Lab 3 - Shellshock Attack Lab

1 Overview

On September 24, 2014, a severe vulnerability in Bash was identified. Nicknamed Shellshock, this vulnerability can exploit many systems and be launched either remotely or from a local machine. In this lab, students need to work on this attack, so they can understand the Shellshock vulnerability. The learning objective of this lab is for students to get a hands-on experience on this interesting attack, understand how it works, and think about the lessons that we can get out of this attack.

2 Lab Tasks

2.1 Environment Setup

The Bash program in Ubuntu 16.04 has already been patched, so it is no longer vulnerable to the Shellshock attack. For the purpose of this lab, we have installed a vulnerable version of Bash inside the /bin folder; its name is bash shellshock. We need to use this Bash in our task. Please run this vulnerable version of Bash like the following and then design an experiment to verify whether this Bash is vulnerable to the Shellshock attack or not.

```
$ /bin/bash_shellshock
```

2.2 Task 1: Attack CGI programs

In this task, we will launch the Shellshock attack on a remote web server. Many web servers enable CGI, which is a standard method used to generate dynamic content on Web pages and Web applications. Many CGI programs are written using shell script. Therefore, before a CGI program is executed, the shell program will be invoked first, and such an invocation is triggered by a user from a remote computer.

Set up the CGI Program. Create a very simple CGI program (called `myprog.cgi`) using the root account in folder /usr/lib/cgi-bin/ as the following. Make this cgi executable by running ”chmod 755 myprog.cgi”. It simply prints out `"Hello, I am Shellshock!!"` using shell script.

```
#!/bin/bash

echo "Content-type: text/plain"
echo
echo
echo "Hello, I am Shellshock!!"
```

The above CGI program in the `/usr/lib/cgi-bin` directory and is executable. This folder is the default CGI directory for the Apache web server. To access this CGI program from the Web, you can either use a browser by typing the following URL: `http://localhost/cgi-bin/myprog.cgi`, or use the following command line program `curl` to do the same thing:

```
$ curl http://localhost/cgi-bin/myprog.cgi
```

In our setup, we run the Web server and the attack from the same computer, and that is why we use `localhost`. In real attacks, the server is running on a remote machine, and instead of using `localhost`, we use the hostname or the IP address of the server.

Task 1A: Get Secret Data. After the above CGI program is set up, you can launch the Shellshock attack. The attack does not depend on what is in the CGI program, as it targets the Bash program, which is invoked first, before the CGI script is executed. Your goal is to launch the attack through the URL `http://localhost/cgi-bin/myprog.cgi`, such that you can achieve something that you cannot do as a remote user. In this task, you need to create an **account.db** file in directory /usr/lib/cgi-bin using the root account with some random content, e.g.,"ABCDE". As a regular user without the permission of accessing cgi-bin folder, you now need to get the content of account.db via the Shellshock attack. **Please describe how your attack works**. Note that, your screenshot of successful attack is required for this task.

Task 1B: Crash the Server. In this task, we want to crash the server with the the Shellshock attack. This kind of attack is typically happens for deny of service. In this task, the **/bin/sleep** function is your good friend in your attack. If you are not familiar with the **sleep** function, you can use *sleep –help* for more information. **Please describe how your attack works**.

2.3 Task 2: Attack `Set-UID` programs

In this task, we use Shellshock and bash function to attack `Set-UID` programs, with a goal to gain the root privilege. Before the attack, we need to first let `/bin/sh` to point to `/bin/bash` (by default, it points to `/bin/dash` in our SEED Ubuntu VM). You can do it using the following command:

```
$ sudo ln -sf /bin/bash /bin/sh
```

The following program is a `Set-UID` program, which simply runs the `"/bin/ls -l"` command. Please compile this code, make it a `Set-UID` program, and make `root` be its owner. As we know, the `system()` function will invoke `"/bin/sh -c"` to run the given command, which means `/bin/bash` will be invoked. Can you use the Shellshock vulnerability to gain the root privilege? Please explain why your attack can work (or not work).

```
#include <stdio.h>

void main()
{
  setuid(geteuid()); // make real uid = effective uid.
  system("/bin/ls -l");
}
```

It should be noted that using `setuid(geteuid())` to turn the real uid into the effective uid is not a common practice in `Set-UID` programs, but it does happen.

Now, remove the `setuid(geteuid())` statement from the above program, and repeat your attack. Can you gain the root privilege? Please show us your experiment results. Please explain your lab results. (Hint: this problem can be considered similarly how the LD PRELOAD environment variable vulnerability happens in Lab 1).

2.4 Task 3: Where is the Vulnerability Comes From?

The vulnerability is from the Bash source code: line-351 in `variables.c`. Please google and figure out why the parse_and_execute() will cause security vulnerability.

2.5 Task 4: Questions

This is a writing task, please answer the following questions in your report:

1. According to secure software principles and rules, what is the fundamental problem of the Shellshock vulnerability? What can we learn from this vulnerability?

3 Submission

You need to submit a detailed lab report to describe what you have done and what you have observed, including screenshots and code snippets. You also need to provide explanation to the observations that are interesting or surprising. You are encouraged to pursue further investigation, beyond what is required by the lab description. Your can earn bonus points for extra efforts (at the discretion of your instructor).

Copyright