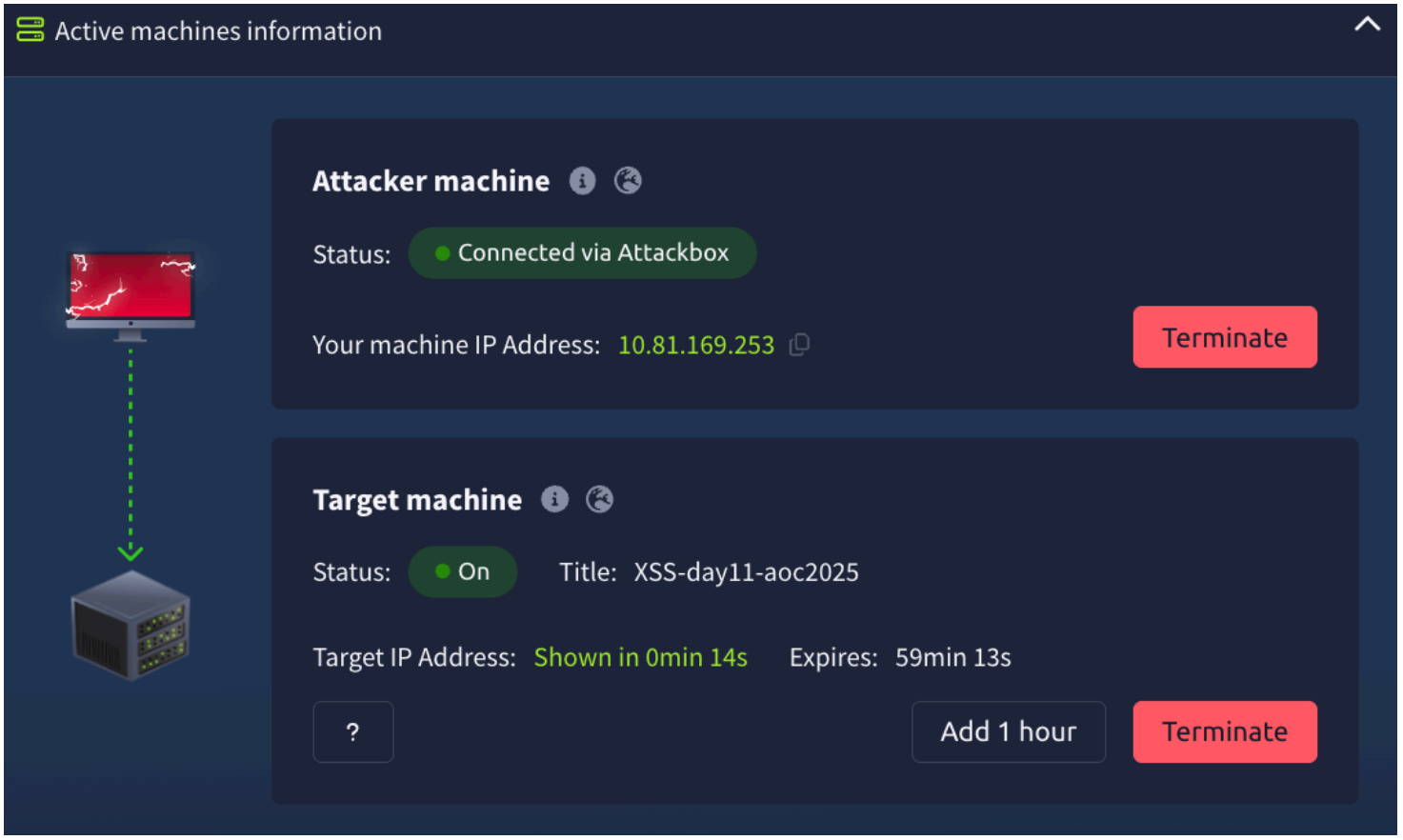
- an alert box appear, demonstrate successful XSS execution

using System Logs we can track the behaviour and see how system interpret our actions.

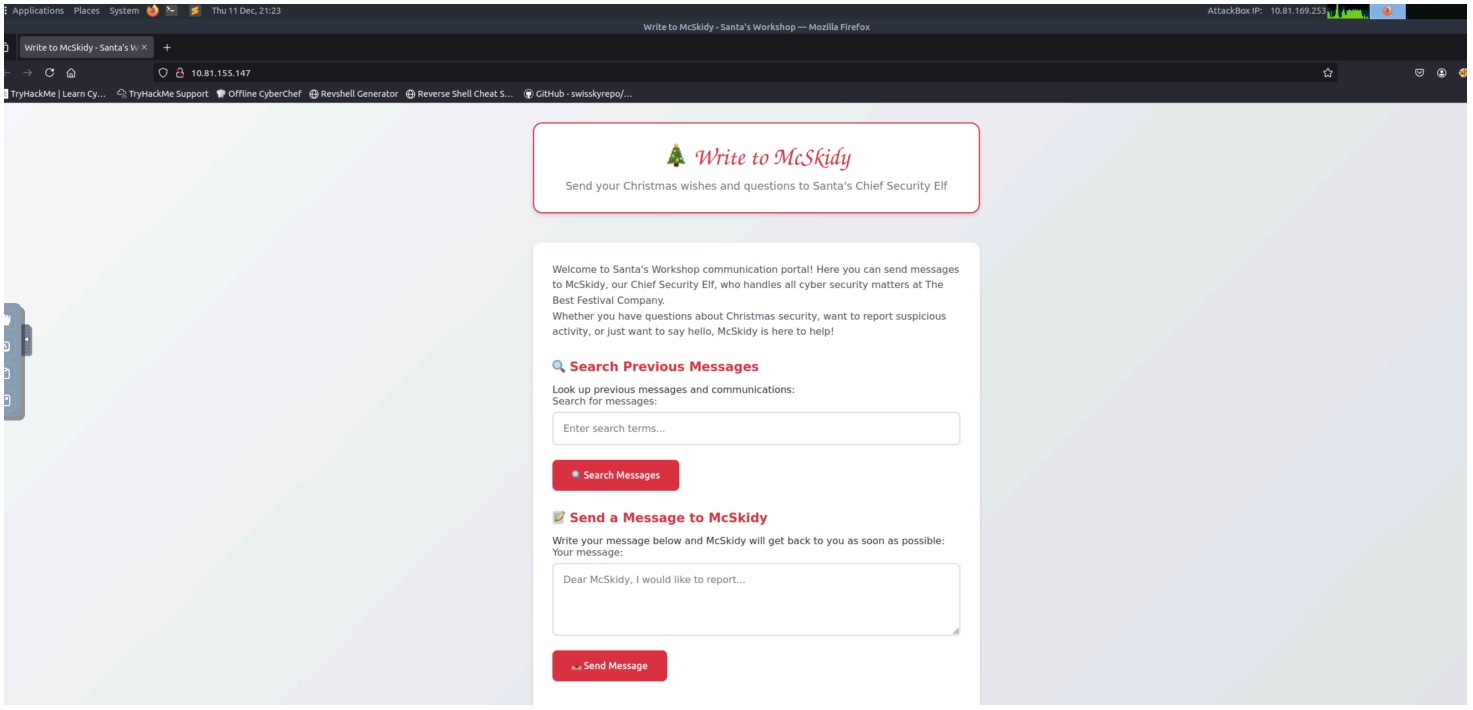
Navigate to message form and enter the malicious payload
`<script>alert('Stored Meow Meow')</script>`
click "Send Message" button, because message is stored on the server every time we navigate to the site or reload the alert will display

Lab

start machine and Attack box :

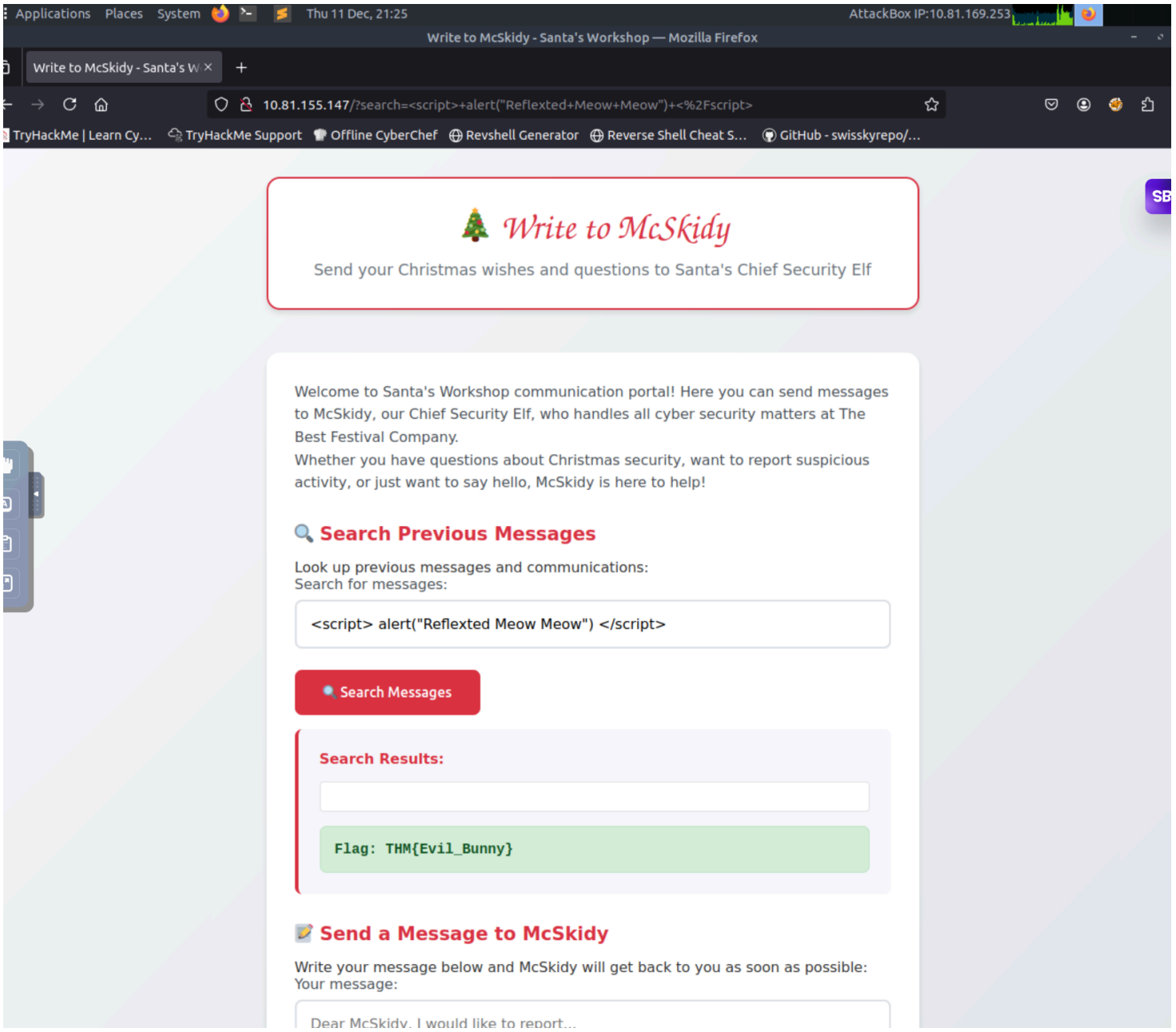


navigate to provide ip to check web app in our case its 10.81.155.147



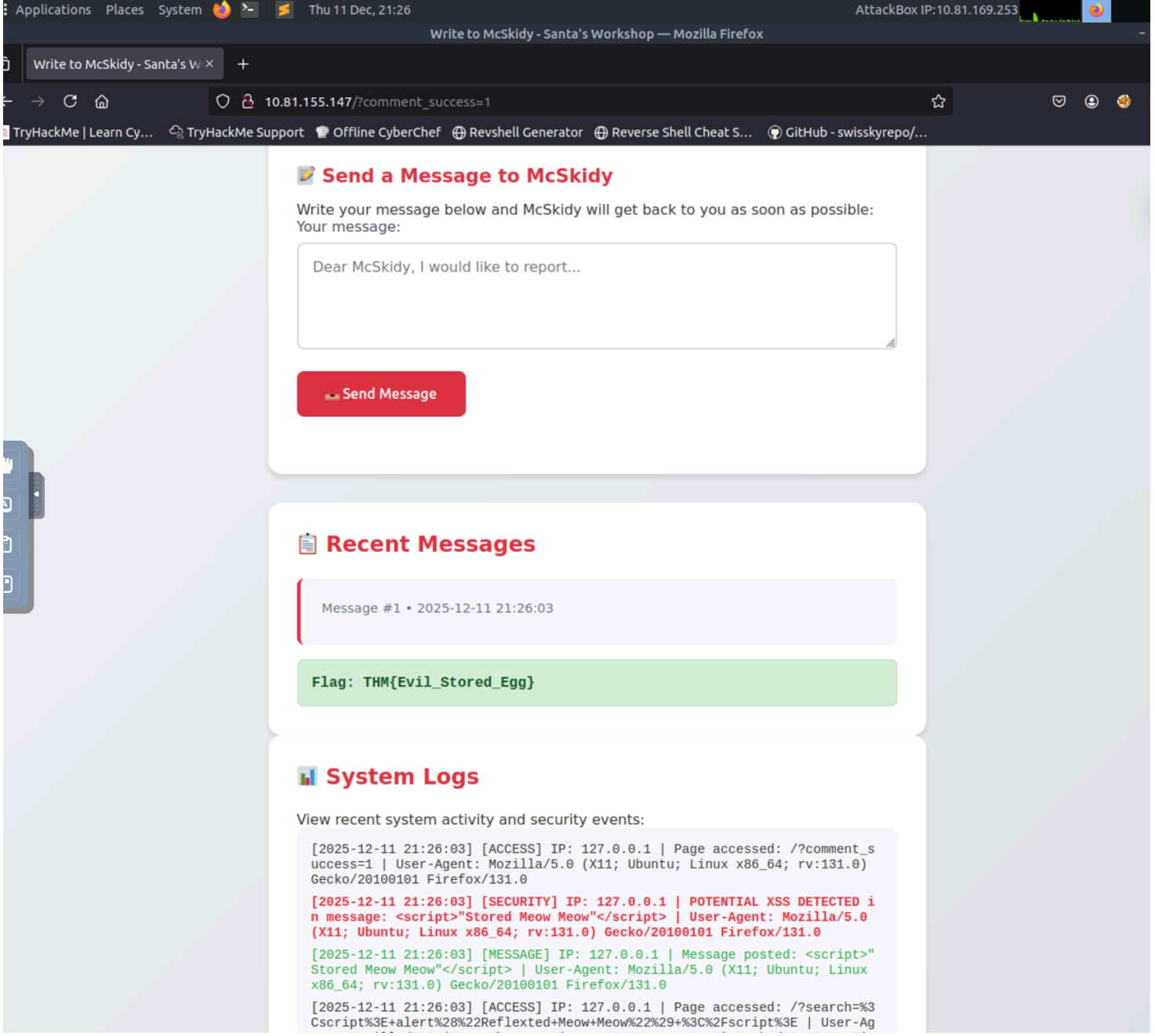
we see two text area to try our xss injection we inject `<script>alert('Reflected Meow Meow')</script>` and click on search button to capture our first flag.

Flag : THM{Evil_Bunny}

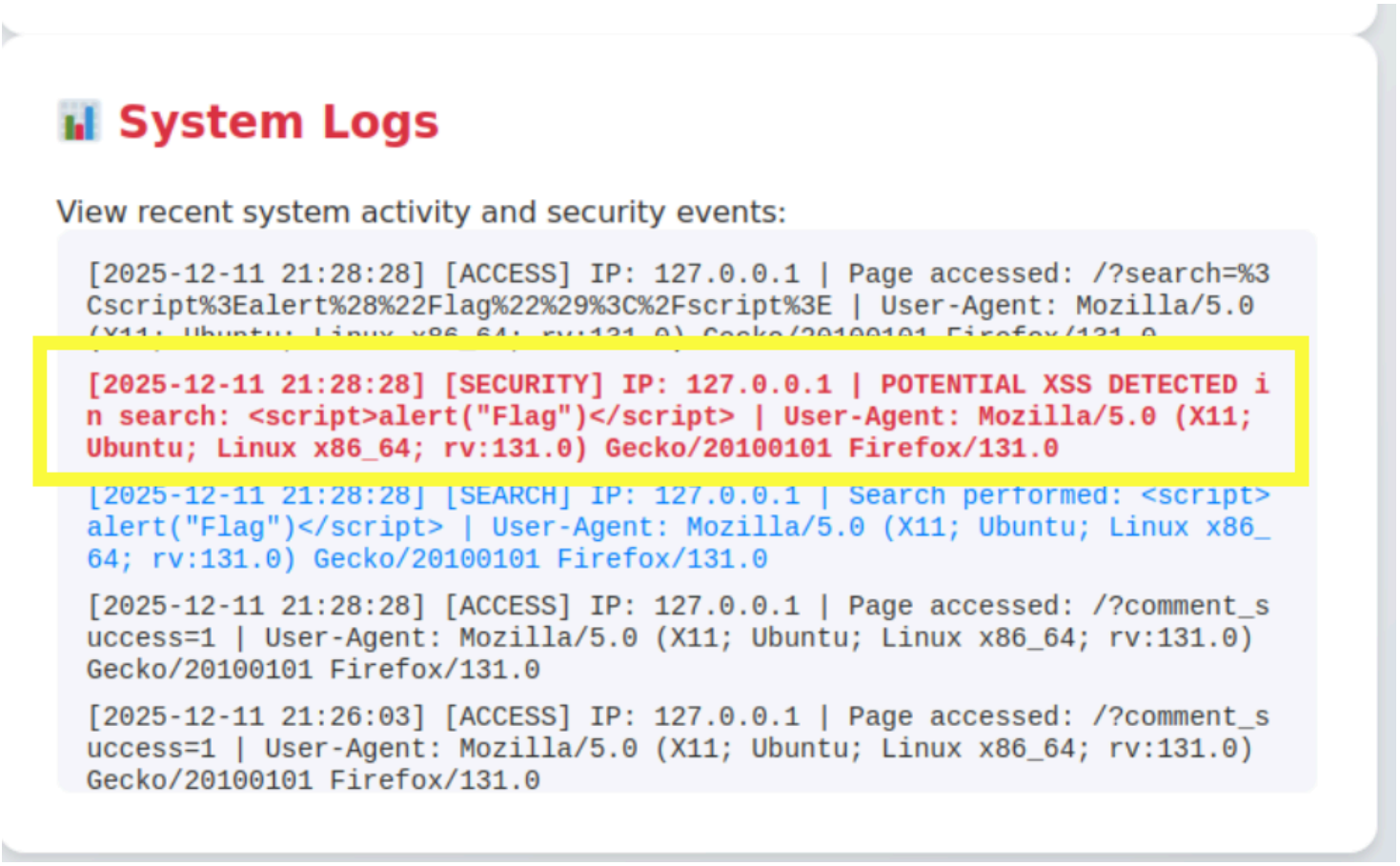


second we try for stored xss on send a message section with `<script>alert('Stored Meow Meow')</script>` it display out second flag.

Flag: THM{Evil_Stored_Egg}


Applications Places System Thu 11 Dec, 21:26 AttackBox IP:10.81.169.253
Write to McSkidy - Santa's Workshop — Mozilla Firefox
Write to McSkidy - Santa's W
10.81.155.147/?comment_success=1
TryHackMe | Learn Cy... TryHackMe Support Offline CyberChef Revshell Generator Reverse Shell Cheat S... GitHub - swisskyrepo/...
Send a Message to McSkidy
Write your message below and McSkidy will get back to you as soon as possible:
Your message:
Dear McSkidy, I would like to report...
Send Message
Recent Messages
Message #1 • 2025-12-11 21:26:03
Flag: THM{Evil_Stored_Egg}
System Logs
View recent system activity and security events:
[2025-12-11 21:26:03] [ACCESS] IP: 127.0.0.1 | Page accessed: /?comment_success=1 | User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:131.0) Gecko/20100101 Firefox/131.0
[2025-12-11 21:26:03] [SECURITY] IP: 127.0.0.1 | POTENTIAL XSS DETECTED in message: <script>"Stored Meow Meow"</script> | User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:131.0) Gecko/20100101 Firefox/131.0
[2025-12-11 21:26:03] [MESSAGE] IP: 127.0.0.1 | Message posted: <script>"Stored Meow Meow"</script> | User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:131.0) Gecko/20100101 Firefox/131.0
[2025-12-11 21:26:03] [ACCESS] IP: 127.0.0.1 | Page accessed: /?search=%3Cscript%3E+alert%28%22Reflexted+Meow+Meow%22%29+%3C%2Fscript%3E | User-Ag

we can check system logs here which show potential XSS detected in search.



Answer the questions below

1. Which type of XSS attack requires payloads to be persisted on the backend?
   Ans : stored
2. What's the reflected XSS flag?
   Ans : THM{Evil_Bunny}
3. What's the stored XSS flag?
   Ans : THM{Evil_Stored_Egg}

Completion Message

Congratulations!

You have successfully completed **Advent of Cyber 2025 Day 11 Walkthrough XSS Merry XSSMas**.