INSA CVL
STI
2SU-5A
Sokhna Mai FALL

M.KAUFFMANN

TD4 : ToolBox

Test des projets suivants :

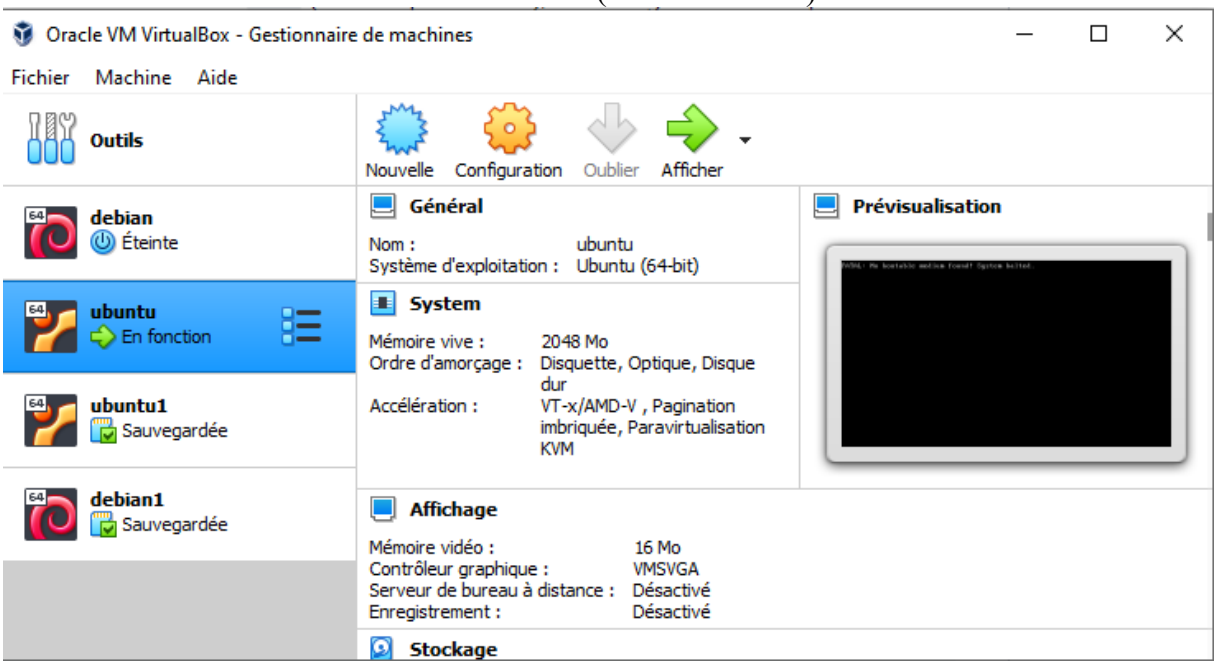
- USB

Lien projet Git : https://github.com/0x7ace80/virtualbox_usb_mitm

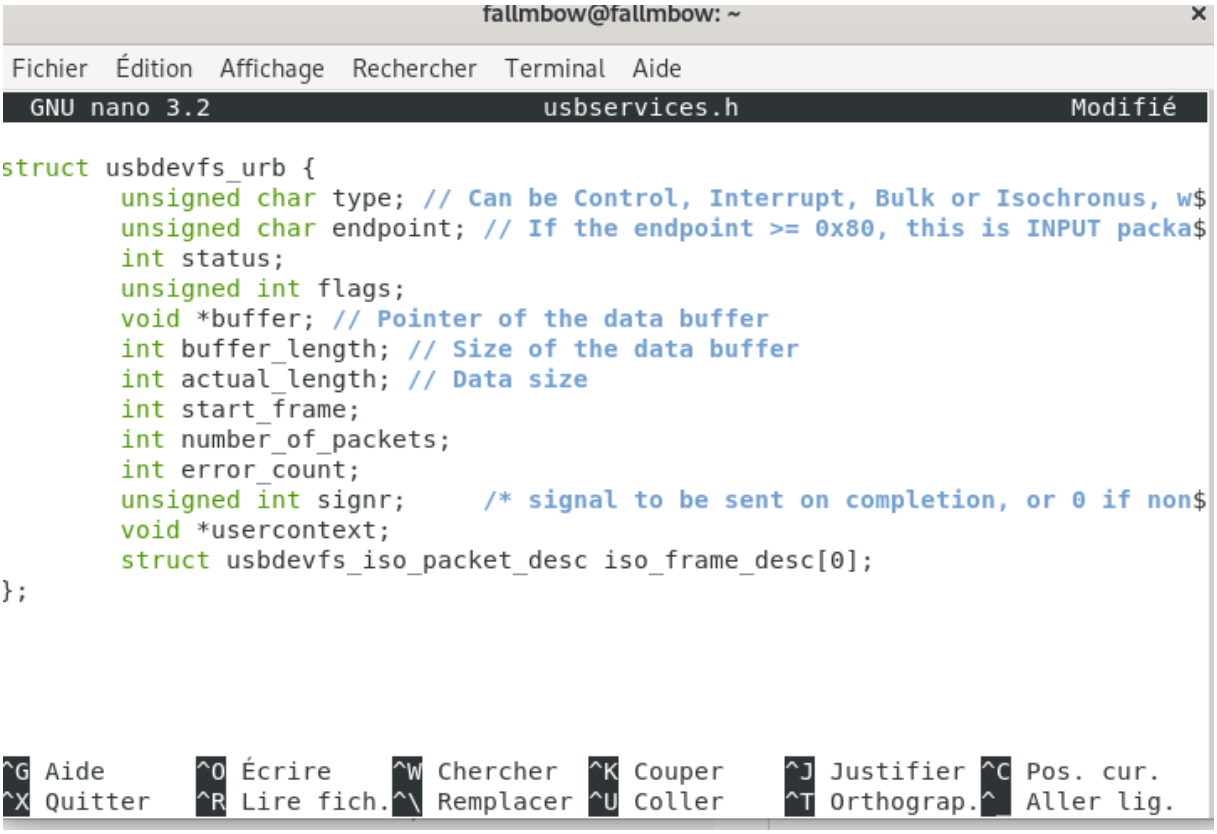
Pratique :

Le projet consiste à implémenter une fonction man in the middle sur un USB.

- Installation Virtual box (outils de travail)



- Création des fichiers



- Hack USB

Le périphérique USB virtualisé VBox et utilisez le périphérique USB virtualisé pour communiquer avec un périphérique USB réel sur le système d'exploitation hôte.
Le modèle de proxy de périphérique USB situé à :
src/VBox/Devices/USB/linux/USBProxyDevice-linux.cpp
La fonction que ce modèle de périphérique USB appellera pour émettre une requête USB est :
static DECLCALLBACK(int) usbProxyLinuxUrbQueue(PUSBPROXYDEV pProxyDev, PVUSBURB pUrb)
La structure qui contient le système URB est pUrbLnx->pKUrb. Et pUrbLnx est le format URB interne utilisé par VBox.

- Bluetooth

Lien Projet GIT : https://github.com/conorpp/btproxy

Pratique :
On travaille avec Debian.

- Installation Paquets

```
root@fallmbow:~# sudo apt-get install bluez bluez-tools libbluetooth-dev python-
dev
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  libbluetooth3 libc-dev-bin libc6-dev libexpat1-dev libpython-dev
  libpython2-dev libpython2.7-dev linux-libc-dev manpages-dev python2-dev
  python2.7-dev
Paquets suggérés :
  pkg-config glibc-doc
Les NOUVEAUX paquets suivants seront installés :
  bluez-tools libbluetooth-dev libc-dev-bin libc6-dev libexpat1-dev
  libpython-dev libpython2-dev libpython2.7-dev linux-libc-dev manpages-dev
  python-dev python2-dev python2.7-dev
Les paquets suivants seront mis à jour :
  bluez libbluetooth3
2 mis à jour, 13 nouvellement installés, 0 à enlever et 103 non mis à jour.
Il est nécessaire de prendre 39,1 Mo/40,2 Mo dans les archives.
Après cette opération, 87,6 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n]
```

- Lancement Btproxy

```
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
root@fallmbow:~# btproxy ff:ff:ff:ff:ff:ff 00:00:00:00:00:00
```

- Détecter adresse MAC

```
                    fallmbow@fallmbow: ~
 Fichier  Édition  Affichage  Rechercher  Terminal  Aide
root@fallmbow:~# hcitool scan
Device is not available: No such device
root@fallmbow:~# hcitool inq
Inquiring ...
Inquiry failed.: No such device
root@fallmbow:~#
```

NB : Normale parce que je n'avais pas branché de périphéries.

- Analyse plus avancée

```
fallmbow@fallmbow: ~
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
 GNU nano 3.2                    replace.py                         Modifié

otproxy <master-bt-mac-address> <slave-bt-mac-address># replace.py
def master_cb(req):
    """
        Received something from master, about to be sent to slave.
    """
    print '<< ', repr(req)
    open('mastermessages.log', 'a+b').write(req)
    return req

def slave_cb(res):
    """
        Same as above but it's from slave about to be sent to master
    """
    print '>> ', repr(res)
    open('slavemessages.log', 'a+b').write(res)
    return res


^G Aide       ^O Écrire     ^W Chercher   ^K Couper     ^J Justifier  ^C Pos. cur.
^X Quitter    ^R Lire fich. ^\ Remplacer  ^U Coller     ^T Orthograp. ^_ Aller lig.
```

- WIFI

Lien projet GIT : https://github.com/Tracks12/man-in-the-middle/blob/master/man-in-the-middle.py

Pratique :

On va travailler un programme python.

- Code Python

```
""" Mode Terminal du program """
def console():
    print("> " + color.GREEN + "Initiate TERM mod_" + color.END)
    print("TCP Dump - by DHS Team {DevilHatSec}\nProgramed by 4N4RCHY and z3r0.")

    while(1):
        print("\n\t\tTCP Dump Command PORT\n")
        print(color.GREEN + "\t1. SAMPLE\n\t2. ALL\n" + color.END + color.YELLOW + "\t3. FTP\n\t4. HTTP\n\t5. HTTPS\n\t6. SMTP\n\t7. POP3\n\t8. HELP\n\t9. MORE\n\t10. INTE
        choice = input("> Select Dump mod: ")

        if(choice == 1): dump_list()
        elif(choice == 2): dump_all()
        elif(choice == 3): dump_ftp()
        elif(choice == 4): dump_http()
        elif(choice == 5): dump_https()
        elif(choice == 6): dump_smtp()
        elif(choice == 7): dump_pop3()
        elif(choice == 8):
            print("> " + color.B_YELLOW + "Get Help_" + color.END)
            print("Aide:\t\t\t\t\n[SAMPLE], lance tout simplement tcpdump\n[ALL], lance la commande traduit en ascii sur tout les ports\n[FTP], lance la commande que sur l

        elif(choice == 9):
            print("> " + color.B_YELLOW + "Get More Info_" + color.END)
            print("More:\n\n\tProgram Created by 4N4RCHY and z3r0.\n\tVersion_1.0-b\n\n\tProgramer:\n\t\t4N4RCHY <anarchy.dar97@gmail.com>;\n\t\tz3r0 <mail>;")

        elif(choice == 10):
            print("> " + color.B_YELLOW + "Choose interface_" + color.END)
            print("Interface: \n\n" + color.GREEN + "\t0. eth0\n\t1. wlan0\n" + color.END)
```

- Exécution

```
> Please, select start mod:

        1. console mod
        2. graphique mod
        0. quit

>>> Mod: 1
> Initiate TERM mod_
TCP Dump - by DHS Team {DevilHatSec}
Programed by 4N4RCHY and z3r0.

                TCP Dump Command PORT

        1. SAMPLE
        2. ALL
        3. FTP
        4. HTTP
        5. HTTPS
        6. SMTP
        7. POP3
        8. HELP
        9. MORE
        10. INTERFACE
        0. QUIT

> Select Dump mod:
```

- NFC

Lien Projet : https://github.com/fmeum/WearAuthn#nfc

Pratique :

- Installation

```
fallmbow@fallmbow: ~
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
root@fallmbow:~# sudo apt install git libudev-dev make pkg-config
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  git-man libdpkg-perl liberror-perl libfile-fcntllock-perl libudev1 patch
  udev
Paquets suggérés :
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk
  gitweb git-cvs git-mediawiki git-svn debian-keyring gcc | c-compiler
  binutils bzr make-doc ed diffutils-doc dpkg-dev
Les NOUVEAUX paquets suivants seront installés :
  git git-man libdpkg-perl liberror-perl libfile-fcntllock-perl libudev-dev
  make patch pkg-config
Les paquets suivants seront mis à jour :
  libudev1 udev
2 mis à jour, 9 nouvellement installés, 0 à enlever et 101 non mis à jour.
Il est nécessaire de prendre 9 375 ko/10,8 Mo dans les archives.
Après cette opération, 42,6 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n]
```

- Téléchargement de notre projet

```
root@fallmbow:~# git clone https://github.com/amluto/u2f-hidraw-policy
Clonage dans 'u2f-hidraw-policy'...
remote: Enumerating objects: 25, done.
remote: Total 25 (delta 0), reused 0 (delta 0), pack-reused 25
Dépaquetage des objets: 100% (25/25), fait.
root@fallmbow:~# cd u2f-hidraw-policy
root@fallmbow:~/u2f-hidraw-policy#
```

- Compilation de notre projet

Pour la compilation et l'exécution de notre projet nous allons utiliser les commandes make et make install permettant ainsi d'avoir une vue sur la sécurité de notre NFC.

- RJ45

Lien Projet : https://www.youtube.com/watch?v=5YI-9eabPSo

Pratique :
Le projet est réalisé grâce à une vidéo YouTube.
Il nous montre une attaque MITM (Man In The Middle) en passant avec les ports RJ45.

- HARDWARE

Lien Projet : https://sepiocyber.com/blog/mitm-hardware-attacks/

Explication :
C'est plutôt une publication qui nous explique l'attaque hardware tout en proposant une solution.
La publication nous informe que la plupart des attaques hardware est au niveau des guichets automatiques surtout quand il s'agit d'une attaque avec l'homme du milieu.
Les guichets automatiques sont des cibles de choix pour les attaques matérielles MiTM grâce à l'abondance d'argent qui y est stocké. Une façon dont cette attaque peut être effectuée est par une attaque de boîte noire. Dans cette attaque, un appareil (contenant généralement un ordinateur Raspberry Pi Zero W) se connectera entre le PC du guichet automatique et le distributeur. Cela permet à l'attaquant d'envoyer des commandes de distribution d'espèces à la machine.

Questions :

- Quel sont les critères qui rendent une vulnérabilité critique ?

  Nous avons eu à remarquer les critères suivants pour qu'une vulnérabilité soit intéressante à exploiter :

  - Toucher un grand nombre de cibles ;
  - Être simple à exploiter ;
  - Avoir un but lucratif direct (ransomware) ou indirect (vol de données à des fins de revente ou de chantage).

- Suivant ces critères quelle interface devrait être testée en premier ? Pourquoi ?

Suivant ces critères, l'interface qui doit être testé en premier est la confidentialité parce qu'il va permettre de limiter l'accès de tout un chacun à une information donnée.

- Quels avantages, inconvénients ont les outils que vous avez choisi ?

Au cours de ses projets, j'ai eu à travailler avec VirtualBox, Python et des paquets linux.

VirtualBox permet d'exécuter autant de systèmes d'exploitation que vous le désirez dans des environnements virtuels (VE). Cette application permet de faire fonctionner n'importe quelle version de Windows, n'importe quelle distribution Linux, OS/2, FreeBSD, NetWare, Solaris. L'utilisateur dispose donc de plusieurs machines tout en exploitant une seule configuration. Cela évite les partitionnements natifs et l'usage de logiciels multiboot qui pourraient réfréner les débutants. Cependant l'utilisation des machines virtuelles diminue les performances de la machine réelle.

Python est le langage le plus utilisé pour la Data Science. Pour cause, ce langage est simple, lisible, propre, flexible et compatible avec de nombreuses plateformes. Il permet le prototypage rapide, et le code peut être exécuté n'importe où : Windows, MacOs, UNIX, Linux… sa flexibilité permet de prendre en charge le développement de modèles de Machine Learning, le forage de données, la classification et bien d'autres tâches plus rapidement que les autres langages. Cependant Python est typé dynamiquement. Cela signifie que vous n'avez pas besoin de déclarer le type de variable lors de l'écriture du code bien que cela soit facile pour les programmeurs pendant le codage, cela peut augmenter les erreurs d'exécution.