**York College of Pennsylvania**

CS 497

Project 3: Cross-Site Scripting Attack Lab

**Introduction**

Cross-site scripting (XSS) is a type of vulnerability commonly found in web applications. This vulnerability

makes it possible for attackers to inject malicious code (e.g. JavaScript programs) into victim’s web browser.

Using this malicious code, the attackers can steal the victim’s credentials, such as cookies. The access

control policies (i.e., the same origin policy) employed by the browser to protect those credentials can be

bypassed by exploiting the XSS vulnerability. Vulnerabilities of this kind can potentially lead to large-scale

attacks.

To demonstrate what attackers can do by exploiting XSS vulnerabilities, we have set up a web-based

project management software named Collabtive. We modified the software to introduce an XSS vulnerability in this project management software; this vulnerability allows users to post any arbitrary message, including JavaScript programs, to the project introduction, message board, tasklist, milestone, timetracker and even the profiles. Students need to exploit this vulnerability by posting some malicious messages to their profiles; users who view these profiles will become victims. The attackers’ goal is to post forged messages for the victims. This lab may be done as a team of 2-3 students. Recommend watching the videos found at <http://www.cis.syr.edu/~wedu/seed/Labs/Web/XSS_Collabtive/> :

* Introduction to XSS
* How XSS Works
* XSS Discussion
* More Details on XSS

Also recommend the information in the Helpful Links section of the same URL mentioned above. Also there is a simple TCP client-server program you may need listed with the URL.

**Lab Environment**

Be sure that the Oracle VirtualBox is installed. Also download the **SEEDUbuntu12.04** VM that can be obtained from <http://www.cis.syr.edu/~wedu/seed/lab_env.html>. In this lab, we will need three things, all of which are already installed in the provided VM image:

1. The Firefox web browser
2. The Apache web server
3. The Collabtive project management web application

In Firefox, you need to use the LiveHTTPHeaders extension to inspect the HTTP requests and responses. The pre-built Ubuntu VM image provided to you has the Firefox web browser with the required extension already installed. A video is also available to help you with the lab and can be found at <http://www.cis.syr.edu/~wedu/education/websec3.html>.

**Starting the Apache Server.** The apache web server is also included in the pre-built Ubuntu image.

However, the web server is not started by default. You have to first start the web server using one of the

following two commands:

% sudo service apache2 start

**The Collabtive Web Application.** We use an open-source web application called Collabtive in this lab. Collabtive is a web-based project management system. This web application is already set up in the pre-built Ubuntu VM image. We have also created several user accounts on the Collabtive server. To see all the users’ account information, first log in as the admin using the following password; other users’ account information can be obtained from the post on the front page.

username: admin

password: admin

**Configuring DNS.** We have configured the following URL needed for this lab. To access the URL , the Apache server needs to be started first:



The above URL is only accessible from inside of the virtual machine, because we have modified the /etc/hosts file to map the domain name of each URL to the virtual machine’s local IP address (127.0.0.1). You may map any domain name to a particular IP address using /etc/hosts. For example you can map http://www.example.com to the local IP address by appending the following entry to /etc/hosts:

127.0.0.1 [www.example.com](http://www.example.com)

If your web server and browser are running on two different machines, you need to modify /etc/hosts on the browser’s machine accordingly to map these domain names to the web server’s IP address, not to 127.0.0.1.

**Configuring Apache Server.** In the pre-built VM image, we use Apache server to host all the web sites used in the lab. The name-based virtual hosting feature in Apache could be used to host several web sites (or URLs) on the same machine. A configuration file named default in the directory "/etc/apache2/sites-available" contains the necessary directives for the configuration:

1. The directive "NameVirtualHost \*" instructs the web server to use all IP addresses in the machine (some machines may have multiple IP addresses).

2. Each web site has a VirtualHost block that specifies the URL for the web site and directory in the file system that contains the sources for the web site. For example, to configure a web site with URL http://www.example1.com with sources in directory /var/www/Example\_1/, and to configure a web site with URL http://www.example2.com with sources in directory /var/www/Example\_2/, we use the following blocks:



You may modify the web application by accessing the source in the mentioned directories. For example, with the above configuration, the web application http://www.example1.com can be changed by modifying the sources in the directory /var/www/Example\_1/.

**Other software.** Some of the lab tasks require some basic familiarity with JavaScript. Wherever necessary,

we provide a sample JavaScript program to help the students get started. To complete task 3, students

may need a utility to watch incoming requests on a particular TCP port. We provide a C program that can be configured to listen on a particular port and display incoming messages. The C program can be downloaded

from the web site for this lab.

**Lab Tasks**

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

The objective of this task is to embed a JavaScript program in your Collabtive profile, such that when another user views the profile, the JavaScript program will be executed and an alert window will be displayed. The following JavaScript will display an alert window:

<script>alert (’XSS’); </script>

If you embed the above JavaScript code in your profile (e.g. in the company field), then any user who views your profile will see the alert window. In this case, the JavaScript code is short enough to be typed into the company field. If you want to run a long JavaScript, but you are limited by the number of characters you can type in the form, you can store the JavaScript program in a standalone file, save it with the .js extension, and then refer to it using the src attribute in the <script> tag. See the following example:



In the above example, the page will fetch the JavaScript program from http://www.example.com, which can be any web server.

**Task 2: Posting a Malicious Message to Display Cookies**

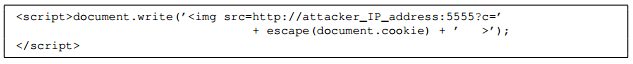
The objective of this task is to embed a JavaScript program in your Collabtive profile, such that when another user views your profile, the user’s cookies will be displayed in the alert window. This can be done by adding some additional code to the JavaScript program in the previous task:

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

**Task 3: Stealing Cookies from the Victim’s Machine**

In the previous task, the malicious JavaScript code written by the attacker can print out the user’s cookies, but only the user can see the cookies, not the attacker. In this task, the attacker wants the JavaScript code to send the cookies to himself/herself. To achieve this, the malicious JavaScript code needs to send an HTTP request to the attacker, with the cookies appended to the request.

We can do this by having the malicious JavaScript insert an <img> tag with its src attribute set to the attacker’s machine. When the JavaScript inserts the img tag, the browser tries to load the image from the URL in the src field; this results in an HTTP GET request sent to the attacker’s machine. The JavaScript given below sends the cookies to the port 5555 of the attacker’s machine, where the attacker has a TCP server listening to the same port. The server can print out whatever it receives. The TCP server program is available from the lab’s web site. “localhost:5555?c=”



**Task 4: Impersonating the Victim using the Stolen Cookies**

After stealing the victim’s cookies, the attacker can do whatever the victim can do to the Collabtive web server, including creating a new project on behalf of the victim, deleting the victim’s post, etc. Essentially, the attack has hijacked the victim’s session. In this task, we will launch this session hijacking attack, and write a program to create a new project on behalf of the victim. The attack should be launched from another virtual machine.

To forge a project, we should first find out how a legitimate user creates a project in Collabtive. More specifically, we need to figure out what are sent to the server when a user creates a project. Firefox’s LiveHTTPHeaders extension can help us; it can display the contents of any HTTP request message sent from the browser. From the contents, we can identify all the parameters in the request. A screen shot of LiveHTTPHeaders is given in Figure1. The LiveHTTPHeaders is already installed in the pre-built Ubuntu VM image.

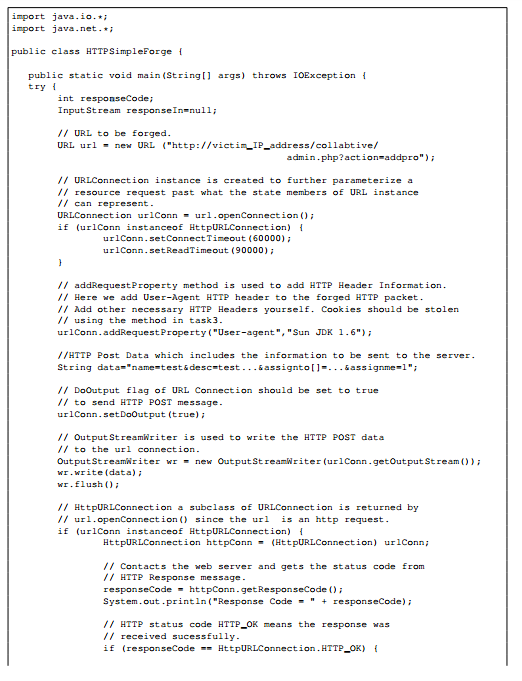
Once we have understood what the HTTP request for project creation looks like, we can write a Java program to send out the same HTTP request. The Collabtive server cannot distinguish whether the request is sent out by the user’s browser or by the attacker’s Java program. As long as we set all the parameters correctly, and the session cookie is attached, the server will accept and process the project-posting HTTP request. To simplify your task, we provide you with a sample java program that does the following:

1. Opens a connection to web server.

2. Sets the necessary HTTP header information.

3. Sends the request to web server.

4. Gets the response from web server.



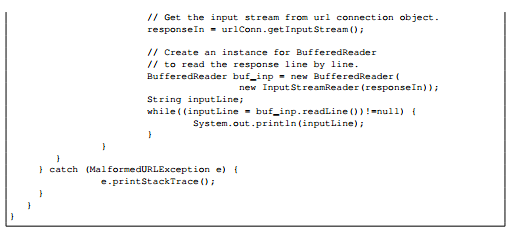


Figure 1. Sample Java Program

If you have trouble understanding the above program, we suggest you to read the following:

* JDK 6 Documentation: http://java.sun.com/javase/6/docs/api/
* Java Protocol Handler: http://java.sun.com/developer/onlineTraining/protocolhandlers/

**Report**

You need to submit a detailed lab report to describe what you have done and what you have observed.

Please provide details using LiveHTTPHeaders, Wireshark, and/or screenshots. You also need to

provide explanation to the observations that are interesting or surprising.

**Grading**

Post your report in Moodle by the scheduled due date in the syllabus. Make sure your name and class information is on your paper. Your grade for this lab will be composed of:

* 33% - Description
* 33% - Details/screenshots
* 34% - Explanation

**References**

[1] AJAX for n00bs. Available at the following URL: <http://www.hunlock.com/blogs/AJAX_for_n00bs>.

[2] AJAX POST-It Notes. Available at the following URL: <http://www.hunlock.com/blogs/AJAX_POST-It_Notes>.

[3] Essential Javascript – A Javascript Tutorial. Available at the following URL:

<http://www.hunlock.com/blogs/Essential_Javascript_--_A_Javascript_Tutorial>.

[4] The Complete Javascript Strings Reference. Available at the following URL:

<http://www.hunlock.com/blogs/The_Complete_Javascript_Strings_Reference>.

[5] Web Based Project Management With Collabtive On Ubuntu 7.10 Server. <http://howtoforge.com/web-based-project-management-with-collabtive-on-ubuntu7.10-server>



Figure 2. Screenshot of LiveHTTPHeaders extension

© 2006-2011 Wenliang Du, Syracuse University