Song
Spring 2015

CS 161
Computer Security

Project 2

Due: Apr. 3, 11:59PM

Getting Started

This project is to be done individually. If you were hoping for some jolly cooperation, sorry! Before you begin, **you should read the instructions for each problem carefully.**

Go to http://chloe.eecs.berkeley.edu:6969/website/id.php. This will allow you to access your Project 2 ID. Just enter your student ID when where it asks for it, and it will give you your Project 2 ID. This is necessary because one of the problems in the lab involves SQL Injection, and we don't want you to directly access the student IDs of other students (university policy). For the rest of the project, ID refers to Project 2 ID. We will specify student ID if we are referencing your student ID.

SQL Injection

For this problem, you will perform a SQL injection attack to infiltrate an online database of informants and retrieve the password of the particular informant that corresponds to your ID. The database stores passwords hashed with a prepended salt. However, the original passwords are limited to only lowercase alphabetical characters (a-z). Use SQL injection to retrieve this hashed password.

After retrieving the hash value from the database, you must reverse the hash and discover the actual password. You may write your own program to crack the password or use an existing program (we suggest Hashcat. You can download the program for Mac, Windows and Linux here. Which executable you use depends on the extension corresponding to your architecture and operating system).

Note: a *salt* is a random string added to users' passwords before hashing to make dictionary attacks harder. A *hash* function is basically the same as those used in dictionaries, and is a one-way function. Don't worry if these aren't quite familiar yet, because password-cracking software will have documentation on how to crack the hashes we've given to you, for commonly used functions like SHA1 and MD5. These are things hackers and programmers commonly have to read and learn.

Start this task by visiting the page that allows you to search for informants based on their primary areas of operation: http://chloe.eecs.berkeley.edu:6969/website/informantlist/search.php. You'll find that this access point has a SQL injection vulnerability. Find it and

trigger it to get more information than the developer intended. There are some defenses, but they are not sophisticated. You might need to try different variants of your attacks.

For submission, create a file called q1.txt whose **first and only** line is the password for the informant with your ID. **Do not write anything else in the file.** You will lose ALL points for not following this format. Note that the question is asking for the password, and not the hashed password (meaning you should be putting the cracked value in q1.txt). You should also submit a q1code.txt file containing the code that you used to solve the question.

Cross-Site Scripting

In this problem, you will be attacking a website that allows students to create a profile online. All students must create a profile that administrators can view to get more information about them. The profile is simple; it consists of your name, an About You paragraph, and a homepage URL. All the other details, available only to administrators, are filled in by the dreaded secret police. You can view your profile, as can administrators, but other students (sheeple!) cannot. Your goal for this problem is to execute code as an administrator by successfully completing an XSS attack through your user profile.

To start this assignment, you will need your user id and password. Your User ID is the same as in Question 1. Your password is the password you have uncovered in Question 1 (your answer to Question 1). Begin by visiting `http://chloe.eecs.berkeley.edu:6969/website/xss/` and logging in with your user id and password.

There are two pages of particular interest: update and view. You will probably want to start by just playing around with the update page, entering in relevant information, and then viewing the results on the view page. Remember that the best hackers begin simply by viewing the source of a page. In order to get credit for this portion of the assignment, you must create a profile that an administrator will visit and fiddle with; he might load the page, scroll, click links, and so on. This should result in the Javascript function winning() being called. We have put that function on the page so you can test your injection.

Submit a q2code.txt which contains the code that you entered to successfully mount the attack.

Writeup

Here, we will examine different attacks and defenses, and how they affect hacking difficulty.

1. What is the purpose of hashing and salting passwords, as opposed to just hashing passwords? How is this a more effective defense?
2. List an advantage and a disadvantage of blacklisting as compared to whitelisting as a form of XSS defense.
3. If you poke around the XSS website we used in the second part of the project and execute some Javascript in the console, you can see that the website uses a session ID cookie to keep track of who's currently logged in. You can just write `document.cookie` in the console to confirm this. Are CSRF attacks with this site possible? How could we prevent them?

Submit your answers in a writeup.txt file.

Password Cracking Help

Help with password cracking will be updated on Piazza as needed. Feel free to go to office hours if you're stuck! We're here to help you.

Submission Summary

In summary, you must submit the following files:

```
q1.txt
q1code.txt
q2code.txt
writeup.txt
```

Copy all of the above files into a proj2 directory on an instructional machine, and run **submit proj2** from inside the proj2 directory.

Congratulations on finishing Project 2! We hoped you liked wearing the attacker's hat, but not too much ;)