H4ck 1n TN

Injection d'objets partagés

Olivier Dautricourt

February 11, 2016

Le processus de compilation

- Analyses (lexicale, syntaxique, sémantique)
- Génération de code intermédiaire, optimisations...
- Génération des fichiers objets
- Edition des liens

Edition des liens

- Objectif : créer un éxécutable à partir de fichiers objets (.o,.obj,.a)
- Les fichiers objets contiennent du code et la table de leurs symboles
- Trois cas:
  - Edition de liens entre plusieurs objets internes au programme
    ex: A.o $\rightarrow$ B.o $\rightarrow$ C.a
  - Edition statique de liens avec des objets externes (sur l'OS)
    ex: A.o $\rightarrow$ /usr/lib/x86_64-linux-gnu/libcrypt.a
  - Edition dynamique de liens avec des objets externes (chargés à l'exécution)
    ex: A.o $\rightarrow$ /lib/i386-linux-gnu/libc.so.6

Petits rappels de compilation

Les bibliothèques dynamiques

Petits rappels de programmation

Injection d'objets dynamiques dans un programme

TP

Où ?

- Sur Linux: .so (Shared object), emplacements: /lib, /usr/lib, /usr/local/lib
- Windows: .dll (Dynamic link library), C:\Windows\system32
- OSX: who cares ?

Quelques commandes utiles

- file <executable>: permet de savoir si le programme utilise des librairies dynamiques
- ldd <binaire>: affiche la liste des dépendances dynamiques
- ltrace <executable>: permet d'intercepter les appels à des librairies dynamiques
- readelf -s <binaire>: affiche la liste des symboles.
- gcc -shared -fPIC prog.c -o prog.so : compiler un .so sur linux/gcc

Petits rappels de compilation

Les bibliothèques dynamiques

Petits rappels de programmation

Injection d'objets dynamiques dans un programme

TP

L'interface dlfcn.h

- void *dlopen(const char *filename, int flag);
- char *dlerror();
- void *dlsym(void *handle, const char *symbol);
- int dlclose(void *handle);
- linkage avec -ldl (ou utiliser __libc_dl*)

En Python

Voir le module ctypes

Utilisation d'un pointeur sur fonction

```
typedef void (*fonction_ptr) (int arg1, ...);

void *handle = dlopen("/path/to/.so", RTLD_LAZY);

char *error = dlerror();

if(error){ printf(error); return; }

fonction_ptr fptr =
(fonction_ptr)dlsym(handle, "fonction");

if (fptr) { fptr(13); }

dlcose(handle);
```

Petits rappels de compilation

Les bibliothèques dynamiques

Petits rappels de programmation

Injection d'objets dynamiques dans un programme

TP

Les différentes méthodes

But

Modifier le comportement d'un programme ou rajouter des fonctionnalités.

Approche sur Windows

- Modification de la clef de registre HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\AppCertDLLs
- Utilisation de la fonction CreateRemoteThread : charge un programme dans le même espace mémoire qu'un autre processus.

Les différentes méthodes

Approche sur Linux

- LD_PRELOAD, variable d'environnement permettant de charger arbitrairement une librairie dans un programme (possibilité de remplacement)
  Avantages: très difficile à détecter et à interdire.
- ptrace, permet notamment d'injecter des objets dans un processus en cours d'exécution (sous certaines conditions)

LD_PRELOAD: How to

On a un exécutable "lol":

```c
#include <stdio.h>

int main(){
    puts("LOL!");
}
```

> gcc lol.c -o lol
> ./lol
> LOL!

LD_PRELOAD: How to

On a une librairie "hook.so":

```c
#include <stdio.h>

int puts(const char *str){
    fprintf(stdout, "Hooked!\n");
}
```

> gcc -shared -fPIC -o hook.so hook.c
> LD_PRELOAD=./hook.so ./lol
> Hooked!

LD_PRELOAD: How to

On peut aussi retrouver la fonction originelle

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

int puts(const char *str){
  fprintf(stdout, "Hooked!\n");
  typedef int (*real_puts_ptr) (const char*);
  real_puts_ptr real_puts = dlsym(RTLD_NEXT,"puts")
  real_puts(str);
}
```

```
> gcc -shared -fPIC -o hook.so hook.c -ldl -D_GNU_SOURCE
> LD_PRELOAD=./hook.so ./lol
> Hooked!
> LOL!
```

LD_PRELOAD: How to

Ou appeler une fonction au lancement du programme

```c
#include <stdio.h>

__attribute__((constructor))
void on_init(){
    printf("Started!\n");
}
```

> gcc -shared -fPIC -o onInit.so onInit.c
> LD_PRELOAD=./onInit.so ./lol
> Started!
> LOL!

Petits rappels de compilation

Les bibliothèques dynamiques

Petits rappels de programmation

Injection d'objets dynamiques dans un programme

TP

Challenges

A vous de jouer !

Le challenge est disponible ici → https://goo.gl/bmeJxn
et pour la rentrée :D → http://tinyurl.com/zf8sax6, avec de la lecture:

- https://www.0x0ff.info/2014/hook-lib-linux-ld_preload/
- https://www.nth-dimension.org.uk/pub/BTL.pdf
- https://blog.maleadt.net/2015/02/25/sudo-escalation/
- http://haxelion.eu/writeup/Crackme_In_Memory_Bruteforce/ (crackme du même type)
- https://www.exploit-db.com/papers/13233/ (orienté reverse)