

# Comment utiliser des clés SSH pour se connecter sans mot de passe

[OpenSSH](#) est un moyen particulièrement sécurisé pour se connecter à distance à une machine Slackware. Mais le moyen le plus facile d'utiliser SSH reste d'utiliser les fonctionnalités de clés.

Le concept de clés publique et privée peut être difficile à expliquer, nous tâcherons de le présenter de manière la plus simple possible.



Permettez-moi de le préciser une fois de plus, à vous tous, crypto-nerds : oui, je sais que c'est une vision très simplifiée de SSH. C'est destiné aux débutants en SSH. Ça roule ?

SSH est principalement basé sur de la cryptographie à clé publique. Cela signifie que vous créez deux clés : une appelée clé publique, et qui est utilisée pour chiffrer les données que vous serez le seul à pouvoir déchiffrer. Vous pouvez donner cette clé à n'importe qui, dès lors qu'elle ne permet que de chiffrer des données, il n'est pas possible de faire grand chose d'autre avec. L'autre clé est nommée clé PRIVEE, et c'est cette clé qui sera utilisée pour déchiffrer les données chiffrées avec la clé publique.

Jusqu'ici tout va bien... Maintenant comment cela est-il utilisé avec SSH ?

Lorsque vous contactez une machine Slackware (ou tout autre machine exécutant OpenSSH, en fait) via le protocole SSH, votre client SSH (le programme installé sur l'ordinateur en face de vous, celui que vous utilisez pour vous connecter) va s'adresser au serveur SSH installé sur la machine distante. Ils vont déterminer ensemble les capacités qu'ils peuvent utiliser tous les deux et quelle version du protocole doit être utilisée pour communiquer de manière sécurisée.

Puis, ils vont essayer de déterminer comment vous (utilisateur) vous pourrez vous connecter sur la machine distante. Si des clés ne sont pas utilisées, SSH utilisera la plupart du temps (mais pas toujours) une demande de mot de passe. En revanche, avec des clés, les machines vont les utiliser de la manière suivante :

1. Le serveur SSH va chiffrer un court message (techniquement, un condensé - *hash*) avec votre clé publique et l'envoyer à votre ordinateur.
2. Votre client SSH va déchiffrer ce message avec votre clé privée (dont le seul exemplaire doit se trouver sur votre ordinateur) et le renvoyer vers le serveur SSH.
3. Le serveur SSH sera alors certain que "vous êtes bien vous". Etant la seule personne (en théorie) capable de déchiffrer le message envoyé, le serveur vous ouvrira l'accès immédiatement.

Si tout cela vous semble un peu compliqué, souvenez-vous simplement de ce point : vous avez une clé publique et une clé privée. La clé publique doit être sur l'ordinateur auquel vous voulez vous connecter, ou ordinateur "distant". La clé privée doit rester sur votre ordinateur.

Passons tout cela en revue étape par étape !

## Créer une paire de clé publique / privée


Pour créer une clé publique et privée, utilisez l'utilitaire `ssh-keygen` d'OpenSSH. Cela générera automatiquement une paire de clés en utilisant les valeurs par défaut. Voici un court exemple :

```
noryungi@mypc:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/noryungi/.ssh/id_rsa): TEST.rsa
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in TEST.rsa.
Your public key has been saved in TEST.rsa.pub.
The key fingerprint is:
1a:99:51:a6:12:69:53:aa:d8:f6:c2:56:66:6e:68:5a noryungi@udon
The key's randomart image is:
+--[ RSA 2048 ]-----+
|      .o. o          |
|      +o +          |
|      .o.o          |
| o . . +           |
| . + + + S         |
| o B   o           |
|  E + .           |
| = o              |
| .                |
+-----+
noryungi@mypc:~$ ls TEST*
TEST.rsa  TEST.rsa.pub
```

Bon, que s'est il passé ?

Premièrement, `ssh-keygen` a généré une paire de clés. Jusqu'ici, tout va bien, prenez le temps de lire la [page de manuel de ssh-keygen](#) pour comprendre toutes les options, et elles sont nombreuses.

Puis `ssh-keygen` indique qu'il crée une paire de clés (publique et privée) RSA. RSA est le nom de l'algorithme de chiffrement utilisé. Trois types de chiffrements sont possibles : DSA, RSA et ECDSA. Le

choix du meilleur est laissé en exercice au lecteur... 

Puis vient une question pour savoir où vous souhaitez sauvegarder votre clé. Dans notre cas, `TEST.rsa` est indiqué afin de pas la confondre avec d'autres clés déjà présentes sur le système. Attribuer un nom de clé significatif est important, car cela rend les choses plus faciles pour savoir quelle clé vous permet de vous connecter à quoi.

Par exemple, si vous avez un accès sur une machine nommée `stang.slackware.com`, `stang_slackware_com.rsa` ou quelque chose de similaire ferait un bon nom de clé.

Puis, `ssh-keygen` vous demande une phrase secrète (*passphrase*). Il est toujours recommandé d'entrer une phrase secrète ! Cela vous permet de protéger votre clé, même si elle tombe entre de mauvaises mains. Si vous êtes absolument certain à 100% que votre clé privée ne tombera **jamaïs** entre de mauvaises mains (vous êtes un grand optimiste), appuyez juste sur Entrée.

Ce qui suit est juste quelques messages informatifs, et vous noterez que la paire de clés a été sauvegardée de la manière suivante :

1. La clé privée est nommée `TEST.rsa`.
2. La clé publique - celle que vous souhaitez copier sur la machine distante - est nommée `TEST.rsa.pub`

Félicitations ! Vous avez parcouru la moitié du chemin !

## Configurer votre clé sur l'ordinateur distant

Très bien, maintenant comment utiliser vos clés publique et privée ? Tout simplement en copiant la clé publique (nommée `TEST.rsa.pub` comme nous l'avons vu) sur l'ordinateur distant. Le meilleur moyen pour cela est d'utiliser `scp` le programme de copie sécurisé d'OpenSSH. Par exemple :

```
noryungi@mypc$ scp TEST.rsa.pub
nr@test.example.com:/home/nr/.ssh/authorized_keys
```

Dans l'exemple ci-dessus, je copie la clé publique `TEST.rsa.pub` sur la machine distante nommée `test.example.com`, en tant qu'utilisateur `nr`. Le fichier est renommé `authorized_keys` qui est le nom du fichier qui doit contenir toutes les clés publiques autorisées pour se connecter au serveur.



**AVERTISSEMENT** : ne lancez pas la commande `scp` ci-dessus si vous avez déjà un fichier `authorized_keys` sur l'ordinateur distant ! Cela écraserait tout le contenu du fichier par votre clé publique ! Si vous avez déjà un fichier `authorized_keys`, exécutez `cat TEST.rsa.pub » authorized_keys` sur la machine distante pour ajouter votre clé publique à la fin de votre fichier de clés autorisées.

Comme toutes les clés que vous utiliserez doivent être dans le répertoire `.ssh`, c'est là qu'elles iront sur la machine distante.

Donc c'est réglé ? Non pas complètement, il reste une petite chose à régler, mais essentielle, car ce peut être la source de beaucoup de problèmes ...

## Vérifier les permissions de la clé publique sur la machine distante

Comme les clés privée et publique sont très sensibles, elles doivent être à l'abri des regards indiscrets. Pour cela, exécutez les commandes suivantes sur les deux machines, locale et distante :

```
nr@test.example.com$ chmod -R -v g-rwx,o-rwx ~/.ssh/
mode of `./ssh/' changed from 0755 (rwxr-xr-x) to 0700 (rwx-----)
mode of `./ssh/authorized_keys' changed from 0644 (rw-r--r--) to 0600 (rw-----)
```

Cette commande vous assure que personne (excepté vous et le serveur SSH) ne pourra lire la clé publique.



Note : si les permissions sur le fichier `authorized_keys` ou sur le répertoire `.ssh` ne sont pas correctes, OpenSSH refusera d'utiliser les clés ! Si vous avez un problème avec des clés privées ou publiques, vérifiez les permissions et/ou exécutez la commande ci-dessus pour être sûr que tout soit correct !

## Se connecter avec la clé SSH nouvellement créée

Essayons de nous connecter, depuis la machine locale, vers le serveur distant nommé `test.example.com` :

```
noryungi@mypc$ ssh -i TEST.rsa nr@test.example.com
```

Notez que j'ai indiqué l'option `-i` juste après la commande `ssh` : cette option sélectionne la clé privée à utiliser pour la connexion, en tant qu'utilisateur `nr` sur le serveur distant nommé `test.example.com`.

Si vous avez choisi de protéger votre clé privée avec une phrase de sécurité (*passphrase*), `ssh` vous demandera cette phrase avant de vous connecter. Sinon ... Si les permissions sont correctes (voir précédemment) vous devriez voir quelque-chose tel que :

```
nr@test.example.com$
```

Ça y est ! Vous êtes connecté à la machine distante, sans avoir eu besoin d'entrer de mot de passe, et avec une sécurité accrue par rapport à un mot de passe - qui pourrait être trouvé, tandis qu'une clé est beaucoup trop longue pour être forcée.

## Qu'est ce qui pourrait ne pas marcher ?

Pas grand chose en fait, excepté la possibilité que votre administrateur système ne souhaite pas que vous vous connectiez avec une paire de clés publique/privée...

Si c'est le cas, admettons-le, ce n'est pas un très bon administrateur système. Toutefois, cela peut être vérifié avec la commande suivante lancée sur la machine distante :

```
nr@test.example.com$ grep -i pubkeyauth /etc/ssh/sshd_config  
#PubkeyAuthentication yes
```

La ligne `#PubkeyAuthentication yes` signifie que c'est la valeur par défaut pour l'authentification à clé publique et comme vous pouvez le voir l'option est positionnée à oui (`yes`). Tout est bon et vous pouvez utiliser votre clé ! À l'opposé, si vous voyez quelque-chose comme ceci :

```
nr@test.example.com$ grep -i pubkeyauth /etc/ssh/sshd_config  
PubkeyAuthentication no
```

Alors vous n'êtes pas autorisé à vous connecter à la machine distante (`test.example.com` ci-dessus) avec une clé publique. Il est temps de contacter votre administrateur système ou votre administrateur sécurité et lui demander poliment de pouvoir le faire.

(Oui, il y a d'autres moyens, plus sournois, pour se connecter sans entrer de mot de passe, mais aucun n'est aussi sûr qu'une paire de clé publique/privée. Peut-être que cela fera l'objet d'une autre documentation sur ce wiki?) 😊

Vous êtes arrivé au terme de cette courte documentation. Allez et utilisez les clés OpenSSH !

## See Also:

\* [The OpenSSH manual pages \(online\)](#)

## Sources

- Version originale par [Noryungi](#)
- Traduction par [Ellendhel](#)

[security](#), [ssh](#), [sshkeys](#)

From:

<https://docs.slackware.com/> - **SlackDocs**

Permanent link:

<https://docs.slackware.com/fr:howtos:security:sshkeys>

Last update: **2012/10/10 00:27 (UTC)**

