Systèmes Informatiques
Travaux Pratiques – Séance N^{o} 6

Ce TP présente différents outils qui permettent de se connecter et de communiquer avec une machine distante. Sauf indication contraire, la machine distante que vous utiliserez sera `hivaoa`, serveur DEC du Centre de Ressources Informatiques de l'Université. Cette machine est similaire à `uranie`, quoique moins puissante. Surtout, elle ne partage pas son espace avec les machines du MIPS : pas de montages NFS, pas d'utilisation du système NIS. Vous disposez donc sur `hivaoa` d'un compte utilisateur indépendant (quant à votre mot de passe et votre espace utilisateur) de celui que vous utilisez habituellement sur les stations du MIPS ou `uranie`.

1 Communication directe et sécurisée avec une machine distante

Le protocole SSH (Secure Shell) a été conçu dans le but de sécuriser les communications entre machines distantes, *i.e.* empêcher toute utilisation frauduleuse des données échangées ou du canal de communication lui-même. Il permet d'établir des *tunnels* de communication entre machines dûment authentifiées et tels que les données circulant entre elles sont cryptées (ce qui garantit leur confidentialité).

SSH dispose de plusieurs méthodes de cryptage des données. Nous utiliserons la méthode RSA qui repose sur le couple *clé publique* et *clé privée*. La clé privée est gardée secrète alors que la clé publique correspondante est librement distribuée à tous les interlocuteurs. Les informations cryptées avec une clé privée ne peuvent être décryptées qu'avec la clé publique correspondante, et inversement, les informations cryptées avec une clé publique ne peuvent être décryptées qu'avec la clé privée correspondante.

1.1 Connexion à distance

La première utilisation que vous allez faire des outils SSH est l'ouverture d'une session sur une machine distante.

1.1.1 Interface textuelle

Lors des TP précédents, vous avez déjà eu à vous connecter sur une machine distante : il s'agissait d'`uranie` et vous aviez déjà utilisé la commande `ssh`. Vous allez renouveler l'expérience en vous connectant cette fois à `hivaoa`. Vous disposez sur ce serveur d'un compte dont le nom de login et le mot de passe sont, par défaut, ceux que vous utilisez sur les machines du MIPS.

1) Dans une nouvelle fenêtre Xterm, qui sera réservée à cela, essayez de vous connecter sur `hivaoa` à l'aide de la commande `ssh`.

La première étape du protocole est *l'identification* des machines souhaitant communiquer. Lorsque vous vous connectez pour la première fois à une machine distante, celle-ci vous soumet son identité par le biais de sa clé publique :

```
The authenticity of host 'hivaoa (134.59.1.59)' can't be established.
RSA key fingerprint is 5c:d1:14:27:9b:e6:80:10:6a:12:5e:a4:49:59:98:66.
Are you sure you want to continue connecting (yes/no)?
```

Afin d'établir une session vraiment sécurisée, vous devez comparer la clé proposée avec celle que vous aura communiquée, de façon sûre (oralement par exemple), l'administrateur de la machine distante. Si la clé est bien la même que celle figurant ci-dessus, répondez `yes`. La clé d'`hivaoa` est désormais stockée dans votre fichier `~/.ssh/known_hosts` et il ne vous sera plus redemandé de la valider :

```
Warning: Permanently added 'hivaoa,134.59.1.59' (RSA) to the list of known hosts
```

À partir de maintenant, tous vos échanges avec la machine distante sont cryptés. La seconde étape est de *vous* authentifier en tant qu'utilisateur valide de la machine distante en donnant votre mot de passe.

2) Votre session se déroule désormais sur `hivaoa` et vos commandes sont donc exécutées par le serveur. Vérifiez par exemple que l'espace utilisateur qui vous est alloué sur `hivaoa` est différent de celui dont vous disposez habituellement. .

Vous pourriez de la même façon vous connecter sur la machine de votre voisin ou, plus généralement, sur toute machine acceptant les connexions SSH et sur laquelle vous possédez un compte.

1.1.2 Interface graphique

Serveur X : Toute application visible sur votre écran (les clients `Xterm` ou `Emacs` par exemple) a obtenu du serveur X de votre machine la permission d'y afficher ses fenêtres.

En effet, dès l'instant de votre connexion, le serveur X garantit que les seules applications qu'il acceptera seront les vôtres, c'est-à-dire celles qui seront lancées depuis votre compte utilisateur sur la machine à laquelle vous êtes connecté. Cette garantie repose sur une *clé*, construite de manière aléatoire par `Xdm` (le gestionnaire d'affichage) et rangée dans un fichier d'autorisations qui vous est propre : `~/.Xauthority`. Chaque client graphique présente cette clé au serveur X pour qu'il accepte d'afficher ses fenêtres.

3) Par la commande `xauth`, essayez les différentes possibilités de faire apparaître les entrées du fichier d'autorisations. Par exemple, `list` donne les entrées sous forme lisible, `extract` en extrait une et la fournit dans un fichier (ou la sortie standard, notée « - »), `nextract` fait la même chose mais en la codant en hexadécimal.

Attention : cette commande admet deux modes de fonctionnement : en ligne de commande, ou en interactif. Si vous tapez simplement la commande `xauth` sans paramètres, vous vous trouvez en mode interactif, que vous devez terminer par la commande `quit`.

4) Envoyez par courrier à votre voisin l'entrée de votre fichier d'autorisations correspondant à votre terminal actuel, et demandez-lui alors d'afficher un `Xterm` (ou un autre client X) depuis son compte et sur votre écran. Utilisez la forme « lisible » de la clé, pour qu'elle puisse être transmise par courrier sans difficultés. D'une part, il devra ajouter votre entrée à son fichier d'autorisations, d'autre part, il devra préciser sur quel écran envoyer le client X grâce à l'option `-display`. Vous ferez vous-même les mêmes actions en sens inverse pour afficher un client X à vous sur son écran.

Transfert X : Un des très gros avantages de la connexion par `ssh` est qu'elle permet, grâce au mécanisme de transfert de port, d'ouvrir à distance des connexions X avec une bonne sécurité : tout ce qui chemine entre votre serveur X et la machine distante passe dans un *tunnel* crypté.

5) Sur votre PC, la variable d'environnement `$DISPLAY` identifie votre visuel. Affichez-en la valeur grâce à la commande `echo`.

6) Connectez-vous à `hivaoa` et supprimez toutes les clés contenues dans votre fichier d'autorisations `~/.Xauthority`.

7) Déconnectez-vous, puis reconnectez-vous à `hivaoa` par la commande `ssh -x`. L'option `-x` empêche la création du tunnel crypté pour la connexion au serveur X. Examinez le contenu de la variable d'environnement `$DISPLAY`. Essayez de lancer depuis `hivaoa` un client X simple, par exemple `xbiff`. Que se passe-t-il ? Pourquoi ? (Listez les clés de votre fichier d'autorisations)

8) Recommencez, mais cette fois en vous connectant normalement à `hivaoa` par `ssh`. Expliquez les différences de comportement.

1.2 Commandes Unix à distance

9) Déconnectez-vous de `hivaoa`. La commande `ssh` permet d'effectuer toute commande `Unix` sur une machine distante, à condition que l'on y dispose d'un compte, sans ouvrir de session intéractive. Utilisez `ssh` pour connaître le chemin absolu de votre compte sur `hivaoa`, pour visualiser le contenu de votre répertoire personnel sur `hivaoa` (affichez les fichiers cachés), puis pour afficher la liste de tous les utilisateurs connectés sur `hivaoa`.

Notez la différence d'interprétation d'une notation telle que `~` suivant qu'elle apparaît entre apostrophes ou non dans la commande passée à `ssh`. Comment expliquez-vous cette différence ?

10) De la même façon, sans ouvrir de session intéractive, vous pouvez lancer un client graphique sur une machine distante. Faites-le en lançant par exemple un client `xeyes`, depuis votre PC mais tournant sur `hivaoa`.

1.3 Utilisation de fichiers distants

Les mécanismes précédents ne vous permettent pas d'échanger des fichiers entre machines distantes. Pour cela, il existe d'autres outils.

1.3.1 Copies de fichiers à distance

`scp` fait partie de la famille des commandes SSH (le `s` initial permet de s'en rappeler). Cette commande permet la copie de fichiers entre deux machines distantes. La syntaxe est la même que la commande `cp`, le nom des fichiers distants étant simplement précédé de leur localisation :

`NomLogin@NomMachineDistante:NomFichier`

Notez que le nom de login n'a pas besoin d'être précisé si c'est le même.

11) Créez sur `hivaoa` un nouveau fichier de contenu quelconque (avec `cat` par exemple). Copiez-le sur votre compte local (dans le répertoire `~/SI/TP5/`), grâce à la commande `scp`.

12) En utilisant cette même commande `scp` et son option permettant la copie récursive, copiez votre répertoire `~/SI/TP2` local sur votre compte sur `hivaoa`, dans un répertoire nommé `TP`.

1.3.2 Transfert de fichiers SFTP

Le protocole SFTP est la version sécurisée (SSH) du protocole FTP. Il offre à l'utilisateur la même interface pour le même usage : transférer des fichiers entre deux machines distantes sur lesquelles il possède un compte. Vous constaterez que son fonctionnement peut s'avérer plus pratique que celui de la commande `scp`, puisque vous vous connectez réellement sur la machine distante, et pouvez en explorer la hiérarchie.

13) Utilisez `sftp` pour vous connecter à `hivaoa` alors que vous vous situez à la racine de votre répertoire personnel. La commande `help`, ou simplement `?`, vous donne la liste des commandes possibles, mais comme elles sont très nombreuses, mieux vaut vous reporter au mémento de cette semaine.

14) Affichez le contenu de votre répertoire personnel sur `hivaoa`. Déplacez-vous dans l'arborescence de `hivaoa`, dans la mesure du possible. De même, déplacez-vous dans le répertoire `~/SI/TP2` de la machine locale, au moyen de la commande `lcd`. Copiez alors un fichier d'`hivaoa` sur votre PC, puis copiez un autre fichier de votre PC sur `hivaoa`.

15) Recommencez l'exercice précédent dans le but de transférer tous les fichiers d'un répertoire. Vous devez effectuer un *transfert multiple*.

1.4 Connexion à distance sans mot de passe

Dans le protocole SSH, la seconde étape d'authentification de l'utilisateur (*cf.* Section 1.1.1) peut être simplifiée si ce dernier possède sa propre paire de clés RSA. Il suffit pour cela que l'utilisateur ajoute sa clé publique dans le fichier `~/.ssh/authorized_keys` des machines distantes auxquelles il

souhaite se connecter sans mot de passe. À chaque connexion, SSH utilise ce fichier pour décider s'il est nécessaire ou non de contrôler le mot de passe de l'utilisateur.

16) Créez une paire de clés RSA avec la commande `ssh-keygen -t rsa`. Utilisez une *passphrase* vide lorsque demandé.

17) Copiez le fichier contenant votre clé publique sur `hivaoa`.

18) Ajoutez maintenant votre clé publique au fichier `~/.ssh/authorized_keys`. Créez ce fichier si nécessaire, puis faites en sorte que seul le propriétaire du fichier ait les permissions de lecture et d'écriture.

19) Enfin, ouvrez une nouvelle session SSH sur `hivaoa`. Que remarquez-vous ?

2 Communication directe non sécurisée avec une machine distante

Les protocoles de communication non sécurisés tendent à disparaître au profit de leur version sécurisée présentée dans la section précédente. Cependant, il existe une application très répandue d'un de ces protocoles non sécurisés : les *serveurs FTP anonymes*. De plus, vous pouvez encore rencontrer des systèmes ne disposant pas des versions sécurisées.

2.1 FTP anonyme

`Ncftp` est une interface de bonne qualité avec le protocole FTP. En particulier, il utilise ce protocole en *mode passif*, d'une manière qui est acceptée par les serveurs FTP ordinaires. Il n'est cependant pas accepté entre les serveurs du MIPS et ceux du CRI, donc entre `uranie` et `hivaoa`.

Vous allez l'utiliser pour charger des fichiers situés sur des machines distantes, sur lesquelles vous ne possèdez pas de compte, et qui jouent le rôle de serveurs FTP anonymes. Il en existe des milliers dans le monde, ils servent en particulier à distribuer les logiciels libres, par exemple ceux du projet GNU.

20) Voici une listes de serveurs acceptant des connexions anonymes. La notation ci-dessous ne peut pas être utilisée telle quelle, ce qui suit le caractère `' :'` indique le répertoire à consulter. La commande `open` de `ncftp`, si elle n'a pas de paramètre, tente toujours d'ouvrir une connexion anonyme.

```
Allemagne - ftp.cs.tu-berlin.de:/pub/gnu
Allemagne - ftp.de.uu.net:/pub/gnu/
Allemagne - ftp.stw-bonn.de:/pub/mirror/ftp.gnu.org/
France - ftp.cs.univ-paris8.fr:/mirrors/ftp.gnu.org/
France - ftp.irisa.fr:/pub/mirrors/gnu
France - ftp.lip6.fr:/pub/gnu
Italie - ftp.oasi.gpa.it:/pub/gnu
Suisse - mirror.switch.ch:/mirror/gnu/
Royaume-Uni - ftp.mcc.ac.uk:/pub/gnu
Royaume-Uni - ftp.warwick.ac.uk:/pub/gnu/
Royaume-Uni - sunsite.org.uk:/gnu/
Royaume-Uni - ftp.hands.com/ftp.gnu.org/
```

Utilisez les commandes de `ncftp` pour consulter un de ces sites.

2.2 Connexion à distance

Telnet est un outil très pratique pour expérimenter les protocoles de communications. Nous utiliserons ici son fonctionnement par défaut qui est d'ouvrir une session intéractive sur une machine distante (l'équivalent de la commande `rlogin`).

21) Dans une nouvelle fenêtre Xterm essayez de vous connecter sur `hivaoa` à l'aide de la commande `telnet`. Vous voyez apparaître un message d'erreur qui vous montre que cette connexion est impossible.

22) Essayez de la même manière une connexion avec `uranie`. Le message d'erreur est différent, mais le résultat est le même.

23) Connectez-vous à `uranie` comme vous l'avez déjà fait, à l'aide de la commande `ssh`. De là, vous pouvez vous connecter à `hivaoa` par la commande `telnet`. Cette fois, tout va bien : vous voyez apparaître quelques lignes, qui vous annoncent que la connexion est établie, puis une invite pour vous identifier. Faites-le.

Vous voyez donc que les administrateurs des machines placent des barrières sur certains chemins, et que seuls certains outils permettent de franchir ces barrières.

24) Que constatez-vous si vous essayez d'exécuter un client graphique tel que `xeyes`.