

Transformer sa machine en serveur

Cette étape vous concerne si vous voulez transformer votre PC en serveur. Par exemple, si vous voulez accéder à votre PC depuis un autre endroit (et donc suivre le reste de ce chapitre), vous devez le transformer en serveur au préalable.

Il faut tout simplement installer le paquet openssh-server :

Code : Console

```
Sudo apt-get install openssh-server
```

Lors de l'installation, vous devriez voir certaines étapes intéressantes s'effectuer automatiquement :

Code : Console

```
Creating SSH2 RSA key; this may take some time ...  
Creating SSH2 DSA key; this may take some time ...  
* Restarting OpenBSD Secure Shell server sshd [ OK ]
```

RSA et DSA sont 2 algorithmes de cryptage asymétrique. Comme je vous l'ai dit plus tôt, SSH peut travailler avec plusieurs algorithmes de cryptage différents. Ce que vous voyez là est l'étape de la création d'une paire de clés publiques et privées pour chacun des 2 algorithmes (RSA et DSA).

Ensuite, le programme de serveur SSH (appelé sshd) est lancé.

Normalement, le serveur SSH sera lancé à chaque démarrage. Si ce n'est pas le cas, vous pouvez le lancer à tout moment avec la commande suivante :

Code : Console

```
Sudo /etc/init.d/ssh start
```

Et vous pouvez l'arrêter avec cette commande :

Code : Console

```
Sudo /etc/init.d/ssh stop
```

Normalement vous ne devriez pas avoir besoin de configurer quoi que ce soit, mais sachez au besoin que le fichier de configuration se trouve dans /etc/ssh/ssh_config. Il faudra recharger SSH avec la commande `sudo /etc/init.d/ssh reload` pour que les changements soient pris en compte.

Voilà, votre machine est désormais un serveur SSH ! Vous pouvez vous y connecter depuis

n'importe quelle machine Linux ou Windows dans le monde.

Nous commencerons dans un premier temps par voir comment accéder à votre PC à distance depuis une machine Linux.

Se connecter via SSH à partir d'une machine Linux

Toutes les machines équipées de Linux proposent la commande ssh qui permet de se connecter à distance à une autre machine.

A partir d'ici, je suppose que vous avez installé openssh-server et que votre machine est allumée. L'idéal serait d'aller chez un ami qui a Linux (ou d'utiliser un autre PC de chez vous équipé de Linux).

Ouvrez une console sur le PC de votre ami, et utilisez la commande ssh comme ceci :

```
Code : Console  
Ssh login@ip
```

Il faut remplacer "login" par votre login (mateo21 dans mon cas) et "ip" par l'adresse IP de votre ordinateur.

Si vous vous connectez depuis chez un ami, il vous faut entrer l'IP internet de votre PC que vous pouvez obtenir en allant sur <http://www.whatismyip.com> par exemple.

Si vous vous connectez depuis un autre PC chez vous (sur le même réseau local), il vous faut entrer l'IP locale que vous devriez voir en tapant la commande ifconfig (par exemple 192.168.0.3).

Si vraiment vous n'avez ni ami sous Linux ni second PC dans la maison, vous pouvez simuler une connexion réseau en vous connectant de votre PC vers votre PC. Utilisez pour cela l'IP 127.0.0.1 (ou le mot localhost), ça marche toujours.

Si je suis chez un ami et que l'IP internet de mon ordinateur est 87.112.13.165, je vais taper :

```
Code : Console  
Ssh mateo21@87.112.13.165
```

Si, faute de mieux, vous voulez tester en vous connectant chez vous depuis chez vous, vous pouvez taper :

```
Code : Console  
Ssh mateo21@localhost
```

Cette seconde méthode marche toujours, mais c'est moins impressionnant parce que vous ne faites

que simuler une connexion réseau. 😊

Normalement, le serveur devrait répondre au bout d'un moment et vous devriez voir quelque chose comme :

Code : Console

```
The authenticity of host 'localhost (127.0.0.1)' can't be established.  
RSA key fingerprint is 49:d9:2d:2a:df:fd:80:ab:e9:eb:59:37:58:34:de:f7.  
Are you sure you want to continue connecting (yes/no)?
```

Si vous n'avez pas de réponse du serveur, vérifiez que vous ne vous êtes pas trompé d'IP. Vérifiez aussi que le port 22 n'est pas bloqué par un firewall, car c'est celui utilisé par SSH par défaut. Si le serveur tourne sur un autre port, il faudra préciser le numéro de ce port comme ceci : `ssh philox47 -p 12451` (si le serveur fonctionne sur le port 12451 au lieu du port 22).

Que se passe-t-il ? On vous dit que le fingerprint (empreinte) du serveur est :

49:d9:2d:2a:df:fd:80:ab:e9:eb:59:37:58:34:de:f7. C'est un numéro unique qui vous permet d'identifier le serveur. Si demain quelqu'un essaie de se faire passer pour le serveur, le fingerprint changera forcément et vous saurez qu'il se passe quelque chose d'anormal. Ne vous inquiétez pas, SSH vous avertira de manière très claire si cela arrive.

En attendant, tapez "yes" pour confirmer que c'est bien le serveur auquel vous voulez vous connecter. Le serveur et le client vont alors s'échanger une clé de cryptage comme je vous l'ai expliqué un peu plus tôt. Normalement, au bout de quelques secondes le serveur devrait vous demander votre mot de passe :

Code : Console

```
mateo21@localhost's password:
```

Vous pouvez entrer votre mot de passe en toute sécurité, la communication est cryptée. 😊

Si vous rentrez le bon mot de passe, la console du PC de votre ami (ou votre propre console) devrait vous afficher un message de bienvenue puis un prompt qui correspond à la console de votre PC. Bravo, vous êtes connecté !

Code : Console

```
mateo21@mateo21-desktop:~$
```

Si on ne vous affiche pas d'erreur, c'est que vous êtes bien loggé et que vous travaillez désormais sur votre machine à distance ! Vous pouvez effectuer toutes les opérations que vous voulez comme si vous étiez chez vous.

Essayez de parcourir les dossiers pour voir que ce sont bien les vôtres, et amusez-vous (pourquoi pas) à créer un fichier (avec nano). Lorsque vous reviendrez sur votre PC vous l'y retrouverez. 😊

Vous pouvez aussi commander l'exécution d'un programme, d'une recherche, etc. Vous savez déjà comment lancer un programme en tâche de fond pour qu'il continue même quand vous n'êtes pas connecté à la machine (vous vous souvenez de nohup et de screen ?).

Pour vous déconnecter, tapez "logout" ou son équivalent : la combinaison de touches Ctrl + D.

Se connecter via SSH à partir d'une machine Windows

Si vous voulez avoir accès à la console de votre machine Linux mais que vous n'avez pas d'autre machine Linux sous la main, pas de panique ! Il existe des programmes pour Windows faits pour ça. Le plus connu d'entre eux, et celui que j'utilise personnellement, s'appelle **PuTTY**.

Vous pouvez [télécharger PuTTY sur son site officiel](#).

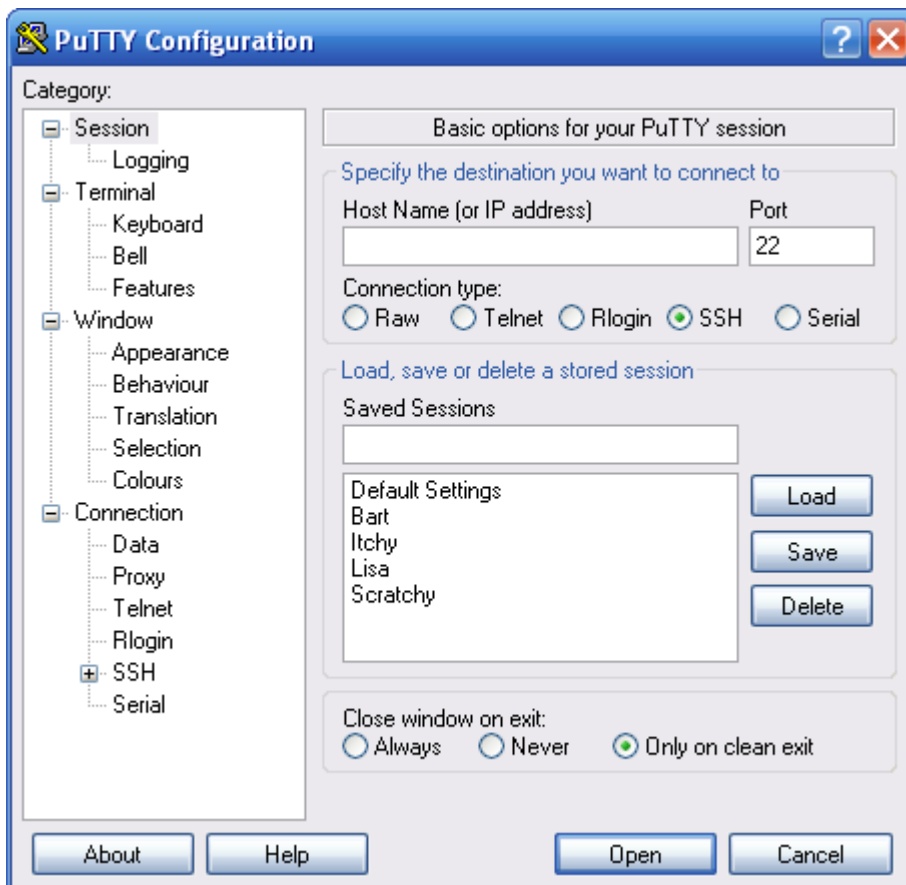
Je sais, vous devez vous dire que ce n'est pas très clair et que vous ne voulez pas chercher sur quel lien cliquer sur cette page. 😊

Repérez la section "Binaries". C'est un tableau. Vous avez le choix entre :

- Cliquer sur "putty.exe" pour télécharger le programme principal. Il ne nécessite pas d'installation.
- Cliquer sur le programme d'installation (par exemple "putty-0.60-installer.exe"). Celui-ci installera PuTTY et d'autres utilitaires dont vous aurez besoin dans quelques minutes.

Putty.exe suffit, mais je vous recommande donc de prendre le package complet en récupérant le programme d'installation.

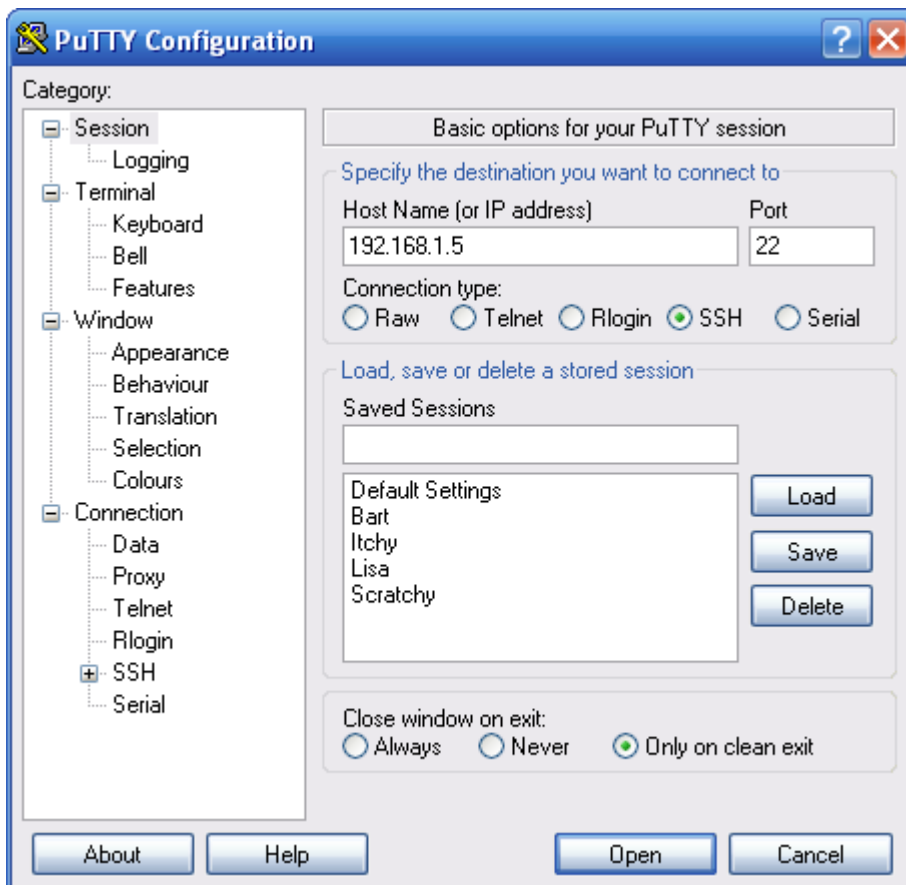
Une fois que c'est fait et installé, lancez Putty. Une fenêtre comme celle-ci devrait s'afficher :



Il y a beaucoup de pages d'options, comme vous pouvez le voir au niveau de la section "Category" sur le côté. Pour le moment, pas de panique, vous avez juste besoin de remplir le champ en haut "Host Name (or IP address)". Entrez-y l'adresse IP de votre ordinateur sous Linux.

J'ai donné quelques explications à propos de l'adresse IP un peu plus haut lorsque j'ai parlé de la connexion SSH depuis Linux. Lisez donc les paragraphes précédents si vous voulez plus d'informations à ce sujet.

Dans mon cas, je vais rentrer l'adresse IP de mon PC sous Linux situé sur mon réseau local (192.168.1.5) :



Vous pouvez changer le numéro du port si ce n'est pas 22, mais normalement c'est 22 par défaut.

Ensuite, vous n'avez plus qu'à cliquer sur le bouton tout en bas "Open" pour lancer la connexion. Rien d'autre !

Si vous voulez sauvegarder l'IP et les paramètres pour ne pas retaper ça à chaque fois, donnez un nom à cette connexion (par exemple le nom de votre ordinateur) dans le champ sous "Saved Sessions", puis appuyez sur le bouton Save. La prochaine fois, vous n'aurez qu'à double-cliquer sur le nom de votre PC dans la liste pour vous y connecter directement.

La première fois que vous vous connectez à votre serveur, PuTTY devrait vous demander une confirmation comme ceci :



C'est la même chose que sous Linux : on vous donne l'empreinte (fingerprint) de votre serveur. Vous devez confirmer que c'est bien chez lui que vous voulez vous connecter. Cliquez sur Oui pour confirmer.

A l'avenir, on ne vous reposera plus la question. Par contre, si le fingerprint change, un gros message d'avertissement s'affichera. Cela signifiera soit que le serveur a été réinstallé, soit que quelqu'un est en train de se faire passer pour le serveur (c'est ce qu'on appelle une attaque man-in-the-middle). Cela ne devrait fort heureusement pas vous arriver, du moins on l'espère. 😊

Le serveur vous demande alors le login et le mot de passe :

A screenshot of a PuTTY terminal window titled "192.168.1.5 - PuTTY". The terminal shows a login prompt "login as: mateo21" and the user has entered "mateo21". Below that, it says "mateo21@192.168.1.5's password:" followed by a green cursor. The terminal background is black, and the text is white. The window has standard Windows-style window controls (minimize, maximize, close) in the top right corner.

Rappelez-vous qu'il est normal que les caractères ne s'affichent pas quand vous tapez votre mot de passe. Il n'y a même pas d'étoiles pour des raisons de sécurité, afin que quelqu'un ne soit pas tenté de compter le nombre de caractères en regardant derrière votre épaule !

Si tout est bon, vous devriez être connecté à votre machine !

```
mateo21@mateo21-laptop: ~
login as: mateo21
mateo21@192.168.1.5's password:
Linux mateo21-laptop 2.6.27-7-generic #1 SMP Tue Nov 4 19:33:20 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
Last login: Mon Nov 17 14:50:35 2008 from localhost
mateo21@mateo21-desktop:~$ █
```

Et voilà, vous êtes chez vous ! Vous pouvez faire ce qui vous chante : lire vos fichiers, écrire des fichiers, lancer une recherche, exécuter un programme, bref vous êtes chez vous. 😊

Pour vous déconnecter, tapez "logout" ou son équivalent : la combinaison de touches Ctrl + D.

L'identification automatique par clé

Il y a plusieurs façons de s'authentifier sur le serveur, pour qu'il sache que c'est bien vous. Les 2 plus utilisées sont :

- Authentification par mot de passe.
- Authentification par clé publique et privée du client.

Nous avons pour le moment vu uniquement l'authentification par mot de passe (le serveur vous demandait votre mot de passe).

Il est possible d'éviter que l'on vous demande votre mot de passe à chaque fois grâce à une authentification spéciale par clés. Cette méthode d'authentification est plus complexe à mettre en place, mais elle est ensuite plus pratique.

Avec cette nouvelle méthode d'authentification, c'est le client qui va générer une clé publique et une clé privée. Les rôles sont un peu inversés.

L'avantage est qu'on ne vous demandera pas votre mot de passe à chaque fois pour vous connecter. Si vous vous connectez très régulièrement à un serveur, c'est vraiment utile. Si vous faites bien les choses, cette méthode est tout aussi sûre que l'authentification par mot de passe.


```
|00+. 00. .0 . |  
+-----+
```

Dans un premier temps, le client génère une paire de clés ("Generating public/private rsa key pair"). Il doit ensuite sauvegarder ces clés dans des fichiers (un pour la clé publique, un pour la clé privée). On vous propose une valeur par défaut : je vous conseille de ne rien changer et de taper simplement Entrée.

Ensuite, on vous demande une passphrase. C'est une phrase de passe qui va servir à crypter la clé privée pour une meilleure sécurité. Là vous avez 2 choix :

- Soit vous tapez Entrée directement sans rien écrire, et la clé ne sera pas cryptée sur votre machine.
- Soit vous tapez un mot de passe de votre choix, et la clé sera cryptée.

Tout le monde ne met pas une phrase de passe. En fait ça dépend le risque que vous avez que quelqu'un d'autre utilise la machine du client et puisse lire le fichier contenant la très secrète clé privée. Si le PC du client est votre PC chez vous et que personne d'autre ne l'utilise, il y a assez peu de risque (à moins d'avoir un virus, un spyware...). Si c'est en revanche un PC public, je vous recommande vivement de mettre une passphrase pour chiffrer la clé qui sera enregistrée.

Si vous hésitez entre les 2 méthodes, je vous recommande de rentrer une passphrase : c'est quand même la méthode la plus sûre.

Envoyer la clé publique au serveur

Il faut maintenant envoyer au serveur votre clé publique pour qu'il puisse vous crypter des messages.

Votre clé **publique** devrait se trouver dans `~/.ssh/id_rsa.pub` (pub comme public). `~` correspond à votre home (`/home/mateo21/` dans mon cas). Notez que `.ssh` est un dossier caché.

Votre clé **privée**, elle, se trouve dans `~/.ssh/id_rsa`. Ne la communiquez à personne ! Elle est normalement cryptée si vous avez entré une passphrase, ce qui constitue une sécurité de plus.

Vous pouvez vous rendre dans le dossier `.ssh` déjà pour commencer :

```
Code : Console  
Cd ~/.ssh
```

Si vous faites un `ls` vous devriez voir ceci :

```
Code : Console  
$ ls
```

```
id_rsa id_rsa.pub known_hosts
```

Les 3 fichiers sont :

- `id_rsa` : votre clé privée, qui doit rester secrète. Elle est cryptée si vous avez rentré une passphrase.
- `id_rsa.pub` : la clé publique que vous pouvez communiquer à qui vous voulez, et que vous devez envoyer au serveur.
- `known_hosts` : c'est la liste des fingerprint que votre PC de client tient à jour. Ca lui permet de se souvenir de l'identité des serveurs et de vous avertir si, un jour, votre serveur est remplacé par un autre (qui pourrait être celui du pirate !). Je vous en ai déjà parlé un peu plus tôt.

L'opération consiste à envoyer la clé publique (`id_rsa.pub`) au serveur et à l'ajouter à son fichier "`authorized_keys`" (clés autorisées). Le serveur y garde une liste des clés qu'il autorise à se connecter.

Le plus simple pour cela est d'utiliser la commande spéciale `ssh-copy-id`. Utilisez-la comme ceci :

Code : Console

```
Ssh-copy-id -i id_rsa.pub login@ip
```

Remplacez-y votre login et l'ip de votre serveur.

Code : Console

```
$ ssh-copy-id -i id_rsa.pub mateo21@88.92.107.7
mateo21@88.92.107.7's password:
Now try logging into the machine, with "ssh 'mateo21@localhost'", and check in:

 .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.
```

Si vous devez vous connecter au serveur par un autre port que celui par défaut, basez-vous sur la commande suivante : `ssh-copy-id -i id_rsa.pub "-p 14521 mateo21@88.92.107.7"`.

On vous demande votre mot de passe (celui de votre compte, pas la passphrase). En fait vous vous connectez par mot de passe encore une fois, pour pouvoir ajouter sur le serveur votre clé publique.

La clé est ensuite automatiquement ajoutée à `~/.ssh/authorized_keys` sur le serveur. On vous invite à vérifier si l'opération s'est bien déroulée en ouvrant le fichier `authorized_keys`, ce que vous pourrez faire plus tard si vous le voulez.

Se connecter !

Maintenant, connectez-vous au serveur comme vous le faisiez auparavant :

```
Code : Console
Ssh login@ip
```

Par exemple :

```
Code : Console
$ ssh mateo21@88.92.107.7
Enter
passphrase for key '/home/mateo21/.ssh/id_rsa':
```

On vous demande la phrase de passe pour décrypter votre clé privée. Rentrez-la.

Normalement, si tout va bien, vous devriez être alors connecté au serveur.

Où je suis le dernier des nuls, ou alors c'est ce système qui est nul. Auparavant on me demandait mon mot de passe. Maintenant on me demande une phrase de passe pour décrypter la clé privée. Où est le progrès ???

Je comprends votre frustration. 😊

En fait, si vous n'aviez pas mis de phrase de passe, on ne vous aurait rien demandé et vous auriez été directement connecté. Heureusement, il y a une solution pour ceux qui ont choisi la sécurité en utilisant une phrase de passe, mais qui ne veulent quand même pas avoir à la rentrer à chaque fois : l'agent SSH.

L'agent SSH

L'agent SSH est un programme qui tourne en arrière-plan en mémoire. Il retient les clés privées pendant toute la durée de votre session.

Tout ce que vous avez à faire est de lancer le programme ssh-add sur le PC du client :

```
Code : Console
$ ssh-add
Enter passphrase for /home/mateo21/.ssh/id_rsa:
Identity added: /home/mateo21/.ssh/id_rsa (/home/mateo21/.ssh/id_rsa)
```

Il va automatiquement chercher votre clé privée. Pour la décrypter, il vous demande la passphrase. Rentrez-la.

Maintenant que c'est fait, à chaque fois que vous vous connecterez à un serveur, vous n'aurez plus besoin de rentrer la passphrase. Essayez de vous connecter à votre serveur pour voir ! 😊

L'intérêt de l'agent SSH est qu'il ne vous demande la passphrase qu'une seule fois au début. Ensuite, vous pouvez vous connecter plusieurs fois au même serveur, ou même à plusieurs serveurs différents, le tout sans avoir besoin de retaper votre passphrase !

Authentification par clé depuis Windows (PuTTY)

Il est tout à fait possible d'utiliser l'authentification par clé avec Putty. C'est là justement qu'il est recommandé d'avoir pris l'installateur, et non pas juste le programme principal putty.exe.

Le principe est le même que sous Linux : il faut d'abord qu'on génère une paire de clés sur le PC du client, puis qu'on les envoie au serveur. Nous retrouverons aussi un équivalent de l'agent SSH pour éviter d'avoir à rentrer une passphrase à chaque fois.

Commençons par la génération des clés.

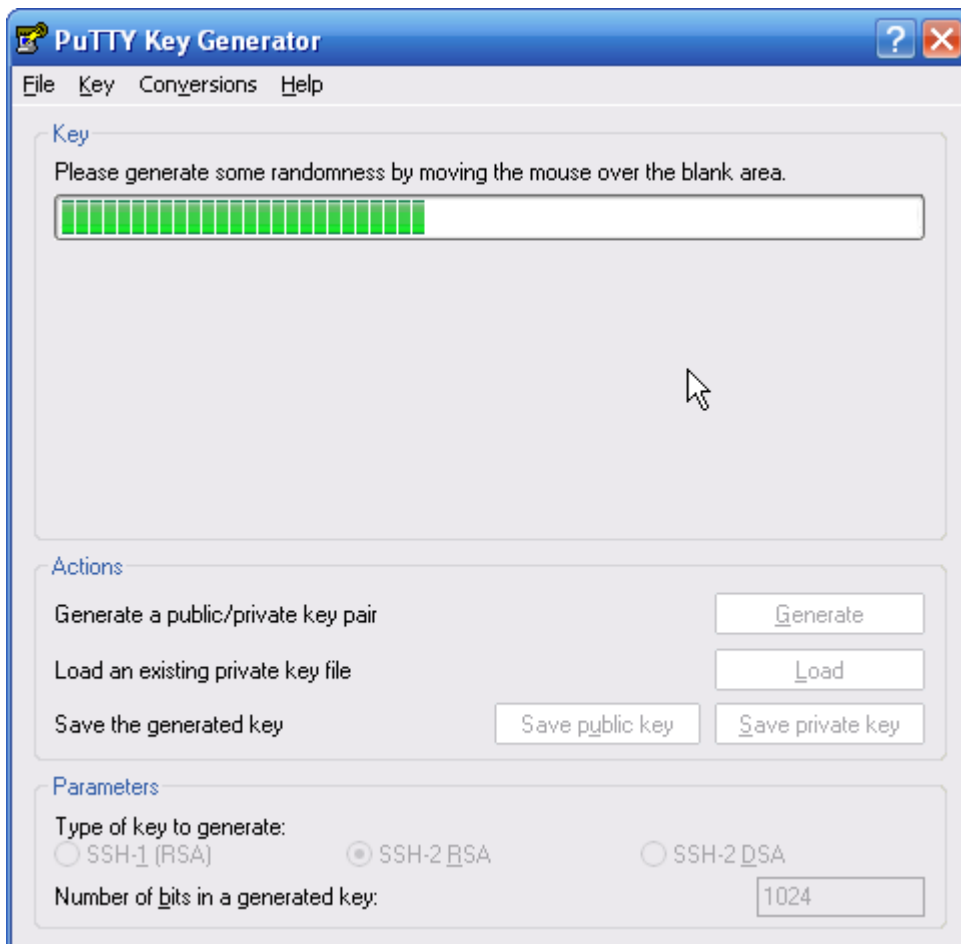
Générer une paire de clés (publique et privée) avec Puttygen

Normalement, vous devriez avoir installé un programme appelé Puttygen (il se trouvait dans l'installateur de Putty). Lancez-le :



En bas de la fenêtre vous pouvez choisir vos paramètres : algorithme de cryptage et puissance du cryptage. Les valeurs par défaut ici (RSA 1024 bits) sont tout à fait convenables. Vous pouvez les changer, mais sachez qu'elles sont sûres et que vous pouvez donc vous en contenter.

Cliquez sur le bouton "Generate". Le programme va générer une paire de clés (publique et privée). Pour aider le programme à générer cette paire, il vous propose quelque chose d'assez amusant : vous devez bouger la souris dans la fenêtre. Comme vous allez le faire aléatoirement, cela aidera Puttygen à générer des clés aléatoires. Sous Linux, on utilise d'autres méthodes pour générer des clés aléatoirement (il faut dire qu'en console on n'a pas de souris 🐭).



Génération des clés grâce aux mouvements de la souris

Une fois que c'est fait, on vous affiche la clé publique :



(Comme vous le voyez, ça ne me dérange pas que tout le monde voie ma clé publique. Le principe c'est justement que tout le monde peut voir cette clé, mais on ne peut rien en faire. Par contre la clé privée doit rester secrète.)

Vous pouvez choisir d'entrer une passphrase ou non. Comme je vous l'ai expliqué plus tôt, cela renforce la sécurité en cryptant la clé privée. Saisissez la passphrase dans les champs "Key passphrase" et "Confirm passphrase".

Ensuite, enregistrez la clé publique dans un fichier en cliquant sur "Save public key". Vous pouvez nommer ce fichier comme vous voulez, par exemple cle.pub. Enregistrez-le où vous voulez. Puis, enregistrez la clé privée en cliquant sur "Save private key". Donnez-lui l'extension .ppk : cle.ppk par exemple.

Ne fermez pas encore Puttygen.

Envoyer la clé publique au serveur

Comme sous Linux tout à l'heure, il faut envoyer la clé publique au serveur pour qu'il nous autorise à nous connecter par clé. Le problème, c'est qu'il n'y a pas de commande pour le faire automatiquement depuis Windows. Il va falloir ajouter la clé à la main dans le fichier `authorized_keys`. Heureusement ce n'est pas très compliqué.

Ouvrez Putty et connectez-vous au serveur comme auparavant (en rentrant votre mot de passe habituel). Rendez-vous dans ~/.ssh :

```
Code : Console
Cd ~/.ssh
```

Si le dossier .ssh n'existe pas, pas de panique, créez-le : mkdir .ssh

Rajoutez votre clé publique à la fin du fichier authorized_keys (s'il n'existe pas il sera créé). Vous pouvez utiliser la commande suivante :

```
Code : Console
Echo "votre_cle" >> authorized_keys
```

Rappel : votre clé publique est affichée dans Puttygen, que vous ne devriez pas avoir fermé. Pour coller la clé dans la console, utilisez la combinaison de touches Shift + Inser plutôt que Ctrl + V.

Par exemple :

```
Code : Console
Echo "ssh-rsa AAAAB3NzaC1yc2E [...] AAAABJQAP++UWB0kLp0= rsa-key-20081117" >> authorized
```

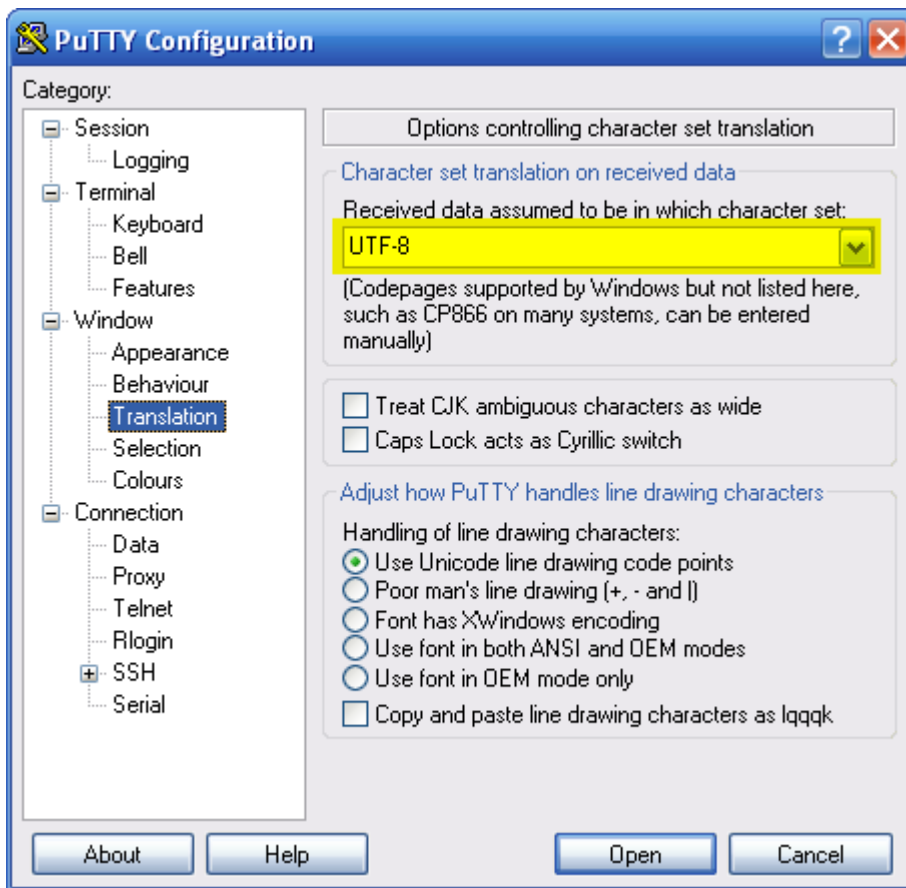
Voilà, c'est fait. 😊

Déloggez-vous, et relancez Putty. On va maintenant le configurer pour qu'il se connecte à l'aide de la clé.

Configurer Putty pour qu'il se connecte avec la clé

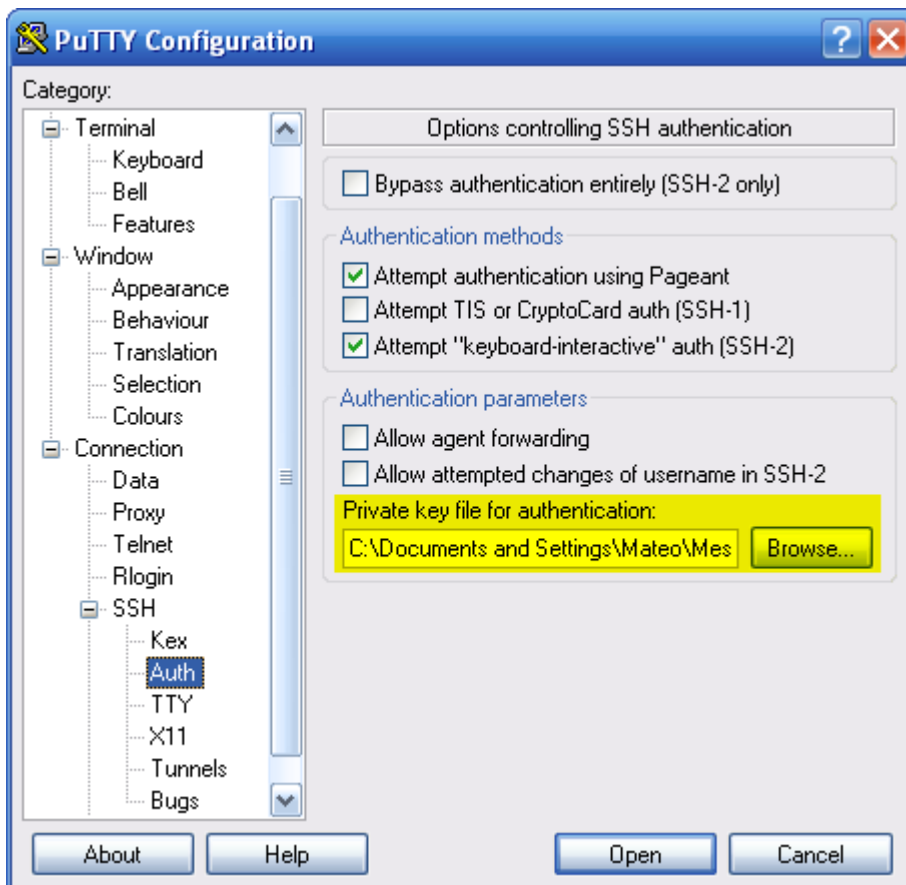
Une fois Putty ouvert, rendez-vous dans la section "Window > Translation" pour commencer. Ça n'a pas de rapport direct avec les clés, mais cela vous permettra de régler le problème des accents qui s'affichent mal dans la console si vous l'avez rencontré.

Réglez la valeur de la liste déroulante à UTF-8 :

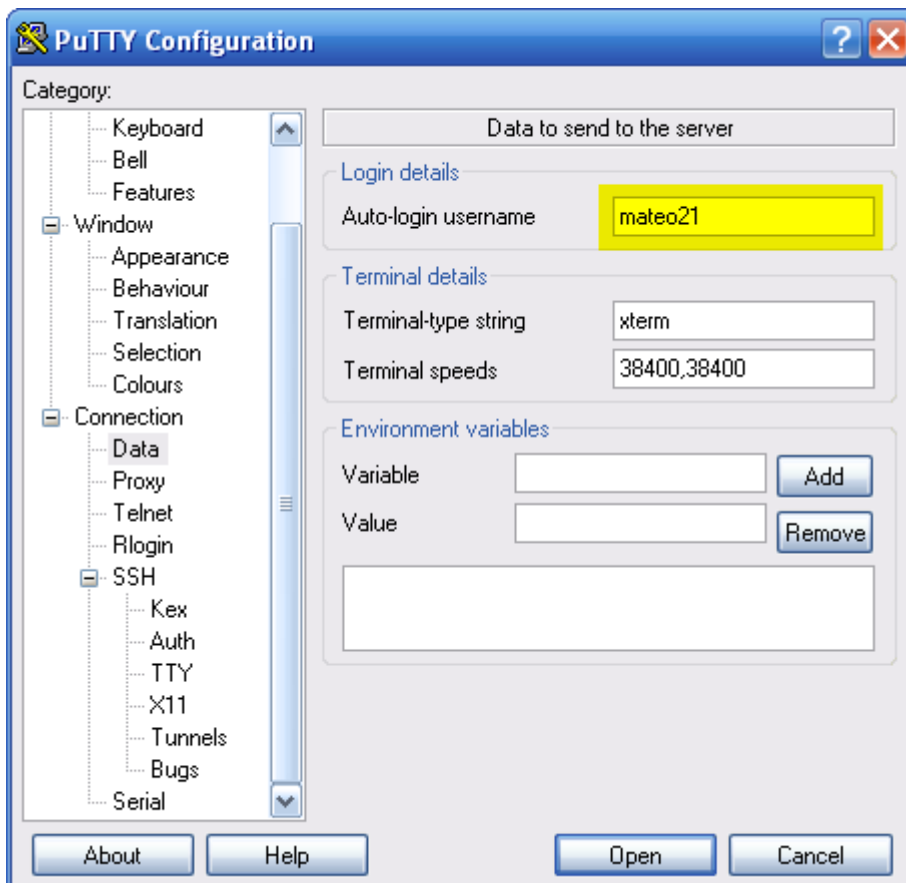


La plupart des serveurs encodent désormais les caractères en UTF-8, cela devrait donc vous éviter des soucis d'affichage.

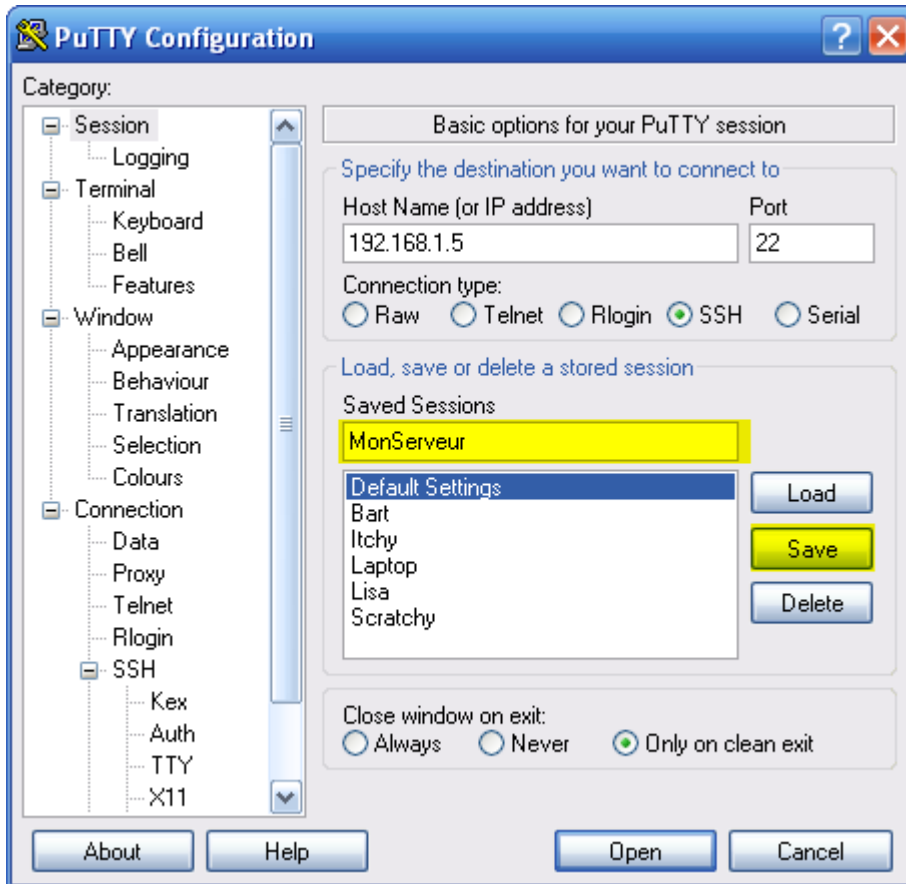
Maintenant, rendez-vous dans "Connection > SSH > Auth". Cliquez sur le petit bouton "Browse" pour sélectionner votre clé privée :



Je vous recommande aussi d'aller dans "Connection > Data" et de rentrer votre login dans "Auto-login username" :



Retournez à l'accueil en cliquant sur la section "Session" tout en haut. Rentrez l'ip du serveur. Ensuite, je vous recommande fortement d'enregistrer ces paramètres.

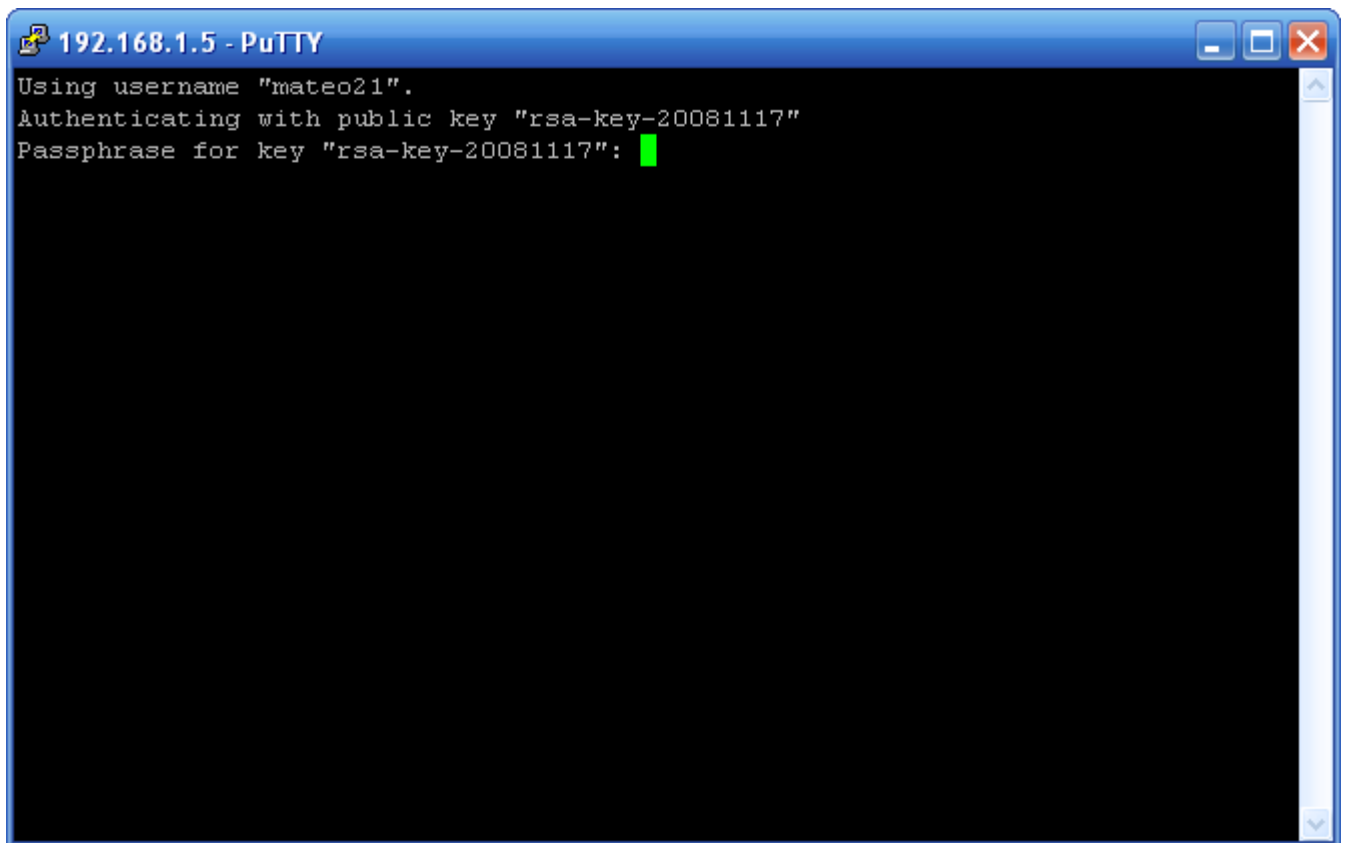


Entrez un nom à votre serveur (par exemple MonServeur) sous "Saved Sessions". Cliquez ensuite sur Save.

A l'avenir, vous n'aurez qu'à double-cliquer sur le nom de votre serveur dans la liste pour vous y connecter directement avec les bons paramètres.

Cliquez sur "Open" pour vous connecter au serveur.

Vous devriez voir Putty utiliser automatiquement votre pseudo, puis vous demander votre passphrase. Rentrez-la pour vérifier que ça marche :



Euh, et si je veux pas avoir à rentrer la passphrase à chaque fois ? Non parce que c'est pareil que de rentrer un mot de passe là... 🤔

En effet, et ma réponse sera la même que pour ceux qui se connectent depuis Linux : il faut utiliser un agent SSH. Ce programme va rester en mémoire et retenir votre clé privée. Il ne vous demandera la passphrase qu'une fois au début, puis ensuite vous pourrez vous connecter autant de fois que vous voulez à autant de serveurs que vous voulez sans avoir à rentrer quoi que ce soit. 😊

L'agent SSH Pageant

L'agent SSH installé avec Putty s'appelle "Pageant". Je vous recommande de le lancer au démarrage de l'ordinateur automatiquement (il ne prend que 4 Mo en mémoire), en le plaçant dans le dossier "Démarrage" du menu Démarrer.

Lorsque vous lancez Pageant, une petite icône d'un ordinateur avec un chapeau s'ajoute dans la barre des tâches à côté de l'horloge :



Faites un clic droit dessus, puis cliquez sur "Add key". On vous demande où se trouve la clé privée (cle.ppk). Rentrez ensuite la passphrase.

C'est bon. Vous avez juste besoin de le faire une fois. Maintenant, vous pouvez vous connecter au

serveur que vous voulez en cliquant droit sur l'icône puis en sélectionnant "Saved Sessions" :



On ne vous demandera plus votre clé. 😊

Notez que si l'agent SSH Pageant est pratique, il vaut mieux l'arrêter si vous devez vous absenter de votre ordinateur un long moment et que quelqu'un risque de l'utiliser. Sinon, n'importe qui peut se connecter à vos serveurs sans avoir à rentrer de mot de passe.

Retenez bien : l'agent SSH est un compromis entre la sécurité et le côté pratique. Il retient les clés pour vous (du moins tant que le programme tourne). Si vous êtes un utilisateur intensif de SSH, cela vous fera gagner beaucoup de temps.

Vous pouvez modifier le raccourci qui lance Pageant pour que celui-ci charge votre clé privée automatiquement dès son lancement. Faites un clic-droit sur l'icône de Pageant, allez dans "Propriétés".

Dans le champ "Cible", rajoutez à la fin en paramètre le chemin de la clé à charger. Par exemple : "C:\Program Files\PuTTY\pageant.exe" c:\cle.ppk. La clé sera alors chargée dès que vous lancerez Pageant.
