CS4235 / CS6035 Intro to InfoSec
Lab 2 Web Security

Release: 3/28/2016
Due: 4/4/2016 (late submission due: 4/6/2016)
Submission: Organize your result document well in a pdf file and submit via Tsquare.
Full points: 20 (late submission yields 2 points deduction)
TA in charge: Yang Ji

Prerequisites: Install the latest version of Chrome browser inside the Kali VM
Target webapp: http://ec2-52-36-128-0.us-west-2.compute.amazonaws.com:<assigned port>
<assigned port> is 808X, where X is your last digit of gtid, e.g., 8080,8081,...8089. To distribute load, please only play with your assigned port.

NOTE

This webapp is specially designed to be vulnerable. Do not do anything harmful (use your common sense) beyond the demonstration of web vulnerabilities specified in this lab.

We encourage cooperation but do not allow plagiarism. Each group can have at most three students. Each group needs only one submission. Indicate your group members in your submission. Similar submissions from different groups will be investigated.

Useful references:

https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)
https://portswigger.net/burp

Problem One: Exploits (15 Points Total)

Cross-site scripting (XSS) is one of the top web application vulnerabilities. XSS enables attackers to inject client-side script into web pages viewed by other users. A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same-origin policy. In this problem, you need to demonstrate the exploit of XSS at a vulnerable web application.

Subproblem 1: Reflected XSS (5 Points)

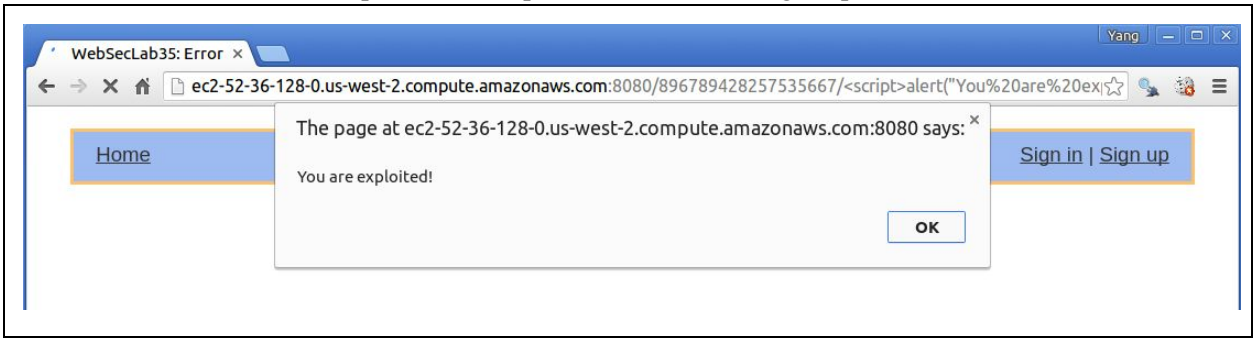
We have found a reflected XSS available at the target website. If we try appending the scripts along with the HTTP request, the website improperly processes the script and returns the script back to the client. We can demonstrate the exploit:

Fill in the URL bar with

```
https://ec2-52-36-128-0.us-west-2.compute.amazonaws.com:808X/123...
/<script>alert("You are exploited!")</script>
```

Note: replace the red part with real ones in your case.

We can see the result of the exploit as we expected in the following snapshot:



Task: Answer the following questions.

a) Try the exploit yourself and provide snapshots. The snapshot should show your signature (something about you uniquely). (1.5 points)
b) Draw a diagram of the exploit and explain the procedure. (1 point)
c) List 2 other forms of payload (e.g., encoding) that give the same result of the exploit (i.e., a popped dialog defined by the attacker when the victim accesses a link). (1 points)
d) What is the cause of this exploit? Any quick fix? (1.5 points)

Subproblem 2: Stored XSS (5 Points)

Find a stored XSS. You need to find a place where the target webapp will store the script and serve it back to another user. Demonstrate your exploit by filling the following report. (Hint: use the snippet to insert payload script.)

Bug report
Name: Stored XSS
Description, payload (2 points):
Snapshots (Try to be explicit, highlight the key parts) (2 points):
Pseudo Code to fix the bug from server side (1 point):

Subproblem 3: Cross-Site Request Forgery (5 points)

Cross-site request forgery (CSRF), is a type of malicious exploit of a website where unauthorized commands are transmitted from a user that the website trusts.

Find a way to delete another account's snippet. Assume you can lure him/her to click a URL link. Please always manipulate only your own created accounts. Do not tamper with other students' accounts. Fill in the bug report.

Bug report
Name: CSRF
Description, payload (2 point):
Snapshots (Try to be explicit, highlight the key parts) (2 points):
Detail the common practice to thwart CSRF (1 points):

Problem Two: Sanitizer (5 Points Total)

To thwart the cross site scripting exploits, sanitizers are deployed to stop or modify the requests with malicious payloads. These sanitizers are usually placed at the server, the client or a middle box between server and client. In this problem, you will provide solution to stop the exploits using Burp, one of the tools in Kali.

Specifically, get familiar with the Burp suite tool. Set up a proxy to intercept the HTTP traffic and establish a rule to stop the exploit of Reflected XSS (Subproblem 1 of Problem One).

a) What is the detailed procedure for setting this rule? (1 point)
b) Provide screen captures that show that your rule stopped an attempt at the exploit. (2 points)
c) What is the limit or side effect of your solution? Provide snapshots of examples to support your arguments. (2 points)