CS 361S - Network Security and Privacy
Spring 2016

Project #1

Due: 11:00am, March 22, 2017

Submission instructions

Follow the instructions in the project description.
If you are submitting late, please indicate how many late days you are using.

Collaboration policy

This assignment can be done individually or in two-person teams. Any cheating (*e.g.*, submitting another person's work as your own, or permitting your work to be copied) will automatically result in a failing grade. The Computer Science department code of conduct can be found at https://www.cs.utexas.edu/academics/conduct.

Late submission policy

This project is due by 11:00am on Wednesday, March 22. All late submissions will be subject to the following policy.

You start the semester with a credit of 3 late days. For the purpose of counting late days, a "day" is 24 hours starting on the assignment's due date/time. Partial days are rounded up to the next full day. You are free to divide your late days among the take-home assignments (3 homeworks and 2 projects) any way you want: submit three assignments 1 day late, submit one assignment 3 days late, *etc.* After your 3 days are used up, no late submissions will be accepted and you will automatically receive 0 points for each late assignment.

Project #1 (50 points + 10 bonus points)

The objective of this project is to give you hands-on experience implementing attacks against vulnerable Web applications. You will do this project on a virtual machine using VMware Player.

You will need:

- **Everything outlined in the VM Setup Guide**
- The project image:
  https://www.cs.utexas.edu/~ojensen/courses/cs361s/vms/cs361s-proj1-2017.ova

Getting Started

1. Download and import the project image. **Refer to the VM Setup Guide for details**. Note that VMWare Player might require you to "relax" the rules when you go to import the .ova file. Do so.

2. The machine has two accounts: root/root and user/user. You will do your work as user, but feel free to explore as root.

The vm has an SSH server. You can SSH into the vm from your machine, using the IP address produced by ifconfig (see VM Setup Guide) as the destination. You can also use this to transfer files into the vm using scp or sshfs.

Some attacks will require an email to be sent to user on the system. You will need a server-side script to automatically email information captured by your client-side JavaScript.
We have provided this script for you at http://hackmail.org/sendmail.php (open this URL from within the vm for more instructions) and use that URL in your attack scripts to send emails. Any mail the user receives will appear in /var/mail/user.

Interacting with the VM

Viewing / Modifying Files

The least painful way of interacting with files on the VM is to use SSHFS. Packages exist for Linux, OSX and Windows.

1. Install the sshfs package on your *host* machine and create an empty netfs directory.

2. Run sshfs root@[vm IP address]:/ netfs and enter the vm's root password. You can now access the vm's filesystem via the netfs directory with your host machine's applications.

Alternatively, you may prefer interacting with the filesystem via SCP. See WinSCP, Cyberduck and Filezilla for Windows, OSX and Linux respectively.

Accessing the VM's websites

You have two options: use the browser installed in the VM (recommended), or use a browser installed on your host machine (not recommended).

To use the VM's browser, log in to the VIA via ssh -Y user@[vm IP address]. You can now run graphical applications such as firefox within the VM.

If you want to use your host machine's browser, you'll need to do a little more work. You'll need to edit your operating system's HOSTS file, and add an entry for dncmail.org and hackmail.org, using the IP of the VM. At this point, accessing dncmail.org or hackmail.org in your host machine's browser should render the VM pages (assuming your VM is powered on). *If you choose to use your host machine's browsers, you may need to turn off your browser's XSS filters!*

DNCMail.org

Due to recent issues regarding electronic communications, the Democratic National Committee Secure Announcement Server is the canonical method for members of the DNC to securely release announcements. You can think of it as a simple purpose-built Twitter. You sign up with an email address and enter your announcement. Anybody can then search DNCMail for your email address, securely reading your announcement and thereby preventing identity fraud. Other members should be able to see your announcement, but should not be able to edit it.

DNCMail is located at this URL (accessible only from within the VM):

http://dncmail.org

You can create a new account by clicking "register here" on the main page and filling out the form. Then, after logging, you can search for other people's announcements and/or upload your own.

The source code of the dncmail website is available within the VM in /var/dncmail/www. Note that the application's database is stored in the /tmp directory, which means that the database will reset whenever you reboot the VM. Note also that dncmail.org was clobbered together from existing source code (as any self-respecting government contractors would do) and as a result may have misleading ids and naming schemes in their source (in particular, "pubkey" should generally be interpreted as "announcement.")

Try it out yourself: search for the announcement associated with victim@naive.com.

Attack #1: Cross-site request forgery (10 points + 5 bonus points)

Hapless Victim (victim@naive.com) has made an important political announcement on dncmail.org. Your goal is to replace Hapless' announcement with your own malicious announcement, so that you can sabotage his/her opportunity to win the primaries.

Create a malicious HTML page that should work as follows. Suppose the victim has logged into the Announcement Server, and, while still logged in, visits your HTML page. Your page should overwrite the victim's current announcement with this malicious announcement, designed to sow discord throughout the party:

OtherCandidate is a tooootaallll looooooosssseeerrr

Important: The victim should be redirected to the dncmail.org website immediately. In particular, he should not see the URL or the content of the malicious HTML page. (It is Ok if the browser displays your malicious page for a fraction of a second before it finishes fetching the dncmail.org page.)

Bonus (5 points)

Instead of redirecting the victim as described in the previous paragraph, make the attack invisible to the victim. In this case, the victim should see only the URL and content of your malicious HTML page. For example, the victim is browsing his favorite forum and sees your link promising a cute picture of a kitten. He clicks your link, sees the kitten, nods appreciatively, then closes the tab, unaware that his data at dncmail.org has been modified.

Attack #2: Cookie theft (15 points + 5 bonus points)

A user whose email address is victim@naive.com has logged into the DNCMail website. Create a URL that looks like this (with EVILMAGIC replaced by your exploit):

http://dncmail.org/account.php?email=EVILMAGIC

When the logged in victim visits this URL, the victim's DNCMail cookie should get sent by email to user.

The user should notice no difference in the behavior or appearance of the web page compared to simply typing victim@naive.com into the text box on http://dncmail.org/account.php and hitting Enter. The source of the page can be arbitrarily different, but it should look and feel exactly the same.

Important: Your solution *may not* cause additional pageloads or redirects. For example, simply submitting the form once more or redirecting to http://dncmail.org/account.php?email=victim@naive.com after stealing the cookie is not what we are looking for. You

must exploit the way that the email variable is used in the PHP script. In particular, your attack code must:

- Pull the victim's record from the database using the SQL query on line 22 of account.php (therefore, SQL must not barf on being given a query constructed from the username part of your URL).
- Result in the correct email address (victim@naive.com) being displayed in the input field on the user page. Thus, when the PHP code spits back the username you gave it on line 47 of account.php, it must somehow render as victim@naive.com. It should be **exactly** that string—you cannot have more text hidden beyond the whitespace in the input box.
- Display the user's email address and announcement in the area to the right. Your code should also ensure that even though the email address you supplied is a long and ugly string, it should render as victim@naive.com in this part of the page as well.

To summarize, you attack should, without redirection, result in a page that looks exactly like the page http://dncmail.org/account.php?email=victim@naive.com

The HTML source will be different, and so will the address bar (it will be your malicious URL) but the content of the page should look and behave the same.

Tip: You are allowed to hardcode the string victim@naive.com wherever you want. You cannot, however, hardcode the value of the announcement. it should be retrieved from the database. That is, we do *not* guarantee that Victim's announcement will be the same on the grader's version of the VM.

Partial credit: If you are not able to email the cookie, at least display it in a pop-up alert. If you are not able to make the page look exactly the same, make it look approximately the same. At the very least, try to make sure that your URL does not result in the "This email address is not registered" warning. While some points will be given for simply sending the email / alerting the cookie, the majority of the points for this question are for cleaning up the display after the email has been sent.

Bonus (5 points)

The team with the **shortest** URL that correctly implements the full attack #2 gets 5 bonus points.

Attack #3: Password theft (10 points)

Create a malicious HTML page that should work as follows. Assume your victim is not logged in. Upon visiting your page, the victim should be redirected to http://dncmail.org/. When the

victim enters a username and password and hits "Log in", an email should be sent to user containing the username and password entered by the victim.

Important: This is *not* a phishing attack. Assuming a valid username/password pair was entered, the victim should be logged into DNCMail.

Attack#4: SQL injection (15 points)

Create an HTML page that the tester will open in his browser. The tester will not be logged in. The HTML page should have a form with a single text field and a submit button (note: the form should not ask the tester for a password). The tester will type a username into the text field and submit the form. You can assume the username submitted by the tester is already associated with a registered account.

As a result, the tester should be logged in as the user whose username he submitted. The browser's location bar should be http://dncmail.org/account.php, and the page should function exactly as if the correct username and password were entered on the real site.

Deliverables

You will submit your project using Canvas. Each attack should be self-contained; your exploits should not depend on anything outside of the VM (including anything on the Internet). If you use more than one file in an exploit, please name them such that it is immediately obvious which files do what (e.g. attack1.html, attack1 image.jpg, attack1 dep1.html, etc.).

Your submission will be a single Gzipped Tar archive, proj1.tar.gz, consisting of the following files:

- Your malicious HTML page implementing Attack #1, attack1.html.
  - If you attempted the bonus, a separate HTML page, attack1_bonus.html.
- A plain text file with your malicious URL implementing Attack #2, attack2.txt.
- Your malicious HTML page implementing Attack #3, attack3.html.
- Your malicious HTML page implementing Attack #4, attack4.html.
- A plain text file, SUBMISSION. The first line should state how many (possibly 0) late days were used. Then give the following on a single line, one line for each student:
  - Your **UT EID**, followed by a single **space**, followed by your **real name**.

You may also include in your Tar archive a README file with comments about your experiences or suggestions for improvement.