Utiliser SSH pour se connecter/travailler à distance

Raymond Namyst (adapaté pour les besoins de l'IUT par Pierre Ramet)

27 octobre 2022

Ce petit guide a pour objectif de vous simplifier les sessions de travail à distance au département informatique de l'IUT, ou simplement le rapatriement de fichiers de travail. Un environnement UNIX est considéré dans cette note, vous pouvez utiliser le logiciel comme `Putty` sous Windows... Ce n'est aucunement un cours sur le chiffrement (a)symétrique d'échanges de données, mais plutôt un guide très pragmatique de commandes utiles.

Note 1 : Si vous disposez déjà d'une paire de clés privée/publique, vous pouvez passer à la section 2.

Note 2 : Il y a 2 passerelles pour accèder au département `info-ssh1.iut.u-bordeaux.fr` ou `info-ssh2.iut.u-bordeaux.fr`. **Pensez à ne pas toujours utiliser la même passerelle pour éviter les surcharges (`info-ssh2` est souvent moins utilisée...)** Une troisième machine `info-ssh3.iut.u-bordeaux.fr` est en cours d'installation.

Note 3 : **Toute la partie clé publique/privée (sections 1 à 3) n'est actuellement pas disponible pour des problèmes de montage NFS/Samba. Vous devrez donc saisir votre mot de passe UB à chaque connexion.**

1 Générer des clés SSH

SSH permet d'éviter l'authentification par mot de passe en utilisant un protocole cryptographique asymétrique s'appuyant sur une paire (clé privée, clé publique). La clé privée restera toujours stockée sur votre ordinateur, tandis que vous déposerez une copie de votre clé publique sur les serveurs auxquels vous souhaitez vous connecter sans taper de mot de passe.

Pour générer votre couple de clés (ici des clés RSA), utilisez la commande `ssh-keygen` :

```
[pbismuth@mymachine] ssh-keygen
```

La commande vous demande alors d'entrer une « *pass phrase* », c'est-à-dire un mot de passe qui protègera votre clé privée. Nous verrons dans la section suivante comment faire pour ne pas entrer ce mot de passe à chaque connexion SSH.

Une fois la *pass phrase* entrée, la commande `ssh-keygen` génère deux fichiers, `id_rsa` (clé privée) et `id_rsa.pub` (clé publique), dans le sous-répertoire `${HOME}/.ssh/`

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/pbismuth/.ssh/id_rsa):

Your identification has been saved in /home/pbismuth/.ssh/id_rsa.
Your public key has been saved in /home/pbismuth/.ssh/id_rsa.pub.
```

```
The key fingerprint is:
...
```

2 Faire confiance à son agent

Par défaut, SSH vous demandera d'entrer votre *pass phrase* à chaque fois qu'il aura besoin d'utiliser votre clé privée. Heureusement, votre agent SSH est là pour ça : ce processus vous demandera votre « *pass phrase* » en début de session, puis se chargera de l'accès à votre clé privée à chaque fois que le protocole SSH l'exigera.

Pour autoriser votre agent à récupérer votre clé privée, utilisez la commande `ssh-add` :

```
[pbismuth@mymachine] ssh-add -K
```

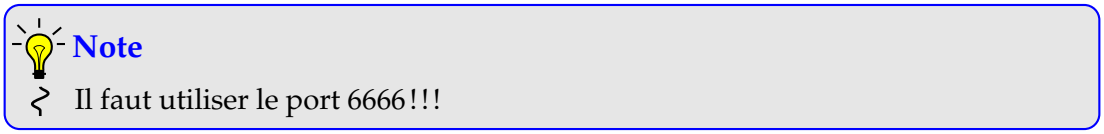
L'agent vous demande alors (une seule fois) votre « *pass phrase* », puis il travaillera discrètement pour vous en arrière plan.

3 Déposer sa clé publique sur le serveur

Il nous reste une étape à réaliser pour autoriser l'authentification sans mot de passe : déposer votre **clé publique** `id_rsa.pub` sur le serveur. Il faut pour cela ajouter votre clé publique au fichier `${HOME}/.ssh/authorized_keys` sur une machine passerelle du département, qui se nomment `info-ssh1.iut.u-bordeaux.fr` ou `info-ssh2.iut.u-bordeaux.fr`.

Vous pouvez soit effectuer cette manipulation manuellement, en copiant d'abord votre fichier `id_rsa.pub` à l'IUT (en utilisant typiquement la commande `scp`), puis en le concaténant au fichier `~/.ssh/authorized_keys`. Mais vous pouvez aussi utiliser la commande `ssh-copy-id` qui le fait pour vous :

```
[pbismuth@mymachine] ssh-copy-id -p 6666 -i ~/.ssh/id_rsa.pub mylogin@info-ssh1.iut.u-bordeaux.fr
```



Notez que, s'il s'agit de votre première connexion sur la passerelle, SSH va vous demander de confirmer qu'il s'agit bien d'une machine de confiance :

```
The authenticity of host 'info-ssh1.iut.u-bordeaux.fr (147.210.94.204)' can't be established.
ECDSA key fingerprint is SHA256:QOCmFO5edpqFNtQisDS/Wd05WE8nblOMPo+2sD1FkxM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'info-ssh1.iut.u-bordeaux.fr,147.210.94.204'
(ECDSA) to the list of known hosts.
```

Répondez `yes` et, normalement, votre clé publique est ajoutée au fichier `authorized_keys`. Désormais, vous pouvez vous connecter sans taper de mot de passe sur la machine passerelle `info-ssh1` :

```
[pbismuth@mymachine] ssh -p 6666 mylogin@info-ssh1.iut.u-bordeaux.fr ls
Bureau     iut-bibliotheque  iut-zrr  Téléchargements  win-downloads
Desktop    iut-remise        Modèles  Vidéos           win-home
Documents  iut-vms           Musique  win-desktop      win-pictures
Images     iut-www           Public   win-documents    workspace
[pbismuth@mymachine]
```

4 Configurer SSH pour se simplifier la vie

Il est possible de créer un fichier de configuration `${HOME}/.ssh/config` dans lequel on pourra définir des "alias" (i.e. pseudonymes pour les machines), son identifiant de connexion par serveur, ou encore les options d'affichage déporté.

Typiquement, la chaine `mylogin@info-ssh1.iut.u-bordeaux.fr` est assez longue à taper. Le préfixe `mylogin@` (qui indique votre identifiant de connexion UB) peut être omis si cet identifiant est le même que sur votre machine locale. Dans le cas général, ce n'est pas le cas.

Pour davantage de commodité, il est possible de créer le fichier `~/.ssh/config` suivant :

```
host iut1
  user pbismuth
  hostname info-ssh1.iut.u-bordeaux.fr
  port 6666
  forwardx11 yes
  forwardagent yes

host iut2
  user pbismuth
  hostname info-ssh2.iut.u-bordeaux.fr
  port 6666
  forwardx11 yes
  forwardagent yes
```

Il est désormais possible de vous connecter au département informatique de l'IUT de cette façon :

```
[pbismuth@mymachine] ssh iut1
mylogin@info-ssh1:~$
```

Ou d'exécuter une commande distante :

```
[pbismuth@mymachine] ssh iut1 pwd
/mnt/roost/users//mylogin
[pbismuth@mymachine]
```

Copier un fichier à distance est tout aussi facile :

```
[pbismuth@mymachine] scp mon_fichier_local iut1:sous_repertoire-distant/
```

Ou, dans l'autre sens :

```
[pbismuth@mymachine] scp iut1:sous_repertoire_distant/mon_fichier_distant .
```

5 Travailler à distance

Les machines `info-ssh1` et `info-ssh2` sont les passerelles qui accueillent toutes les connexions entrantes. Pour cette raison, elles ne doivent pas être surchargées et elles ne peuvent donc pas être utilisées pour des compilations ou pour lancer des applications.

Il faut donc utiliser une passerelle à titre *transitoire* pour se connecter sur une machine banalisée (par exemple `info-basson`).

Supposons que vous vouliez travailler sur la machine `info-basson`, vous devez donc procéder ainsi :

```
[pbismuth@mymachine] ssh iut1
mylogin@info-ssh1:~$ ssh info-basson
mylogin@info-basson:~$
```

Note

En temps normal, vous seriez invité à entrer votre mot de passe lors du second `ssh` (et non pas une *pass phrase*) car il n'y a pas de couple de clé privée/publique sur votre compte IUT… Toutefois, ici on exploite une fonctionnalité très pratique de SSH qui permet de « faire suivre » les requêtes d'authentification sur `info-ssh1` à l'agent qui s'exécute sur votre machine locale : c'est l'objet de la ligne `ForwardAgent` dans le fichier `~/.ssh/config` établi en section 4.

6 Rebondir automatiquement

Pour simplifier encore la connexion avec la machine choisie à l'IUT, il est possible d'automatiser le « rebond SSH » effectué précédemment. Voici le nouveau fichier `~/.ssh/config` permettant de se connecter sur `info-basson` sans chichi :

```
host iut1
  user pbismuth
  port 6666
  hostname info-ssh1.iut.u-bordeaux.fr
  forwardx11 yes
  forwardagent yes
```

```
host iut2
  user pbismuth
  port 6666
  hostname info-ssh2.iut.u-bordeaux.fr
  forwardx11 yes
  forwardagent yes

host basson
Hostname info-basson.iut.bx1
User mylogin
ProxyCommand ssh iut1 -W %h:%p
ForwardX11 yes
```

Désormais, vous pouvez vous connecter sur `info-basson` comme ceci :

```
[pbismuth@mymachine] ssh basson
mylogin@info-basson:~$
```

Mieux encore, les clients SSH proposent une option directement en ligne de commande pour faire un saut intermédiaire sur une machine :

```
[pbismuth@mymachine] ssh -J iut1 info-basson
mylogin@info-basson:~$
```

7 Afficher des fenêtres graphiques à distance

Lorsque les applications ne sont pas gourmandes en bande passante, vous pouvez déporter l'affichage d'application s'exécutant à distance sur votre ordinateur local. Pour cela, vous devez disposer d'un serveur X11 (standard sous Linux ou Mac, installable facilement sous Windows).

Avant de lancer une application graphique, il faut s'être connecté via `ssh -Y` pour spécifier que vous souhaitez rediriger les requêtes d'affichage graphique vers votre poste. Ou bien, comme nous l'avons fait dans le fichier de configuration précédent, utiliser l'option `ForwardX11`.

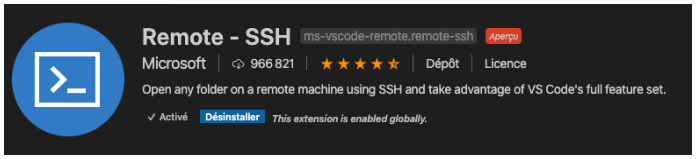
Par exemple, il est possible d'exécuter la commande Unix `xterm` sur la machine `info-basson` avec un affichage sur votre machine locale :

```
[pbismuth@mymachine] ssh basson xterm
```

8 Éditer les fichiers à distance

Bien que vous puissiez lancer sur la machine distante un éditeur de texte s'affichant chez vous en suivant la méthode exposée en section 7, c'est fortement déconseillé en raison de la bande passante requise par le protocole X11.

Il est préférable de lancer un éditeur de texte *localement* qui soit capable d'éditer les fichiers distants de manière transparente. C'est le cas de Visual Studio Code via l'extension « `remote- SSH` » :



Dans ce dernier cas, il suffit de cliquer dans la zone verte en bas à gauche de la fenêtre : vous serez invité à entrer le nom de la machine distante puis vous pourrez vous y connecter en toute transparence, y lancer des compilations, etc.

Vous pouvez également faire un montage à distance de votre `${HOME}` IUT avec la commande `sshfs` :

```
[pbismuth@mymachine] sshfs mylogin@info-ssh2.iut.u-bordeaux.fr:/mnt/roost/users/mylogin \
                          /home/pbismuth/mnt/iut -p 6666 -o follow_symlinks
```

9 Rediriger le port d'un serveur

Vous savez maintenant vous connecter en SSH sur un serveur du département. Maintenant, on peut avoir accès à un port spécifique d'un autre serveur, pour l'utilisation d'un client par exemple.

Note1 : dans toutes les procédures ci-dessous, il faudra bien sûr que le port que vous souhaitez utiliser en local ne soit pas déjà utilisé. Par exemple, si le serveur mySQL est déjà installé sur votre machine, vous devrez utiliser un autre port que le 3306 pour rediriger le port mySQL de l'autre serveur (3307 par exemple).

Note2 : les serveurs sont redémarrés la nuit (à 5h du matin), ne laissez pas de travail non enregistré en suspens le soir.

9.1 Pour les possesseurs de machine sous Linux ou Mac

Par exemple, pour rediriger en local (i.e. sur votre ordinateur) le port 3306 (MySQL) du serveur info-basson (172.16.1.17) en passant par info-ssh2 :

```
[pbismuth@mymachine] ssh -p 6666 -fNL 3306:172.16.1.17:3306 mylogin@info-ssh2.iut.u-bordeaux.fr
```

=> le port 3306 du serveur MySQL hébergé sur info-basson est désormais mappé sur votre machine.

Exemple d'accès :

```
[pbismuth@mymachine] mysql -u jdupont -p -h 127.0.0.1 mabase
```

Ou, si vous avez utilisé le port 3307 pour la redirection :

```
[pbismuth@mymachine] mysql -u jdupont -p -h 127.0.0.1 -P 3307 mabase
```

9.2 Pour les utilisateurs de Windows

Vérifiez la présence de putty sur votre machine : cliquez sur l'icône Windows en bas à gauche puis tapez directement au clavier p puis u puis t... si Windows n'affiche pas "PuTTy" dans les meilleurs résultats, c'est qu'il est absent. Dans ce cas, téléchargez la version 'standalone' de putty : https://the.earth.li/~sgtatham/putty/latest/w64/putty.exe

Lancez l'exécutable. Dans "Host Name" (voir figure 1), saisissez l'un ou l'autre des serveurs SSH (voir plus haut), un nom quelconque dans "Saved Sessions" (ici, SSH-DEPINFO), puis cliquez sur "Save". Le nom s'ajoute alors dans la liste des sessions enregistrées :

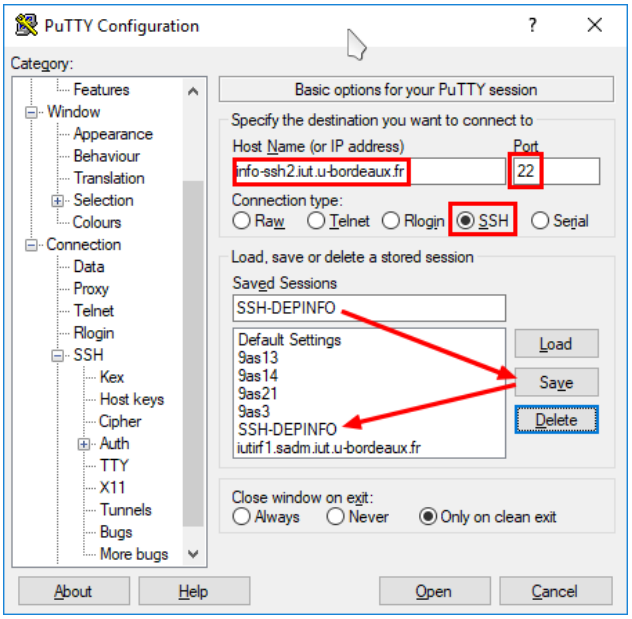


FIGURE 1 – Host Name

Note

Pensez à remplacer le port 22 par 6666!!!

Dans le menu de gauche, allez ensuite dans "Connection/SSH/Tunnels" (voir figure 2), saisissez le port source (local), le serveur ciblé avec son port d'écoute (format adresse-ip :port) puis cliquez sur "Add", un nouvel item apparaît alors dans la liste des ports redirigés :

Revenez sur "Sessions" pour enregistrer ces modifications ("Save") puis cliquez sur "Open" pour lancer la connexion. Vous devrez alors vous identifier classiquement.

Ne fermez pas la session ouverte, sinon, la redirection du port s'interrompra avec! Vous pouvez néanmoins réduire la fenêtre.

Avec votre client mySQL, vous pouvez alors vous connecter à votre base en choisissant localhost ou 127.0.0.1 comme "serveur". N'oubliez pas de changer le port le cas échéant.

Listes de ports usuels : http://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-sg-fr-4/ch-ports.html

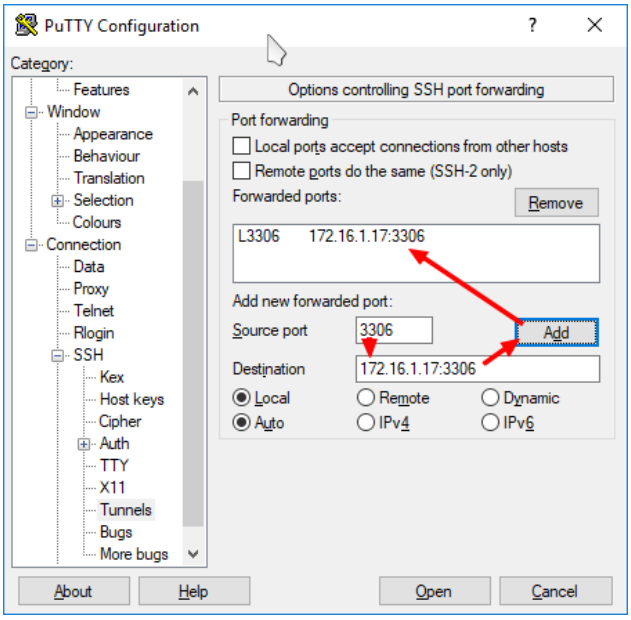


FIGURE 2 – Tunnels