George, Levi

CS44500 – Computer Security

03/18/2023

Lab 7 – Shell Shock Attacks

Env Setup.

DNS SETTINGS

I open the /etc/hosts file



I added 10.9.0.80 to my /etc/hosts file



I added the lab files and put everything from it into SHLSHK\_LAB

Then I ran dcbuild

Text

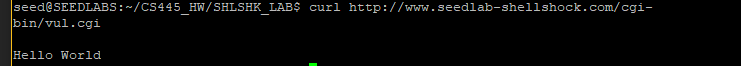
Description automatically generated

Then I ran dcup

Text

Description automatically generated

I test the CGI by using the curl command from the lab.



TASK 1. EXPERIMENTATION WITH BASH

I ran the shellshock vulnerability in my personal VM.

I was able to run the basic vulnerability as shown in the class notes.

Text

Description automatically generated

I ran the same commands above but instead of invoking the vulnerable shell, I ran bash, the lack of the “extra” command being run shows that my bash is not vulnerable.

Text

Description automatically generated

TASK 2. PASSING DATA TO BASH VIA ENV VAR

For the browser approach, I was able to navigate to

This was what I got from HTTP Header Live when I went to the associated URL, when I match that up with what the browser sent to me. Everything looks just about the same.

Text

Description automatically generated with medium confidence

Graphical user interface, text, application, email

Description automatically generated

For via the Command Line (using CURL)

My first Curl command got me back this output

Text

Description automatically generated

An interesting thing to note is that CURL accepts only \*/\* data, additionally, it seems that the entire request is more generalized than what our browser sends.

When I ran [curl -A “newData” -v [www.seedlab-shellshock.com/cgi-bin/getenv.cgi](http://www.seedlab-shellshock.com/cgi-bin/getenv.cgi)]

I got back this output, and if we look closely, we see that the only thing that changed was the HTTP\_USER\_AGENT data. i.e. User-Agent=newData in our request.

Text

Description automatically generated

The next request I ran (curl -e ….) changed the HTTP\_REFERER of my request.

The final code (curl -H …..) actually created an HTTP environment variable based on what I entered, in this case DATAAGE became the env var HTTP\_DATAAGE, and it’s value was newData

Text

Description automatically generated

As it stands the -H option is our best bet for influencing and creating new env variables, -e and -A only let us modify two pre-existing vars.

TASK 3. SHELLSHOCK ATTACK

I ran a couple of commands

The first one below, returned no output, I presume because I didn’t format it correctly.



However, once I placed the space in front of echo, I got real data back.

Text

Description automatically generated

TASK 3.A – Return etc/passwd



I ran the above command and was able to get the server to send me this list of logins.

Text

Description automatically generated

TASK 3.B – Return Process User ID

I ran the command shown next, and was able to get back the ID (33) of the web server’s user which is the www-data user.

Text

Description automatically generated

TASK 3.C – Create a tmp file

I ran what I suspected would create the file in the temp folder

Text

Description automatically generated

I then ran the subsequent command to get the file info (we can actually verify that the commands worked the way I intended since the time returned from the server (3:41:06 GMT) matches the time from ls -l.

Text

Description automatically generated

TASK 3.D – Goodbye file1.txt

I ran a similar command that I did to create the file, except I used rm to delete it.

Text

Description automatically generated

I then ran the command to check the file using curl and found nothing

Text

Description automatically generated

TASK 3 – Questions

Q1. – Can we get the /etc/shadow file?

No, this is a protected file (highly so), only root would be able to get to it. However, our current user level is not high enough.

Q2. – Can we use the modified curl command using GET queries to launch our attack?

Yes, we can, however it may be faster to not do it that way. Simply since we would need to convert a bunch of our command characters into the appropriate format for HTTP requests. As shown below. I shellshock an echo “world” command to complete the phrase “Hello World”

Unless we have code to convert our malicious payloads into HTTP formatted code, it would just be quicker and less error prone to do it the way we have previously.

Text

Description automatically generated

TASK 4 – Getting a reverse shell

I started up NetCat on a separate putty terminal for my server



Then, I initiated the reverse shell by inserting a shell prompt into my shellshock attack from previous tasks. I suspect that the attack has succeeded based on the new User-Agent value and the fact that the current terminal didn’t show me the command line handle.

Text

Description automatically generated

I check my other putty terminal and find that my attack did succeed.

Text

Description automatically generated

I even run a command and see that it returns info from the current directory.

Text

Description automatically generated

Task 5 – Redoing Task A using the Patched BASH

First, I thought I would be clever and use the current shell I got through the previous task to edit the cgi program, but it just gave me an error.

Text

Description automatically generated

So, I decided it would be expedient to just open a shell to the container and edit the file there.

Text

Description automatically generated

Text

Description automatically generated

After the changes, I re-initiate the task 3A attack, but I was unable to get the content back from the passwd file.

Text

Description automatically generated