CSCI 493-81
Lab 11
Security Lab
Topic: Cross-Site Scripting (XSS), Encrypted Mail

Prof. Dietrich

May 7, 2025

Group size: 4 — Setup needed: Windows/macOS, Unix hosts, SPHERE.[1,2]

1 General description

In this lab, we are experimenting with cross-site-scripting attacks, and encrypted mail. This is based on a lab by Brandon Paulsen and Peter Peterson.

1.1 Requirement

- You use the instructions provided with this lab. You should use SPHERE, as this experiment can do damage.
- Your output should demonstrate that you have done the experiments, such as screenshots, Unix typescripts, and logs showing blocks.

2 Installation

See the included file lab-xss.zip for instructions and guidance.

3 Setup

1. Create an instance of this exercise by following the SPHERE instructions, using `xss` as Lab name. Your topology will look like below:
2. After setting up the lab, access your `s`erver node.

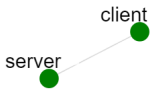

Figure 1: SPHERE topology view

Note: After running the installation script, it will finish running once it states "Successful page open!" This is a script that will access all topics in the forum, and it's important not to end this, as it runs the exploit for this lab. Simply open another terminal to SSH into the server node. Make sure that you save your work as you go.

Your XSS payload will need to obtain the forum authentication cookie value of any viewers and will also need to submit that value using a GET request to the steal.php script described under Experiment Configuration (below).

3.1 Experiment Configuration

This experiment has two nodes: a client at 10.0.1.2 (King Teshwan's computer) and a server at 10.0.1.1 (the host for slothsunlimited.com and Lord Dingwall's personal computer). The URL for the forum is http://10.0.1.1/ (although if you use port forwarding it will be http://localhost:PORTNUM/). Lord Dingwall periodically accesses each topic on slothsunlimited.com using a headless web browser. This means if you inject javascript into the forum, Dingwall's browser will eventually execute it.

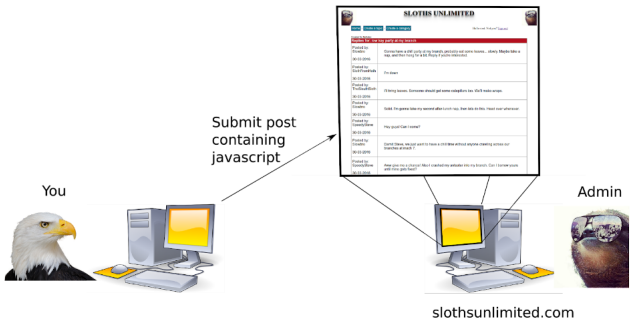

Figure 2: SPHERE view

4 Tasks

4.1 XSS (Cross-Site Scripting)

For years, the sloths and the bald eagles have coexisted in the plains of Mount Olympus under the treaty of Two's Complement. As the sloths ranks expanded, Lord Dingwall, branch manager of the

sloths, realized the need for better communication among his people. Thus he collaborated with the spider nation to create the web domain slothsunlimited.com. Unfortunately, Dingwall accidentally insulted the eagles toupees, precipitating the dissolution of the treaty and upsetting the balance of power in the greater Mount Olympus metropolitan area. Now, King Teshwan, chieftain of the eagles, bent on revenge, has decided to attack the sloths at the heart of their communication.

Your task, as King Teshwan, is to devastatingly embarrass the sloths by creating a category called "Eagles" on the website by performing a persistent XSS attack on the forum (i.e., by injecting javascript into the website). Dingwall obsessively rereads each thread on the forum (every 5 seconds or so) and his browser has an authentication cookie that keeps him logged in to the forum as an administrator. If Dingwall reads a malicious post containing Javascript in his browser, the Javascript in the post can access the authentication cookie in Dingwall's browser (as allowed by the Same Origin Policy). Then, the same Javascript can exfiltrate (i.e., leak) this cookie value to Teshwan. Once Teshwan has the cookie, he can use it to perform tasks as Dingwall on the forum. While there are many ways you could do this, you only need to find one way.

The sloths will be so embarrassed by this that they will apologize to the eagles and hopefully come to peace again. Then, in the role of Lord Dingwall, you must patch the vulnerability exploited by King Teshwan by adding input sanitization to the forum code.

Note: You will probably want to set up port forwarding for tunneling HTTP over ssh so you can test the web applications with a browser on your own desktop.

4.1.1 Viewing it from your computer

If you have already setup port forwarding using the port-forward.sh script from /share, you may access the forum with this command:

```
$ ssh -L 8080:server:80 proj-xdc-user
```

where proj is your project name, xdc is the name of your XDC, and user is your Merge username.

If you prefer to use the mrg testbed tool, you may use the following commands:

```
$ mrg login username
```

```
$ mrg xdc ssh xdc.proj -L 8080:server:80{
```

...where username is your Merge username, xdc is the name of your XDC, and proj is the name of your project.

Once you have set up port forwarding with one of these methods, you can access the site using http://localhost:8080/index.php.

For more information regarding port forwarding, you may check out the Merge documentation if you are using the mrg CLI. Otherwise, you may view the documentation for port-forward.sh instead.

The sloths have written four sanitization functions located in /var/www/html/sanitize.php. The first is a "no sanitize" function, and the next three attempt to sanitize strings by stripping html tags in increasingly sophisticated ways.

The injected Javascript should access Lord Dingwall's authentication cookie for the forum and send it to Teshwan so that he can create the embarrassing "Eagles" category.

Teshwan's computer is running an apache server which has a script called steal.php, located in /var/www/html/steal.php (with the URL http://10.0.1.2/steal.php). This script takes a cgi parameter "cookie", and crafts a request to slothsunlimited.com which will create the category "Eagles" using cookie's value. The cookie parameter should be the entire cookie value (Ex: PHP_SESS=XYZ). Assuming the value of cookie is Dingwall's session token, steal.php will handle the rest.

Sloths Unlimited Web Forum The sloths unlimited web forum code is located in /var/www/html on 10.0.1.1. It uses PHP to interface with a MySQL backend. You can access the backend by logging in as the user 'root' with the password 'root'. (See the SQL lab for information on using the MySQL CLI).

Software Tools No new software tools should be required for this exercise. For instructions on using the mysql client to look at the database for slothsunlimited.com, see the lab manual for the SQL Injection lab. Similarly, see previous lab manuals for instructions on using diff and patch.

Changing the Sanitization Routine For instructions on setting up SSH tunneling, see SPHERE's tutorial on port forwarding. The sanitization routines are located in /var/www/html/sanitize.php. Initially, all routines except the first one will be commented out. To change to a different routine, you only need to comment out the currently active routine, and uncomment the routine you want to enable.

Resetting the Forum You will want to reset the forum to its original state to ensure that your old payload isn't triggering the exploit, rather than your current payload. To reset the forum's state, run the script located in /lab/reset.sh.

Note that this script will destroy any payloads you submitted to the server.

How Will I Know If My Exploit Worked As mentioned before, there is a script running on the server which repeatedly opens every topic page it finds on the server. The script waits five seconds between each topic that it accesses. Initially, there are three topics on the forum, so if you inject a working exploit in one of these three topics, you should see the "Eagles" category appear on the homepage within 15 seconds. Any new topics you add will increase the time you have to wait to see your exploit work. Tasks

Your job is to write payloads that will defeat the first three sanitization functions. The fourth function is extra credit (10 points). Defeating entails submitting some payload to slothsunlimited.com anywhere on the website which steals Dingwall's cookie and sends it to the steal.php script on Teshwan's computer. Your exploit will probably make a request that looks like this: http://10.0.1.2/steal.php?cookie=COOKIE_VALUE

Fix the flaws that allowed you to steal Dingwall's session cookie by writing your own sanitize function in sanitize.php. Your function should not remove characters, but instead encode dangerous characters so that they will display properly but not be executed.

PHP has built-in functions for encoding dangerous characters!

Create a patch against sanitize.php. Write two short memos, one from King Teshwan's perspective to Lord Dingwall, describing how you defeated each of the sanitization routines and one from Lord Dingwall's perspective to King Teshwan, apologizing for "Toupeegate" and explaining how the forum was patched to be immune from this attack.

4.1.2 Summary for submission

Make a directory called xss with the following files:

1. The payloads used to exploit each sanitization routine (text files payload1, payload2, and payload3).
2. Your sanitize.php patch (sanitize.php.patch)
3. The two memos, attack-memo and fix-memo (1-2 paragraphs each) describing the attack and fix.

Make a tarball out of xss called xss.tar.gz.

4.2 Mail security

Install GNU Privacy Guard (GPG), an open-source implementation of Pretty Good Privacy.

1. Using GPG, generate a public/private key pair for your group, and request the public key of the TA for this lab. Choose a key size adequate for modern times. Consult `keylength.com`, if necessary.
2. Create a message, sign it and encrypt it to their public key. The message should be ASCII-armored to allow for inline inclusion in an e-mail, but do not actually send the email. Make sure both the cleartext and encrypted messages are included in your submission.

5 Word Problems

1. To what extent do utilities like NoScript protect against cross-site-scripting attacks?
2. How would you break the security provided by GPG?

6 Deliverables

1. A zip file containing:
   - A report describing all your findings above, including the xss.tar.gz file.
   - The messages with the signatures you generated.
   - The public keys you generated.
   - Answers from the word problems.
2. Submit by class on May 14, 2025.

7 Grading

Points will be subtracted if any of the pieces of the deliverables are missing or incomplete. The late submission policy applies.