Lab 4: Web Security

50.020 Security

Hand-out: February 21
Hand-in: 9pm, February 28

1 Objectives

- Perform SQL Injection and Cross-Site Scripting on a vulnerable web site
- Perform Command Injection to execute arbitrary commands
- We provide you the source code. In case you get stuck, you can have a look to understand better what you are doing

2 Part 1: SQL Injection

We distribute a `lab4server.py` server script, run it with python3 locally.
After starting the server locally, go to `http://localhost:5005` to be greeted by a login page. Alice's email is alice@alice.com, but you don't have her password. Based on the example attacks from the class, find a way to log in as Alice!

3 Part 2: Cross-Site Scripting

Restarting the webserver will refresh the SQL database

3.1 Persistent (second order) attack

Log in as Alice again, and check the different options the website gives you. Can you use one of them to conduct a second order XSS attack? Some hints on the idea to conduct this attack:

- Your goal is to inject some code on a page that the admin user will look at
- Your injected code (HTML or .js) should make the admin disclose his cookie to you
- In particular, assume that the admin sees a list of all news when he logs in

Some suggested steps of conducting a persistent XSS attack:

- Find a way to inject a simple XSS string into the database
- Exploit this to obtain a victim's `session` cookie and disclose the cookie as a public news. Some hints (assume the victim is admin):

  - Update your XSS to trick the admin's browser to do an HTTP GET to a URL of the attacker's choice
  - Make the admin visit `http://localhost:5000/news?text=`<adminsessioncookie>
  - This will create a public news entry with the admin's `session` cookie displayed.
- Test the persistent XSS works (again, assume the victim is admin):
  - Login as admin: use the second password from the `secrets` to log in.
  - Logout (as admin) after displaying the main site.
  - Login as alice again.
  - Your XSS attack works if you (alice) are able to see a news item with admin's cookie displayed!

3.2 Reflected (first order) attack

Login as alice (as attacker), and inject code (by crafting a URL) to conduct a reflected XSS attack. Exploit your injection to obtain the `session` cookie of the victim (again):

- Clicking the link your designed should make the admin send his `session` cookie to you (saved in a file).

4 Command Injection

4.1 Listing secret file

- This time, we will not inject any html/JavaScript code.
- Find a way to execute a shell command on the server by using one of the provided input fields
- Can you exploit this vulnerability to get the content of a local file, e.g. list the file "secret"?

4.2 Opening a reverse shell

- This time, we will try to get a local shell on the target server!
- A "reverse shell" is a program started on the victim machine, that connects to the attacker (to pass through firewalls and similar).
- We need a program waiting on your machine to accept incoming connections. Netcat can be a good tool for you: use Netcat on your machine (attacker) to open a listener
- Find a suitable command to run on the victim (the webserver, in this case, also localhost) to connect a shell to your machine. After the victim connects to you, you should be able to type commands, which will be executed on the victim's machine. Hint: have a look at `http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet`

5 Hand-In

Write a short report covering the following topics:

5.1 SQL

- Explain the string you used for the SQL injection and explain why it works.
- How can this vulnerability be fixed in the provided script?

5.2 XSS

- Explain which content you inserted in the persistent XSS attack, and how it works
- Explain which content you inserted in the reflected XSS attack, and how it works

5.3 Injection

- Explain which command you used to obtain the content of "secret", and why it works
- Explain which commands you used to open a listener and obtain the reverse shell, and why it works