Utilisation de ssh

Linux Gazette numéro 61

Matteo Dell'Omodarme

matt@martine2.difi.unipi.it

Jérôme Fenal

jerome@fenal.org

Nicolas Prigent

PrigentN@thmulti.com

1. Introduction

Chaque fois que nous nous connectons par telnet à une machine distante, les données de la connexion traversent le réseau local, ce qui donne à un intrus éventuel la possibilité d'épier la connexion, et éventuellement d'y insérer des commandes malicieuses. L'utilisation de systèmes de chiffrement fort permettra une énorme amélioration de la sécurité du réseau.

Nous apprenons de la page de manuel de ssh les points suivants : « Ssh (Secure Shell) est un programme pour se connecter sur une machine distante et exécuter des commandes sur une machine distante. Il est destiné à remplacer les commandes rlogin et rsh, et à fournir une communication sécurisée par chiffrement entre deux machines non maîtrisées à travers un réseau peu sûr. Les connexions X11 ainsi que celles entre des ports TCP arbitraires peuvent aussi être realisées à travers le canal sécurisé ». Ssh est un programme puissant et très facile à utiliser, qui utilise un chiffrement fort pour protéger toutes les données confidentielles qui transitent, y compris les mots de passe.

Il existe à l'heure actuelle deux protocoles SSH, connus sous les noms de SSH2 et SSH1, le premier étant une amélioration du protocole SSH1. SSH2 supporte maintenant d'autres méthodes d'échange de clés que l'échange par double chiffrement RSA. La distribution actuelle met en oeuvre l'échange de clés Diffie-Hellman[1] et autorise l'utilisation de DSA ainsi que d'autres algorithmes de chiffrement à clé publique que RSA.

SSH2 peut être rendu compatible avec SSH1, mais ne l'est pas par défaut ; le serveur SSH2 seul ne sait pas gérer une connexion SSH1 et vous devez mettre en place un serveur SSH1 pour ce faire[2].

2. Obtention et installation de SSH

Vous pouvez obtenir les clients et serveurs SSH2 et SSH1 depuis le serveur FTP principal (http://www.ssh.com) ou depuis les sites miroirs. La dernière version en date de SSH1 est ssh-1.2.30.tar.gz, tandis que pour SSH2, vous pouvez télécharger ssh-2.3.0.tar.gz.

Le processus d'installation est très simple. La première étape est de dépaqueter les sources de SSH1 :

```
tar -zxf ssh-1.2.30.tar.gz
```

Cela créera un répertoire ssh-1.2.30. Allez dans ce répertoire et lancez le processus de configuration :

```
cd ssh-1.2.30
./configure
```

Le script *configure* transmet toutes les informations nécessaires à l'étape de la compilation, recherchant sur le système les bibliothèques et programmes requis. Quand le script a fini son travail, vous pouvez lancer la compilation

```
make
```

Après l'étape de compilation, passez administrateur et installez binaires, fichiers de configuration et clés du système en tapant :

```
make install
```

Cela installera les clients (scp1, ssh-add1, ssh-agent1, ssh-askpass1, ssh-keygen1, ssh1) dans le répertoire /usr/local/bin, et le serveur (sshd1) dans /usr/local/sbin. Remarquez que, dans /usr/local/bin, il existe des liens symboliques (sans le suffixe « 1 ») vers les exécutables.

La prochaine étape est d'installer SSH2. Les opérations requises sont les mêmes que pour SSH1 :

```
tar -zxf ssh-2.3.0.tar.gz
cd ssh-2.3.0
./configure
make
```

et en tant qu'administrateur :

```
make install
```

2.1. Compatibilité SSH1 - SSH2

Dans ce qui va suivre, nous allons supposer que SSH1 et SSH2 sont installés. Pour faire en sorte de rendre capable le serveur SSH2 de gérer une connexion SSH1, vous devez

éditer les fichiers de configuration de SSH2, qui se trouvent normalement dans le répertoire /etc/ssh2. Modifiez le fichier *sshd2_config*, fichier de configuration de sshd2 (Secure SHell Daemon) qui est le serveur pour ssh2. Ajoutez les lignes suivantes :

```
Ssh1Compatibility yes
Sshd1Path /usr/local/sbin/sshd1
```

Bien entendu, vous devez modifier l'information /usr/local/sbin/sshd1 en rapport avec le chemin de votre installation de sshd1. Ainsi, le serveur sshd2 transmettra les requêtes d'un client SSH1 à sshd1.

3. Démarrer SSH

Il y a essentiellement deux moyens de démarrer sshd à l'initialisation du système.

- Allez dans le répertoire /etc/rc.d, et modifiez le fichier *rc.local* en ajoutant à la fin les lignes :

  ```
  echo "Démarrage de sshd ...."
  /usr/local/sbin/sshd
  ```

  Ainsi, à la fin de la prochaine ré-initialisation de votre système, sshd sera lancé et le message « Démarrage de sshd ... » sera affiché à l'écran. Pour démarrer sshd sans redémarrer la machine, tapez à la ligne de commande[3] :

  ```
  /usr/local/sbin/sshd
  ```

- Vous pouvez aussi, sur les systèmes utilisant les scripts d'initialisation System V, ajouter le script *sshd2.startup*, disponible dans la distribution, dans le répertoire /etc/rc.d/init.d, en le renommant sshd2. Allez ensuite dans le répertoire rc$nombre, où $nombre est votre niveau d'exécution par défaut. Si vous ne le connaissez pas, recherchez dans le fichier /etc/inittab la ligne le spécifiant :

  ```
  id:5:initdefault
  ```

ou

```
id:3:initdefault
```

Dans le premier cas, votre niveau d'exécution est 5, dans le second, 3. Dans le répertoire /etc/rc.d/rc$nombre.d, lancez la commande :

```
ln -s ../init.d/sshd2 S90sshd2
```

Allez ensuite dans /etc/rc.d/rc0.d et lancez :

```
ln -s ../init.d/sshd2 K90sshd2
```

Faites de même dans le répertoire /etc/rc.d/rc6.d. Ceci fait vous pouvez démarrer sshd2 sans redémarrer la machine en lançant simplement le script[4] :

```
/etc/rc.d/init.d/sshd2 start
```

4. Établir une connexion SSH

Dès que sshd tourne sur votre machine, vous pouvez tester votre configuration en essayant de vous connecter avec le client ssh. Supposons que votre machine ait pour nom *serveur1* et que votre compte soit *mon_nom*. Pour démarrer une connexion ssh, utilisez la commande :

```
ssh -l mon_nom serveur1
```

Le client ssh2 essaie ainsi de se connecter au port 22 (port par défaut) de *serveur1*. Le démon sshd2, qui tourne sur *serveur1* accepte la requête et demande le mot de passe pour le compte *mon_nom*. Si le mot de passe est correct, il autorise la connexion et lance un interpréteur de commande.

4.1. Création et gestion des clés ssh

Ssh autorise un autre mécanisme d'authentification, basé sur des *clés d'authentification*, une méthode cryptographique à clé publique. Chaque utilisateur désireux d'utiliser ssh avec une authentification à clé publique doit lancer la commande **ssh-keygen** (sans option) pour créer les clés d'authentification. La commande commence la création de la paire de clés (publique et privée) et demande à l'utilisateur de saisir une phrase de passe pour les protéger.

Deux fichiers sont créés dans le répertoire $HOME/.ssh2/ : *id_dsa_1024_a* et *id_dsa_1024_a.pub*, respectivement clés privée et publique de l'utilisateur.

Supposons que nous ayons deux comptes utilisateurs, *mon_nom1* sur la machine *serveur1* et *mon_nom2* sur *serveur2*. Nous voulons nous connecter de *serveur1* à *serveur2* en utilisant l'authentification à clé publique. Pour ce faire, quatre étapes sont nécessaires :

1. Sur *serveur1*, créez la paire de clés en utilisant **ssh-keygen**, et choisissez une phrase de passe pour la protéger.
2. Connectez vous à *serveur2*, en utilisant l'authentification par mot de passe de ssh, et répétez l'opération précédente. Allez ensuite dans le répertoire $HOME/.ssh2 et créez le fichier nommé *identification*, contenant les lignes suivantes :

   ```
   # identification
   IdKey  id_dsa_1024_a
   ```

   Ce fichier est utilisé par sshd pour identifier la paire de clés à employer lors des connexions.
3. Depuis *serveur2*, récupérez la clé publique de votre compte sur *serveur1* et renommez là de façon à la reconnaître (par exemple : *serveur1.pub*) :

   ```
   ftp serveur1
   [...]
   cd .ssh2
   get id_dsa_1024_a.pub serveur1.pub
   ```

À la fin de la copie FTP, une copie de la clé publique de *serveur1*, nommée *serveur1.pub*, réside dans le répertoire $HOME/.ssh2 sur *serveur2*.

4. Créez le fichier des autorisations *authorization* contenant les lignes suivantes :

   ```
   # authorization
   Key     serveur1.pub
   ```

   Ce fichier énumère toutes les clés publiques ssh à qui l'utilisateur fait confiance placées dans le répertoire $HOME/.ssh2. Quand une connexion ssh est commencée par un utilisateur dont la clé publique figure dans le fichier d'autorisation, le processus d'authentification à clé publique commence.

Afin de tester la configuration précédente, vous pouvez essayer de vous connecter de *serveur1* à *serveur2* en utilisant ssh. Sshd doit répondre en demandant une phrase de passe, ou si le mot de passe du compte est demandé, quelque erreur a dû se produire dans le processus de configuration et vous devez contrôler soigneusement les étapes 1 à 4. La phrase de passe exigée est celle qui est *locale* (c.-à-d. celle protégeant la clé publique sur la machine *serveur1*).

5. Bientôt sur vos écrans...

Le prochain article présentera d'autres programmes et utilitaires de la suite ssh: ssh-agent et ssh-add (deux programmes utiles de gestion de phrase de passe), ainsi que sftp et scp (une manière sécurisée de transférer des fichiers à travers le réseau).

Paru dans le numéro 61 de la Linux Gazette, janvier 2001.

Traduction française par Jérôme Fenal <jerome@fenal.org>.

Notes

1. NdT : du nom des inventeurs
2. NdT : avec OpenSSH, pas de problème de ce point de vue là. Tout est faisable, ce n’est qu’une option de configuration à modifier
3. NdT : en tant qu’administrateur...
4. NdT : idem