Travaux Pratiques n°5

Etudes et configuration de Ssh et affichages distants

But du TP

Dans ce TP nous allons étudier la configuration d'un serveur `openssh` et l'utiliser pour accéder de manière sécurisée à des machines distantes et afficher le bureau graphique à distance.

Mise en place

Cette manipulation peut être effectuée sur chacun des postes des salles 410, 405 ou 102. Tous les PC peuvent être à la fois clients et serveurs, cependant durant toute la manipulation, il va être question d'un PC « serveur » et d'un PC « client ». Cela suppose que vous installez les services sur le « serveur » et que vous faites les accès à partir du « client ». Attention ce TP est particulièrement compliqué pour savoir « *où l'on est* ». Il va y avoir des affichages sur le serveur, côté machine virtuelle mais aussi hôte et sur le client aussi en hôte et en virtuel, tout cela transitant en clair ou en crypté.

Je déconseille donc, pour une fois, d'effectuer la manipulation de manière croisée : vous ne sauriez plus du tout ce que vous visualisez !

Comme vous devez être administrateur sur le système, vous devrez travailler sur une machine virtuelle. Ce TP suppose l'utilisation du disque virtuel `Debian1102-20220323-pm...`, Il faut donc créer une machine virtuelle comme indiqué en adaptant le nom de votre vm. Attention au paramètre supplémentaire !

```
createvm
la commande sans paramètre vous donne la syntaxe à utiliser.
createvm Debian1102-20220323-pm.....vdi nomVM VRDE
```

Ne pas changer l'adresse MAC, cela empêcherait la VM d'obtenir une adresse IP prévisible. On suppose aussi qu'un câble droit relie directement l'interface `en1` de votre PC au switch de la salle (SW102, SW410 ou SW405).

Faites démarrer votre VM, logez-vous en root ou en rt.

Dans tout le TP il va être question d'adresses IP et de noms de machines. Les exemples utilisent soit la salle 405, soit la 410, soit la 102 : il convient de tout ramener à la salle dans laquelle vous effectuez la manipulation.

I. Configuration ssh

Le serveur ssh est déjà installé et opérationnel sur votre machine, car il a été installé sur le disque de la machine virtuelle. Pour repartir de la base, nous allons le désinstaller.

1. Mise à neuf

Nous devons être logés en `root`.

Nous commençons par tout enlever, fichiers de configuration compris.

```
# apt-get remove --purge openssh-server openssh-client
```

Vérifions que tout a été enlevé :

```
# dpkg -l openssh*
```

Vous devriez voir que les paquets clients et serveurs ne sont pas présents dans le système.

On réinstalle tout à neuf

```
# apt-get update
# apt-get install openssh-server openssh-client
```

En fin d'installation les scripts indiquent la création de clés RSA, ECDSA et ED25519 (les clés DSA ne sont plus considérées comme fiables). Ce sont des clés privées et publiques qui vont permettre d'identifier de façon unique et sûre le serveur `sshd`.

Lister le répertoire `/etc/ssh/`. Vous y voyez un fichier de configuration par défaut du client `ssh` (`ssh_config`), un fichier de configuration du serveur `sshd` (`sshd_config`).

Les autres fichiers sont de types `ssh_host_*_key` et `ssh_host_*_key.pub`. Les `.pub` sont des clés publiques, elles ont vocation à être partagées et distribuées alors que les autres sont des clés privées qui ne doivent être connues que du serveur `sshd` (observez-en les droits).

2. Identification de l'hôte

Lorsque vous vous connecterez au serveur `sshd`, il vous sera demandé d'attester que le serveur est le bon et votre client ssh recevra la clé publique du serveur qu'il mettra dans le fichier `~/.ssh/known_hosts`.

a) Premier contact

À partir de l'autre PC de la paillasse (le « client ») connectez-vous en `rt` sur la machine dont vous venez de configurer le serveur `sshd`. Testez les différentes étapes et vérifiez la présence de la clé publique dans le fichier `known_hosts`.

```
ssh rt@adresseIpMachDist
```

b) Plusieurs clés de contact

Reconnectez-vous plusieurs fois en mettant soit le nom de la machine distante, soit son nom complet soit un alias. Vous verrez qu'à chaque fois le fichier `~/.ssh/known_hosts` s'allonge d'une ligne.

Le nom de la machine n'est pas présent, il est remplacé par un `hashage`. Pour voir la clé (donc la ligne) qui est issue d'un contact avec un hôte particulier, il faut utiliser une commande adaptée.

```
ssh-keygen -F nomDhôte
```

Pour effacer cette ligne c'est assez facile avec la commande qui suit :

```
ssh-keygen -R nomDhôte
```

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
38:30:c3:5f:18:ef:d3:60:de:ec:18:c8:88:c3:5d:15.
Please contact your system administrator.
Add correct host key in /home/rt/.ssh/known_hosts to get rid of this message.
Offending RSA key in /home/rt/.ssh/known_hosts:67
RSA host key for 10.4.110.108 has changed and you have requested strict checking.
Host key verification failed.
```

Listing 1: Message d'alerte lors d'un `login` quand la clé d'hôte du serveur ssh a changé

c) On a changé la serrure ?

Si lors d'une connexion au serveur ssh, le client vous indique quelque chose du type de ce qui est affiché dans le listing n°1 page 3, c'est que la clé du serveur a été modifiée ou que vous êtes victime d'un « man in the middle ». Si ce n'est pas une attaque, il vous faut effacer la ligne (et donc la clé publique) présente dans le fichier `known_hosts`. Dans l'exemple il s'agit de la clé 67.

Pour obtenir ce message refaire sur le serveur ssh les deux étapes du paragraphe I. 1. d'abord le `remove` puis l'`install`.

Si malgré la ré-installation de `sshd`, vous n'avez pas l'affichage montré sur le Listing n°1 page 3, c'est que dans les nouvelles distributions d'`openssh`, par défaut, on

ne bloque pas le login dans le cas d'une mauvaise clé. Le logiciel pose une question, et la reposera à chaque prochain login. Il vaut mieux avoir un serveur `ssh` strict, c'est-à-dire qui refuse la connexion s'il y a incohérence entre les différentes clés.

Dans le fichier `/etc/ssh/sshd_config`, du serveur, ajoutez la ligne

`StrictHostKeyChecking=yes`

Si le paramètre `StrictHostKeyChecking` est déjà présent modifiez sa valeur à « `yes` ».

Redémarrez le serveur `sshd`.

Une fois que vous avez réussi à obtenir le message de refus, effacez la clé incriminée par « `ssh-keygen -R` » et vérifiez que vous pouvez vous reloger.

3. Les clés du client

Nous pouvons nous connecter facilement avec le mot de passe, mais pour faire mieux, nous allons maintenant configurer le `login` par clé.

a) Fabrication de vos clés

La commande pour les générer :

```
ssh-keygen
```

Acceptez les noms des fichiers proposés pour sauvegarder les clés. Avoir des noms de fichiers originaux interdit de les utiliser facilement.

Les options par défaut sont plutôt correctes. Choisissez une *passPhrase* pour protéger la clé privée, que vous pouvez mémoriser facilement, il va falloir la fournir plusieurs fois. Dans la vraie vie, c'est elle qui vous protègera en cas d'intrusion sur votre machine.

Dans le répertoire `~/.ssh/` vous devez avoir en plus les fichiers `id_rsa` et `id_rsa.pub`, l'un correspondant à la clé privée et l'autre à la clé publique.

b) Les clés autorisées

Il faut que la machine sur laquelle vous allez vous loger par `ssh` connaisse les clés publiques nécessaires (contenues dans les `.pub`) et les autorise pour le compte sur lequel vous voulez vous loger.

Côté client[1]

- copiez la clé publique du fichier `~/.ssh/id_rsa.pub` …

Côté serveur[2]

- … dans le fichier `~/.ssh/authorized_keys`

1 Il peut être plus facile d'utiliser le petit utilitaire `ssh-copy-id`.

2 Si c'est sur le compte de root que vous voulez copier la clé publique, il n'est pas possible de vous loger en root avec le mot de passe. Il peut être pratique de copier la clé sur le compte de rt, puis de la copier dans le fichier `/root/.ssh/authorized_keys`.

- faites bien attention que lors de la copie il n'y ait pas des caractères supplémentaires comme des retours à la ligne ou des « \ » !

Maintenant logez-vous de la machine cliente vers la machine serveur : elle va vous demander la *passPhrase* pour pouvoir lire la clé privée sur votre client, mais pas le mot de passe du compte sur le serveur.

c) Appelez un agent !

Si vous devez vous connectez fréquemment ou faire des accès fréquents par `scp`, à chaque fois il va falloir fournir la *passPhrase* : c'est lassant, il est alors tentant d'en avoir une trop simple et si vous la tapez en public, les autres la connaîtront rapidement.

Une solution pour la fournir une fois mais pas chaque fois est l'usage d'un `agent-ssh`. Attention cela veut aussi dire que si quelqu'un a accès à votre compte lorsque l'agent a déverrouillé votre clé, il n'aura pas non plus de *passPhrase* à fournir !

On démarre l'agent soit dans un nouveau `shell` ou un nouveau terminal, soit dans le `shell` courant comme ici :

```
eval $(ssh-agent -s)
```

L'agent est maintenant démarré avec les bons paramètres, mais il ne connaît pas encore la bonne identité à fournir :

```
ssh-add
```

Donnez la *passPhrase*, c'est tout !

Maintenant quand vous voulez vous loger avec la même identité sur le même compte de la machine distante (le serveur `sshd`), vous n'avez plus de *passPhrase* à fournir.

Le vérifier.

4. Creusons un tunnel ssh

Dans la suite, il va falloir traverser le réseau de façon discrète : nous allons le faire au moyen d'un tunnel `ssh`.

Au moyen de la commande affichée sur la figure n°1 page 6, l'utilisateur logé sur `machLoc` se loge à distance sur `machDist`. L'option `-L` permet de créer un tunnel entre le port local 2100 et le port 443 sur la machine `srvDist`. Seule la partie entre `machLoc` et `machDist` est cryptée par `ssh`, la partie `machDist` → `servDist` n'est pas cryptée par `ssh`. Elle l'est cependant probablement, car sur la figure, il s'agit de `https`.

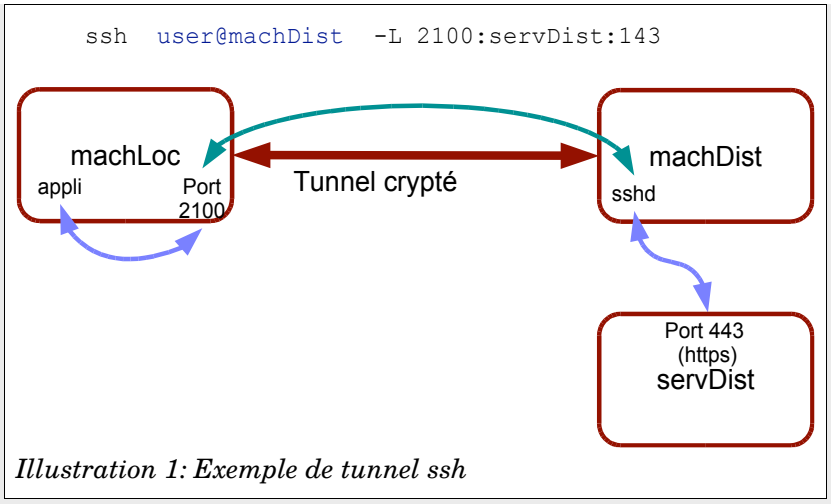

Illustration 1: Exemple de tunnel ssh

Sur votre machine locale, logez-vous sur votre machine serveur `ssh`.

```
ssh rt@ipDeVotreServeur -L 2100:centre.gtr.tp:443
```

On devrait ainsi voir sur le port 2100 en local, le site web de la machine `centre` en `https`. Vous aurez probablement à accepter le certificat de `centre`.

On se loge ainsi sur la machine distante et on voit le site d'une autre machine.

```
firefox https://127.0.0.1:2100
```

Vous pouvez cependant voir que `centre` a du mal à reconnaître votre machine, car le tunnel la déguise.

II. Affichage par RDP

Le protocole RDP (Remote Desktop Protocol) a été défini par Microsoft, pour pouvoir exporter à distance le « bureau » de Windows. Les dernières versions permettent de disposer sur le client de l'affichage du bureau, du clavier et de la souris, mais aussi du son, du copié/collé, des clés USB, des imprimantes… et crypte en TLS au besoin.

L'affichage par RDP va être possible de plusieurs façons mais cela va sembler un peu compliqué. En effet `VirtualBox` permet de voir la console et l'affichage complet à travers un affichage distant mais Windows le permet aussi et c'est assez facile d'effectuer l'affichage en Linux.

En résumé, nous allons essayer d'utiliser le protocole RDP (Remote Desktop, port 3389) de différentes manières.

1. Visualisation du bureau *VirtualBox*

Un petit mot sur l'application `rdesktop`. Sur les machines virtuelles vous avez l'application communautaire qui exporte bien écran, clavier souris, le reste est difficile. Sur la machine physique vous avez `rdesktop-vrdp`, la même application, mais améliorée par Oracle pour fonctionner avec `virtualBox`, et qui est distribuée avec `virtualBox`.

a) Linux visualise VB

Au moyen de l'application `rdesktop-vrdp`[1], on vous demande de visualiser l'affichage de `virtualBox` sur la machine hôte cliente. Faites la même chose avec `rdesktop` sur la machine virtuelle cliente. Le manuel de `rdesktop` devrait vous renseigner sur les rares options nécessaires. Attention la machine virtuelle dont vous devez visualiser l'affichage n'est peut-être pas configurée complètement pour ça. Activez le service de « bureau à distance » de virtualBox, ce qui devrait avoir été fait si vous avez ajouté le paramètre `VRDE` à la commande `createvm`.

Notez bien que, dans les deux cas, l'affichage est géré par le serveur **hôte**.

Dans le test fait sur la VM, on observe une drôle d'image. Expliquez !

C'est un peu facile ! Oui c'est un peu facile de pirater la liaison entre les deux machines.

b) C'est plus discret

Pour rendre la chose plus confidentielle, il va falloir établir un tunnel `ssh` entre le serveur et le client, en vous logeant du client sur le serveur avec `ssh` et en faisant passer la communication RDP à travers le tunnel `ssh` créé avec l'option `-L`. Effectuez le test en tunnel à partir de la machine physique ou de l'autre machine de la paillasse. Utilisez la syntaxe vue au paragraphe I. 4. page 5

Montrez avec `wireshark` que la liaison est effectivement cryptée par `ssh`.

2. Linux visualise Windows7

On va utiliser le serveur RDP de Win7[2]. Fabriquez pour cela une nouvelle VM en Windows 7 comme indiqué dans la note.

Vérifiez que le mode réseau de la machine virtuelle est en « *bridge* » (ou pont), activer le partage de bureau windows et indiquer dans Win7 que `rt` peut utiliser ce service (probablement déjà fait).

Visualisez le bureau Windows à partir de la machine virtuelle Linux du PC client : dans un premier temps en direct puis à travers un tunnel `ssh` entre le PC client et le serveur.

1 Il est normalement déjà installé, car il fait partie du paquet `virtualBox`, sur la machine **physique**.

2 Pour lancer une machine Win7 facilement utilisez la commande `createvm Win7_32p.Windows7.MEM1400.VRAM128.IOAPIC.VTX.NP.SATA.en1.vdi votreVmWin7` modifiez l'adresse MAC de cette nouvelle VM pour qu'elle soit différente de celle de l'autre VM. rt/rt est administrateur.

III. Affichages aveugles

Pour bien appréhender la puissance des techniques présentées, nous allons utiliser des machines virtuelles « serveur » sans « tête » (*headless*) et malgré tout « voir » le bureau à distance.

1. On démarre dans le noir

Pour être bien sûr de ne pas être pollué par des effets de bord, créez une nouvelle machine virtuelle Linux par `createvm`.

La faire démarrer sans interface graphique :

```
VBoxManage startvm maVm --type headless
```

Quand elle a fini de démarrer, vérifiez les ports ouverts de la machine server sur l'hôte et la virtuelle.

```
(pc client)# nmap -sS addresseIPserverHôte
(pc client)# nmap -sS addresseIPserverVirtuel
```

Les ports 3389 et 590x ne devraient pas être ouverts mais le port 22 oui.

2. Ssh et X11

Pour commencer vérifions que X11 passe bien à travers ssh

```
ssh -X rt@adresseIPserveurVirtuel
```

Lancer une application comme `xclock` ou `gedit` et vérifiez qu'elle s'affiche bien. Si vous omettez le `-X` l'affichage ne devrait pas être possible.

3. Un serveur RDP

a) Liaison directe

Logez vous en `root` (d'abord en rt puis avec un « su - ») sur la machine virtuelle qui est démarrée en *headless* et installez `xrdp`.

```
# apt-get update ; apt-get install xrdp
```

Vérifiez quel nouveau port est maintenant ouvert. Connectez-vous en `rdp` donc avec `rdesktop-vrdp` : vous pouvez apprécier la vitesse d'affichage. Si vous trouvez que l'affichage est petit, ajoutez `-f` à la commande `rdesktop-vrdp`, vous aurez alors le plein écran.

b) Liaison cryptée

Une question : la liaison est en clair ou en crypté ?

Trouvez la technique pour qu'elle soit cryptée.

4. Le serveur VNC

Nous allons utiliser cette fois-ci un serveur `vnc` directement. Il faut redémarrer la VM en mode « *headless* ».

Débarrassons-nous de ce que nous venons d'installer sur le serveur :

```
# apt-get remove --purge vnc4server xbase-clients xrdp
```

Et installons le serveur VNC

```
# apt-get install tightvncserver
```

Il faut lancer le serveur à la main (il est configuré au niveau de chaque utilisateur).

```
tightvncserver
```

Répondre aux questions du mot de passe. Vous pouvez le lancer plusieurs fois cela vous fera plusieurs affichages, le premier sur :1 le second sur :2 et ainsi de suite.

Sur la VM du PC client installez le client vnc

```
# apt-get install xtightvncviewer
```

et le lancer vers l'écran « **:1** »

```
xvncviewer adresseIPserverVirtuel:1
```

Rentrez le mot de passe et vous avez accès au bureau à distance : c'est un bureau virtuel (pas sur la carte vidéo physique) dans une machine virtuelle.

Il peut y avoir un problème : quand vous vous connectez, vous voyez un écran gris et pas du tout le bureau de la VM. Il y a un défaut d'initialisation. Retournez sur la VM en *Headless* en ssh et tuez le serveur vnc et relancez-le, comme ceci :

```
tightvncserver -clean -kill :1
tightvncserver
```

Remplacez dans cette dernière commande le « 1 » par le numéro du *display* que vous voulez réinitialiser.

Si vous avez tout fait, bon retour dans le monde réel

Fichier *TP-5_sshAff-2324.odt* imprimé le 9 juin 2024