

# Monitorando Syscalls

+ O **strace** intercepta e registra as chamadas de sistemas que determinado programa faz

-----ass.c-----

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

```
int main (){  
    printf("Desec Security\n");  
    return 0;  
}
```

→ Montamos um programa em C que executa as mesmas funções

→ Se compararmos os dois com o strace, poderemos ver as syscalls chamadas

```
(root@DESKTOP-NJHHNK6)-[/home/kali]  
# strace ./ass  
execve("./ass", ["../ass"], 0x7ffc4941bc50 /* 33 vars */) = 0  
[ Process PID=234505 runs in 32 bit mode. ]  
write(1, "Desec Security\n", 15Desec Security  
) = 01:22:15.486 [ ] [debug] autos: = ?  
exit(0) :07:08.498 [ ] [debug] autos: = ?  
+++ exited with 0 +++  
05-01-22:09:08.471 [ ] [debug] autos: = ?  
  
(root@DESKTOP-NJHHNK6)-[/home/kali]  
# strace ./ass2  
execve("./ass2", ["../ass2"], 0x7ffcd7d51540 /* 33 vars */) = 0  
write(1, "Desec Security\n", 15Desec Security  
) = 01:22:15.531 [ ] [debug] autos: = ?  
exit(0) :15:08.514 [ ] [debug] autos: = ?  
+++ exited with 0 +++  
[ ] [debug] autos: = ?
```

→ Ao monstarmos o programa em c que faz a mesma coisa, vemos que ele será do formato 64-bits

```
(root@DESKTOP-NJHHNK6)-[/home/kali]  
# nano ass.c  
  
(root@DESKTOP-NJHHNK6)-[/home/kali]  
# gcc ass.c -o ass  
  
(root@DESKTOP-NJHHNK6)-[/home/kali]  
# file ass  
ass: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=6cf5d116f5b67e6288807e10e952d81023649, for GNU/Linux 3.2.0, not stripped  
  
(root@DESKTOP-NJHHNK6)-[/home/kali]  
# ./ass  
Desec Security
```

→ Se compararmos os tamanhos, vemos que os programas em Assembly são bem menores

→ Além disso, os em c se usam de libs para se comunicar com a máquina, enquanto o outro já comunica diretamente

```
(root@DESKTOP-NJHHNK6)-[/home/kali]
# ls -l *ass
-rwxr-xr-x 1 root root 15952 May  1 22:10 ass
05-01 22:10:08.478] [debug] autos:

(root@DESKTOP-NJHHNK6)-[/home/kali]
# ls -l *ass*
-rwxr-xr-x 1 root root 15952 May  1 22:10 ass
-rw-r--r-- 1 root root  180 May  1 14:27 ass.asm
-rw-r--r-- 1 root root   75 May  1 22:10 ass.c
-rw-r--r-- 1 root root  624 May  1 14:27 ass.o
-rwxr-xr-x 1 root root 8848 May  1 21:53 ass2
-rw-r--r-- 1 root root  271 May  1 21:45 ass2.asm
-rw-r--r-- 1 root root  864 May  1 21:52 ass2.o
-rw-r--r-- 1 root root 1801 Feb 26 10:53 labpasswd
05-01 22:21:08.530] [debug] autos:

(root@DESKTOP-NJHHNK6)-[/home/kali]
# ldd ass
linux-vdso.so.1 (0x00007ffd13eee000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f8848cfd000)
/lib64/ld-linux-x86-64.so.2 (0x00007f8848ef9000)
05-01 22:27:08.474] [debug] autos: (gdb)

(root@DESKTOP-NJHHNK6)-[/home/kali]
# ldd ass2
not a dynamic executable
05-01 22:29:05.450] [debug] autos: (gdb) si
in _main ()
```

→ Se quisermos debugar o código em c veremos o quão grande ele é

```
strace ./ass
```

```
objdump -d -M intel ass
```

```
ltrace ./ass
```

```
ltrace -S ./ass
```