INSA CVL
STI
2SU-5A
Sokhna Mai FALL

M.KAUFFMANN

TD3 : SpeedRun

Pratique:

- Création fichier Dockerfile

*Document 1 sans titre | log.txt | Dockerfile

```
FROM ubuntu:18.04
ENV LC_CTYPE C.UTF-8
RUN dpkg --add-architecture i386 && \
apt-get update && \
apt-get install -y strace ltrace curl wget gcc net-tools vim gdb python python3 python3-pip wget
git make procps libpcre3-dev libdb-dev libxt-dev libxaw7-dev python-pip libc6:i386
libncurses5:i386 libstdc++6:i386 && \
(wget -q -O- https://github.com/hugsy/gef/raw/master/scripts/gef.sh | sh) && \
pip install capstone requests pwn r2pipe  && \
pip3 install unicorn capstone ropper keystone-engine && \
mkdir tools && cd tools && \
git clone https://github.com/JonathanSalwan/ROPgadget && \
git clone https://github.com/radare/radare2 && \
cd radare2 && sys/install.sh && \
wget https://developer.arm.com/-/media/Files/downloads/gnu-rm/7-2018q2/gcc-arm-none-eabi-7-2018-q2-
update-linux.tar.bz2?revision=bc2c96c0-14b5-4bb4-9f18-bceb4050fee7?
product=GNU%20Arm%20Embedded%20Toolchain,64-bit,,Linux,7-2018-q2-update
```

- Build de notre docker

```
97550K .......... .......... .......... .......... .......... 99% 11.6M 0s
97600K .......... .......... .......... .......... .......... 99% 7.87M 0s
97650K .......... .......... .......... .......... .......... 99% 11.9M 0s
97700K .......... .......... .......... .......... .......... 99% 11.7M 0s
97750K .......... .......... .......... .......... .......... 99% 11.3M 0s
97800K .......... .......... .......... .......... .......... 99% 12.0M 0s
97850K .......... .......... .......... .......... .......... 99% 11.5M 0s
97900K .......... .......... .......... .......... .......... 99% 11.6M 0s
97950K .......... .......... .......... .......... .......... 99% 11.6M 0s
98000K .......... .......... .......... .......... .......... 99% 8.64M 0s
98050K .......... .......... .......... .......... .......... 99% 12.1M 0s
98100K .......... .......... .......... .......... .......... 99% 11.2M 0s
98150K .......... .......... .......... .......... .......... 99% 11.7M 0s
98200K .......... .......... .......... .......... ..         100% 14.1M=41s

021-12-08 14:36:55 (2.36 MB/s) - ‘gcc-arm-none-eabi-7-2018-q2-update-linux.tar.
z2?revision=bc2c96c0-14b5-4bb4-9f18-bceb4050fee7?product=GNU Arm Embedded Toolc
ain,64-bit,,Linux,7-2018-q2-update’ saved [100600407/100600407]

emoving intermediate container 3bf04fa10fdd
---> 040a2daadbb8
uccessfully built 040a2daadbb8
uccessfully tagged ubuntu18:ctf
ser@optiplex-1504:~/Sec_Emb$
```

- Image docker crée

```
user@optiplex-1504:~/Sec_Emb$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
ubuntu18     ctf       040a2daadbb8   52 seconds ago   1.65GB
ubuntu       18.04     5a214d77f5d7   2 months ago     63.1MB
```

- Lancement de notre docker avec docker run

```
user@optiplex-1504:~/Sec_Emb$ sudo docker run --rm -v $PWD:/pwd --cap-add=SYS_PTRAC
   --security-opt seccomp=unconfined -p 5555:5555 -i ubuntu18:ctf
```

- Vue détaillée de notre docker

```
user@optiplex-1504:~$ sudo docker ps
[sudo] Mot de passe de user :
CONTAINER ID   IMAGE          COMMAND   CREATED          STATUS          PORTS
                                  NAMES
5eeda244e1de   ubuntu18:ctf   "bash"    10 minutes ago   Up 10 minutes   0.0.0.0:5
5->5555/tcp, :::5555->5555/tcp   upbeat_poincare
user@optiplex-1504:~$
```

- connection à container

```
user@optiplex-1504:~$ sudo docker exec -it 5eeda244e1de /bin/bash
root@5eeda244e1de:/#
```

- Speedrun :

```
user@optiplex-1504:~$ sudo docker exec -it 5eeda244e1de /bin/bash
root@5eeda244e1de:/# nc speedrun-001.quals2019.oooverflow.io 31337
bash: nc: command not found
root@5eeda244e1de:/# apt-get install netcat
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  netcat-traditional
The following NEW packages will be installed:
  netcat netcat-traditional
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 65.1 kB of archives.
After this operation, 157 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu bionic/universe amd64 netcat-traditional amc
4 1.10-41.1 [61.7 kB]
Get:2 http://archive.ubuntu.com/ubuntu bionic/universe amd64 netcat all 1.10-41.1 [
436 B]
```

- Activer et lance le speedrun

```
root@5eeda244e1de:/pwd# chmod +777 speedrun
chmod: cannot access 'speedrun': No such file or directory
root@5eeda244e1de:/pwd# chmod +777 speedrun-001
root@5eeda244e1de:/pwd# ls
Dockerfile  out      program.c  speedrun-001
log.txt     program  rootfs     vmlinuz-qemu-arm-2.6.20
root@5eeda244e1de:/pwd# ./speedrun-001
Hello brave new challenger
Any last words?
Alarm clock
```

- Vérification des protections avec checksec

```
gef➤  checksec
[!] Command 'checksec' failed to execute properly, reason: No current process:
ust name one.
gef➤  checksec /pwd/speedrun-001
[+] checksec for '/pwd/speedrun-001'
Canary                        : ✘
NX                            : ✓
PIE                           : ✘
Fortify                       : ✘
RelRO                         : Partial
gef➤
```

Comme vous pouvez le voir, le bit NX est activé, ce qui signifie que la pile n'est pas exécutable, donc le shellcode est un non-non. De plus, il n'y a pas de PIE (Position Independent Executable), ce qui signifie qu'il n'y a pas d'ASLR (Address Space Layout Randomization).

```
oot@5eeda244e1de:/pwd# file ./speedrun-001
./speedrun-001: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically
linked, for GNU/Linux 3.2.0, BuildID[sha1]=e9266027a3231c31606a432ec4eb461073e1ffa9
, stripped
oot@5eeda244e1de:/pwd#
```

Questions :

- Devenez root. Qu'est-ce qu'un attaquant peut faire une fois root ?

Root est le nom conventionnel de l'utilisateur qui possède toutes les permissions sur le système. Ainsi, Un attaquant possédant ce privilège équivaut à un utilisateur suprême, doté de fonctions supérieures et d'accès privilégiés, Donc il peut tout faite sur ce système.

- Qu'est-ce que je dois prendre en compte dans mon modèle d'attaque ?

La sécurité d'un système est basée sur le DIC (Confidentialité, Intégrité et Disponibilité) qui fait essentiellement la sécurité d'un système. Ainsi dans mon modèle d'attaque, Ce sont des facteurs en prendre en compte mais aussi les attaques multi-étapes et de ne pas se limiter à des sondes de détection pour surveiller un système.

- Comprendre le lien avec les bugs / cette méthode est-elle applicable dans le cas d'un use-after-free ? Pourquoi ?

Le bug est un défaut de conception à l'origine d'un dysfonctionnement et ce dysfonctionnement peut aller à des défauts mineurs à majeurs.

L’application de cette méthode dans un use-after-free peut se faire parce qu’use-after-free est lié à une utilisation incorrecte de la mémoire dynamique et si un programme n’efface pas un pointeur après avoir libéré un emplacement mémoire celle-ci peut être utiliser par un attaquant pour pirater un programme.

- Qu'est-ce que je peux faire pour diminuer / contrer les bugs ?

Les bugs sont des erreurs mineures qui peuvent parfois provoquer des catastrophes majeures. Ils dépendent le plus souvent de notre environnement de travail.
Parmi les pour diminuer les bugs, on a :

Vider la cache : rapide pour la correction mais c’est à faire régulièrement.
Auditer le système,
Reproduire des bugs et utiliser les logs.

Bonus : Quelle différence si les canary et l'ASLR sont présents ?

La présence de de l’ASLR aurait permis de placer aléatoirement des zones comme la pile …lorsqu’il est compilé avec le support PIE permettant ainsi à notre système de conduire l’attaquant vers une erreur de segmentation.
La présence de canary permettra d’éviter la modification de la mémoire en mémoire tampon changeant la position des valeurs régulièrement.