La sécurité avec SSH

Atelier ccTLD Dakar, Sénégal

Hervey Allen

Contenu

- Où obtenir le Shell Securisé (Secure SHell)
- Activation et configuration de SSH
- Des clients SSH et SCP pour Windows
- Authentification du serveur auprès du client (clés hôte / host keys)
- Implication du changement de la clé de l'hôte
- Authentification du client auprès du serveur par mot de passe
- Authentification du client auprès du serveurs par chiffrement à clé publique (RSA/DSA)
- Pratique de scp/sftp, échange de clés

Applications et couches de chiffrement

Nous avions précédemment évoqué les applications suivantes correspondant à différentes couches réseau:

- Couche lien (2) — Chiffrement PPP, WEP
- Couche réseau (3) — IPSEC
- Couche transport (4) — TLS (SSL)
- Couche application — SSH, PGP/GPG

Couche de sécurité applicative SSH

Dans cette section nous couvrirons SSH au niveau de la couche applicative, en vue de faire à la fois de l'authentification et du chiffement de données.

Nous allons ignorer les problèmes de clé RSA version 1 avec SSH Version 1 car RSA1 and SSH version 1 et 1.5 ne sont plus sûrs.

Préoccupations principales

SSH vise directement ces trois aspects de la sécurité:

- **Confidentialité**
  - Protéger nos données des yeux indiscrets
- **Authentification**
  - Cette personne est-elle bien celle qu'elle prétend être ?
- **Authorisation**
  - Cette personne a-t-elle le droit de faire ce qu'elle veut faire ?

Où obtenir SSH ?

D'abord voir si SSH existe sur votre système:

```
ssh -V
```

Si vous désirez une version à jour de OpenSSH (la version actuelle est 4.1) vous pouvez aller voir:

sur FreeBSD: /usr/ports/security/openssh-portable/
http://www.openssh.org/ (version libre)
http://www.ssh.com/ (version originale, commerciale)

OpenSSH est recommandé pour FreeBSD.

La version distribuée avec FreeBSD 5.4 est OpenSSH Portable portable 3.8.1p1

Activation et configuration d'OpenSSH

Sur nos machines c'est déjà fait, mais si vous aviez fait:

```
cd /usr/ports/security/openssh/; make install
```

- Vous devriez vous assurer que rc.conf est configuré:

```
sshd_enable="YES"
```

- Regardez /etc/ssh/ssh_config et /etc/ssh7sshd_config. Dans /etc/ssh/sshd_config, remarquez:

```
PermitRootLogin yes/no  # ("no" est à préférer)
```

et dans /etc/ssh/ssh_config (cela pourrait poser des problèmes):

```
Protocol 1,2   # (il ne doit y avoir que "2")
```

Il y a *beaucoup* d'options dans ssh_config et sshd_config. Il est recommandé de les lire, même si vous ne comprenez pas tout, pour voir si rien ne vous semble anormal.

Des clients SSH pour windows

Il y a plusieurs clients SSH, libres et partagiciels, pour Windows:

Voir http://www.openssh.org/windows.html pour une liste.

Quelques uns qui implémentent SSH version 2:

- Putty: http://www.chiark.greenend.org.uk/~sgtatham/putty/
- OpenSSH pour Windows (via Cygwin):
  http://www.cygwin.com/
  http://sshwindows.sourceforge.net/

•Commerciaux:

- Client SSH de ssh.com (utilisation personnelle):
  http://www.ssh.com/products/ssh/download.cfm
- SecureCRT: http://www.vandyke.com/products/securecrt/
- Et WRQ sur http://www.wrq.com/products/reflection/ssh/
  si vous voulez payer.

Des clients SSH pour windows (suite)

- Voir aussi Filezilla pour le transfert SCP

Quelques références SSH utiles

- Une très bonne introduction aux clés SSH RSA/DSA, en trois parties, par l'ancien directeur de Gentoo.org, Daniel Robbins, sur le site IBM Developer Works.

- Les trois parties:

  Gestion de clés OpenSSH, Partie 1
  http://www-106.ibm.com/developerworks/library/l-keyc.html
  Gestion de clés OpenSSH, Partie 2
  http://www-106.ibm.com/developerworks/library/l-keyc2/
  Gestion de clés OpenSSH, Partie 3
  http://www-106.ibm.com/developerworks/library/l-keyc3/

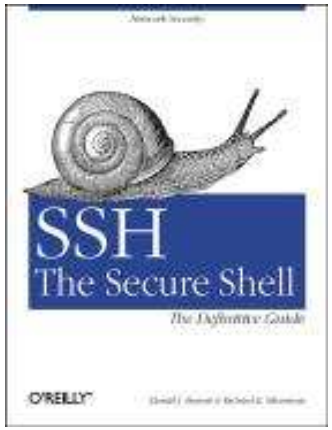
Des références supplémentaires

Pour une comparaison de SSH v1 et v2 voir:

http://www.snailbook.com/faq/ssh-1-vs-2.auto.html

Un très bon livre à consulter:

SSH, The Secure Shell
The Definitive Guide,
seconde édition,
par Daniel J. Barrett,
Richard Silverman, et
Robert G. Byrnes
Mai 2005, ed. O'Reilly
ISBN: 0-596-00895-3



Méthodes de connexion SSH

Un certain nombre de choses se passent quand vous utilisez SSH pour vous connecter depuis votre machine (client) vers une autre machine (serveur):

- La clé publique du serveur est renvoyée au client, et si elle ne s'y trouve pas déjà, ajoutée à . ssh/known_hosts. Sinon, elle y est consultée et vérifiée.

- Une fois le serveur vérifié, soit:
  - Le mot de passe de l'utilisateur est demandé
  - Si votre clé publique se trouve sur le serveur dans .ssh/authorized_keys, un échange de clé RSA/DSA prend place et vous devez taper votre phrase de code pour déchiffrer votre clé privée.

Conseils SSH

Vous avez un choix pour les types de clé d'authentification – RSA ou DSA

Les fichiers importants:

/etc/ssh/ssh_config
/etc/ssh/sshd_config
~/.ssh/id_dsa et id_dsa.pub
~/.ssh/id_rsa et id_rsa.pub
~/.ssh/known_hosts
~/.ssh/authorized_keys
Et remarquer les clés de serveur /etc/ssh_host_*

Lisez *en entier* les pages de manuel de ssh et sshd: man ssh, man sshd

Authentification SSH

La clé privée peut-être protégée par une phrase code:

- Soit il faut la taper à chaque connexion
- Ou on peut utiliser “ssh-agent” qui garde une copie de la clé en mémoire pour éviter de devoir la taper

Pas besoin de changer les mots de passe sur des dizaines de machines!

Désactivez complètement les mots de passe:

/etc/ssh/ssh_config

```
#   PasswordAuthentication yes
PasswordAuthentication no
```

Pour des raisons historiques, il y a *trois* types de clés SSH différentes:

SSH1 RSA, SSH2 DSA, SSH2 RSA

Attaque “homme au milieu”

La première fois que vous vous connectez à une machine, sa clé est stockée dans ~/.ssh/known_hosts

La prochaine fois que vous vous connectez, si la clé à changé, c'est peut-être que quelqu'un est en train d'intercepter la connexion!

Ou peut-être que quelqu'un a réinstallé/mis à jour le serveur, et remplacé là clé. A vous de trouver la cause...

Vous serez averti si la clé change.

Echange de clés d'hôte

Première connexion avec SSH:

```
ssh utilisateur@pc1.cctld.sn
The authenticity of host 'pc1.cctld.cn (196.1.97.123)' can't be
established.
DSA key fingerprint is 91:ba:bf:e4:36:cd:e3:9e:8e:92:26:e4:57:c4:cb:da.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'pc1.cctld.sn, 196.1.97.123' (DSA) to the list
of known hosts.
utilisateur@pc1.cctld.sn's password:
```

A ce moment, le client ajoute dans ~/.ssh/known_hosts la clé publique de pc1.cctld.sn (qui se trouve dans /etc/ssh/ssh_host_dsa_key.pub sur le serveur)

Connexion suivante:

```
[root@mamachine]$ ssh utilisateur@pc1.cctld.sn
utilisateur@pc1.cctld.sn's password:
```

La machine est mantenant “connue”.

Echange de clés d'hôte (suite)

Commande	Type de clé générée	Clé publique
ssh-keygen -t rsa	RSA (SSH protocol 2)	id_rsa.pub
ssh-keygen -t dsa	DSA (SSH protocol 2)	id_dsa.pub

- La taille par défaut de la clé est de 1024 bits
- Les clés publiques sont en texte clair
- Les clés privées sont chiffrées si vous utilisez une phrase code (mais stockée en texte)

Le fichier utilisé sur le serveur pour l'échange de clé est “authorized_keys”.

Echange de clés d'hôte (suite)

Comment SSH décide-t-il quels fichiers comparer ?

Regarder dans /etc/ssh/sshd_config. Pour OpenSSH version 3, le serveur est par défaut configuré en protocole 2 .

Un client SSH version 2 tente de se connecter dans l'ordre suivant:

clé RSA version 2
clé DSA version 2
authentification à base de mot de passe (même si une clé RSA version 1 est presente)

Faire attention au réglage "HostKeyAlgorithms" dans /etc/ssh/ssh_config, qui controle l'ordre d'utilisation des protocoles – sinon on peut forcer l'ordre depuis le client via les paramètres en ligne de commande (man ssh)

SSH – la “phrase magique”

Concept de base pour comprendre comment une connexion SSH est faite en utilisant une clé RSA/DSA

- Client X contacte le serveur Y sur le port 22.
- Y génère un nombre aléatoire et le chiffre en utilisant la clé publique de X. La clé publique de X déjà se trouver sur Y.
- Le nombre chiffré est renvoyé à X.
- X déchiffre le nombre aléatoire en utilisant sa clé privée et le renvoie à Y.
- *Si le nombre déchiffré correspond au nombre chiffré d'origine, alors une connexion chiffrée est établie.*
- Le nombre chiffré envoyé de Y to X est la “phrase magique” (“Magic Phrase”)

Nous essayerons d'illustrer ceci...

SSH - Exercices

Nous allons pratiquer ces concepts:

- L'utilisation du fichier .ssh/known_hosts
- connexion SSH avec auth. par de mot de passe
- génération d'une clé RSA protocole version 2
- copie de la clé publique (comment faire ?...)
- connexion par clé privée avec phrase code en utilisant l'authentification par échange de clé.
- utilisation de scp avec authentification par clé RSA
- quelques “trucs” ssh sans mot de passe.

SSH – Exercices (suite)

L'utilisation du fichier known_hosts

Connect to the machine next to your machine using ssh:

```
ssh admin@pcN.cctld.sn
```

Si c'est votre première connexion à la machine, vous devriez voir:

```
pc1# ssh admin@pc2.cctld.sn
The authenticity of host 'pc2.cctld.sn(196.1.97.123)' can't be established.
RSA1 key fingerprint is 60:f7:04:8b:f7:61:c4:41:6e:9a:6f:53:7d:95:cb:29.
Are you sure you want to continue connecting (yes/no)?
```

Répondez “yes” ici, mais nous discuterons des implications. Comment éviter ce problème ? Y-a-t-il un risque d'attaque “homme au milieu” ? Quel fichier est créé ou mis à jour ? Pourquoi ?

SSH – Exercices (suite)

Connexion ssh avec authentification par mot de passe

A l'invite où vous avez répondu “yes”, on vous a demandé de taper le mot de passe de l'utilisateur admin pour pc2.cctld.sn:

```
host1# ssh admin@pc2.cctld.sn
The authenticity of host 'pc2.cctld.sn (196.1.97.123)' can't be established.
RSA2 key fingerprint is 60:f7:04:8b:f7:61:c4:41:6e:9a:6f:53:7d:95:cb:29.
Are you sure you want to continue connecting (yes/no)? yes
```

Et vous auriez dû voir quelque chose de similaire:

```
Warning: Permanently added 'pc2.cctld.sn' (RSA2) to the list of known hosts.
                                        [/etc/ssh/ssh_host_key.pub]
admin@pc2.cctld.sn's password:
```

Vous êtes maintenant connecté de manière sécurisée à pc2. Nous discuterons de ce qui s'est déroulé pendant cette connexion.

SSH – Exercices (suite)

Generation de clé RSA1/RSA2/DSA

Nous allons générer une clé RSA SSH protocole 2 de 2048 bits. Pour faire ceci, nous allons – depuis votre machine – lancer la commande suivante.

```
ssh-keygen -t rsa -b 2048
```

Il vous sera demandé de préciser un emplacement là où la clé devra être stockée, ainsi qu'une phrase de code pour chiffrer le fichier de clé privé. Il est important d'utiliser une phrase code. Les clés sans phrase code peuvent être dangereuses si utilisées incorrectement. Nous parlerons de ceci quand nous auront complété cet exercice. A vous de choisir – et retenir! -- une phrase de code.

Si vous oubliez votre phrase code, il est impossible de déchiffrer la clé privée. Vous serez forcé de re-générer une nouvelle paire de clé et remplacer la clé publique sur tous les serveurs distants.

SSH – Exercices (suite)

Génération de clé RSA 2

Voici la sortie de la commande: “ssh-keygen -t rsa -b 2048”:

```
pc1# ssh-keygen -t rsa -b 2048
Generating public/private rsa key pair.
Enter file in which to save the key
(/admin/.ssh/id_rsa): [enter]
Enter passphrase (empty for no passphrase): [pw]
Enter same passphrase again: [pw]
Your identification has been saved in /
admin/.ssh/id_rsa.
Your public key has been saved in /
admin/.ssh/id_rsa.pub.
The key fingerprint is:
0f:f5:b3:bc:f7:5b:c8:ce:79:d0:b1:ab:2c:67:21:62
admin@pc1.cctld.sn
pc1#
```

SSH – Exercices (suite)

Copie de la clé publique

Maintenant que vous avez généré un jeu de clés RSA(2) vous pouvez commencer à en tirer profit. Nous allons copier la clé publique sur la machine sur laquelle nous nous étions connecté précédemment, et la stocker dans le fichier ~/.ssh/*authorized_keys*. Ensuite nous allons nous déconnecter et reconnecter pour voir la différence.

D'abord copier la clé vers la machine que vous aviez utilisé précédemment (pcX.cctld.sn):

```
cd ~/.ssh
scp id_rsa.pub admin@pcX.cctld.sn:/tmp/.
```

Il vous sera demandé de taper le mot de passe de l'utilisateur sur la machine sur laquelle vous allez vous connecter (dans ce cas ci, “admin” sur la machine “pcX.cctld.sn”).

SSH – Exercices (suite)

Copie de la clé publique

La sortie de la commande précédente devrait ressembler à ceci:

```
pc1# scp *.pub admin@pc2.cctld.sn:/tmp/.
admin@pc2.cctld.sn's password:
id_rsa.pub           100% |*****************************|   408       00:00

pc1#
```

La clé publique, qui vour servira à vous connecter par authentification à clé publique, est maintenant placée dans /tmp sur la machine sur laquelle vous allez vous connecter. L'étape suivante consiste à placer le contenu de la clé dans le fichier approprié, c'est à dire:

Copier le contenu de la clé publique RSA depuis /tmp/id_rsa.pub dans *~/.ssh/authorized_keys*

Vous pouvez tenter de le faire vous-même, sinon voyons la page suivante...

SSH – Exercices (suite)

Copie de la clé publique

Pour copier la clé publique au bon emplacement, faire:

```
ssh admin@pcN.cctld.sn
cat /tmp/id_rsa.pub >> ~/.ssh/authorized_keys
rm /tmp/id_rsa.pub
exit
```

Si vous ne comprenez pas ce que ces commandes font, nous en discuterons après. Il y a d'autres manières de copier la clé. A vous de choisir la méthode la plus adaptée.

Sur la page suivante vous allez vous connecter avec vos clés publiques / privées.

SSH – Exercices (suite)

Connexion via clé publique/privée

Pour vous connecter en utilisant votre clé RSA protocole 2 il suffit alors de taper:

```
ssh admin@pcn.cctld.sn
```

Et vous devriez voir (pc1 vers pc2 par exemple):

```
host1# ssh admin@pc2.cctld.sn
Enter passphrase for RSA key 'admin@pc1.cctld.sn':
```

C'est plutôt intéressant! Vous n'avez pas eu besoin de taper votre mot de passe pour le compte admin@pc2.cctld.sn. A la place, on vous demande la phrase code de la clé publique RSA protocole 2 que vous aviez créée avec la commande "ssh-keygen -t rsa -b 2048" - La phrase code sert à décoder le nombre aléatoire échangé entre les machines (rappelez-vous de la "Phrase Magique?").

Pourquoi la clé RSA a-t-elle utilisée, plutôt que le mot de passe ?

Réessayer avec ssh -v ...

SSH – Exercices (suite)

SCP avec clé publique/privée

D'abord se déconnecter de la machine:

```
exit
```

Maintenant essayons de copier un petit fichier (par exemple /etc/motd) avec la commande SCP (SeCure coPy):

```
scp /etc/motd admin@pcN.cctld.sn:/tmp/.
```

Qu'avez-vous remarqué? Vous devriez avoir remarqué qu'il ne vous est pas demandé de mot de passe, mais plutôt la phrase code de la clé publique RSA protocole 2.

C'est attendu -- SCP et SSH font partie du même logiciel- OpenSSH et utilisent tous deux les clé RSA et DSA de la même manière.

SSH – Exercices (suite)

Un autre outil SSH: SFTP

En plus de ssh et scp, un autre outil pour faire du transfert de fichier sécurisé: SFTP

Utilisons sftp pour récupérer le fichier /etc/motd depuis la machine de votre voisin et le copier dans /tmp sur votre machine:

```
sftp admin@pcN.cctld.sn
```

Une fois connecté:

```
sftp> lcd /tmp     [changer le répertoire local à /tmp]
sftp> cd /etc      [changer le répertoire distant à /etc]
sftp> get motd     [récupérer /etc/motd dans /tmp/motd]
sftp> ?            [aide en ligne]
sftp> bye          [fin de connexion]
ls /tmp/motd       [vérifier que le fichier est bien arrivé]
```

SSH – Exercices (suite)

Utilisons la flexibilité de scp...

Copier tout un répertoire et les fichiers dedans:

Copions tous les fichier depuis /usr/ports/palm vers votre machine avec une seule commande (1.4Mo de contenu):

```
scp -r /usr/ports/palm/* admin@pcN.cctld.sn/tmp/.
```

- "-r" pour copie récursive
- "/tmp/." pour placer les fichiers dans /tmp/ sur la machine du voisin.

SSH – Exercices (suite)

Utilisons encore scp!

(Exercice facultatif)

Copier un fichier depuis une machine distante vers une autre.

Copions /etc/fstab sur la machine de votre voisin de gauche vers /tmp/fstab.copy sur la machine de votre voisin de droite:

```
scp  admin@pcGAUCHE.cctld.sn:/etc/fstab
   admin@pcDROITE.cctld.sn/tmp/fstab.copy
```

- Si le mot de passe de amin est le même sur les deux machines, il n'y a besoin que de le taper une fois.
- Avez-vous remarqué que nous avons changé le nom du fichier sur la destination ?

SSH: Conclusion

SSH, SCP, et SFTP sont des outils remarquables pour se connecter à distance et copier des fichiers, ceci de manière entièrement sécurisée.

Nous recommandons que vous supprimiez telnet et FTP de votre système. Ou tout au moins n'utiliser que le FTP anonyme.

Vous pouvez utiliser – deuxième partie de cette présentation – ssh pou créer des “tunnels” sécurisés entre deux machines.

Rappel: utiliser les références fournies pour obtenir des informations plus détaillées – entre autres “`man ssh`” et “`man sshd`” pour plus d'information.