- **first what registers to use**

```
man 2 syscall
```

```
x86-64 syscall rax rax rdx - 5
x86-64 rdi rsi rdx r10 r8 r9 -
```

- next what is the sys call number

```
less /usr/include/x86_64-linux-gnu/asm/unistd_64.h
```

```
#define __NR_read 0
#define __NR_write 1
#define __NR_open 2
#define __NR_close 3
#define __NR_stat 4
#define __NR_fstat 5
#define __NR_lstat 6
```

- **Now Code ASM**
- `sudo apt install nasm`

```
global main

SECTION .data
msg: db "Hello World from x86_64 machine", 0Ah, 0h
len_msg: equ $ - msg

SECTION .text
main:
    mov rax,1
    mov rdi,1
    mov rsi,msg
    mov rdx,len_msg
    syscall          ; write(1, msg, len_msg);

    mov r15, rax
    mov rax,60
    mov rdi,r15
    syscall          ; exit(write_len)
```

- **Compile it**

```
nasm -f elf64 syscall64.nasm
ld syscall64.o --entry main -o my64syscall
```

```
If you added sbrk, you will need to link libc:
ld -Map=output.map syscall64.o --entry main -o my64syscall  /lib/x86_64-linux-
gnu/libc.a
```

- Trace SysCalls of a Command

```
strace ./my64syscall

execve("./my64syscall", ["./my64syscall"], 0x7ffc3d5ff870 /* 59 vars */) = 0
write(1, "Hello World from x86_64 machine\n"..., 33Hello World from x86_64
machine) = 33
exit(33)                                      = ?
+++ exited with 33 +++
```