ETTORI Bastien	**BTS SIO 2ème année**
23 mars 2016	**Année scolaire : 2015/2016**
Option : SISR	**Version 1.1**

OPENSSH DEBIAN

SOMMAIRE :

I) Objectif

Dans cette procédure, nous allons voir comment configurer un serveur **SSH** sur la même machine du serveur **SSL** sous Debian.

II) Prérequis

Pour réaliser cette procédure, nous avons besoin des éléments suivants :

- Un serveur **SSL** fonctionnel.

Nombre de machines	SE serveur SSL /SSH	Nom du serveur SSL / SSH	Adresse IP du serveur SSL	Utilitaire
2	**Debian 7.7**	**debian**	**192.168.1.108**	**Putty**

III) Définitions

- **OpenSSL** (**Open S**ecure **S**ocket **L**ayer) est une boîte à outils informatiques qui permet de chiffrer et d'échanger des données entre 2 ou plusieurs ordinateurs à distance de manière sécurisée.
- **OpenSSH** (**Open S**ecure **SH**ell) est un ensemble d'outils informatiques qui permet d'établir des communications sécurisées à travers un réseau en utilisant le protocole **SSH**. Il peut également sécuriser plusieurs connexions et chiffre le trafic réseau pour éviter les attaques et les contrôles de connexion.
- Une **clef publique** est une clef publiable qui permet d'authentifier, de vérifier et de chiffrer des données.
- A l'inverse d'une clef publique, une **clef privée** est une clef confidentielle mais celle-ci possède les mêmes fonctions qu'une clef publique.

IV) Installation OpenSSH

- Tout d'abord, nous mettons à jour les paquets :

```
root@debian:~# apt-get update
```

- Nous devons installer le paquet « **openssh-server** » (si celui-ci n'est pas installé automatiquement) :

```
root@debian:~# apt-get install openssh-server
```

- (Ou nous installons le paquet « **ssh** » :

```
root@debian:~# apt-get install ssh
```

V) Connexion d'un utilisateur via le service SSH

- Nous allons nous connecter avec un utilisateur via **SSH**, Ici l'utilisateur se nomme « **bastien** ».

```
root@debian:~# ssh bastien@192.168.1.108
```

- Nous répondons « **yes** » pour continuer la connexion :

```
The authenticity of host '192.168.1.108 (192.168.1.108)' can't be established.
ECDSA key fingerprint is dd:c4:f2:fe:78:38:62:6c:6f:95:c7:44:e8:e4:b2:a0.
Are you sure you want to continue connecting (yes/no)? yes
```

- Nous saisissons son mot de passe :

```
bastien@192.168.1.108's password:
Linux debian 3.2.0-4-amd64 #1 SMP Debian 3.2.73-2+deb7u3 x86_64

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Mar 23 08:54:09 2016
bastien@debian:~$
```

VI) Génération des clefs privées et plubliques

- Une fois connecté en tant qu'utilisateur, nous allons générer les clefs privée et publique sur la machine cliente :

```
bastien@debian:~$ ssh-keygen -t rsa
```

- Nous ne saisissons rien et nous tapons directement sur « **Entrer** » :

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/bastien/.ssh/id_rsa):
Created directory '/home/bastien/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/bastien/.ssh/id_rsa.
Your public key has been saved in /home/bastien/.ssh/id_rsa.pub.
The key fingerprint is:
5d:87:1e:49:e4:97:c3:88:ba:6f:49:3c:4a:91:a5:9a bastien@debian
The key's randomart image is:
+--[ RSA 2048]----+
|           .o    |
|          .+ = . |
|         +. B *  |
|        +o o + . |
|       oSo. .    |
|      E ..+      |
|       ..o o     |
|         ..o     |
|          ..     |
+-----------------+
bastien@debian:~$
```

- Nous listons le répertoire de la machine cliente dans le dossier « **.ssh** » et nous constatons que les clefs ont bien été générées :

```
bastien@debian:~$ ls -a .ssh/
.  ..  id_rsa  id_rsa.pub
bastien@debian:~$ _
```

- Ensuite, nous copions la clef publique du serveur en se connectant avec le compte « **root** » de la manière suivante :

```
bastien@debian:~$ ssh-copy-id -i .ssh/id_rsa.pub root@192.168.1.108
```

- Nous tapons « **yes** » pour continuer la connexion et nous saisissons le mot de passe « **root** » :

```
The authenticity of host '192.168.1.108 (192.168.1.108)' c
ECDSA key fingerprint is dd:c4:f2:fe:78:38:62:6c:6f:95:c7:
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.108' (ECDSA) to the
root@192.168.1.108's password:
Now try logging into the machine, with "ssh 'root@192.168.

  ~/.ssh/authorized_keys

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

bastien@debian:~$ _
```

VII) Connexion avec « Putty » via une authentification de clef

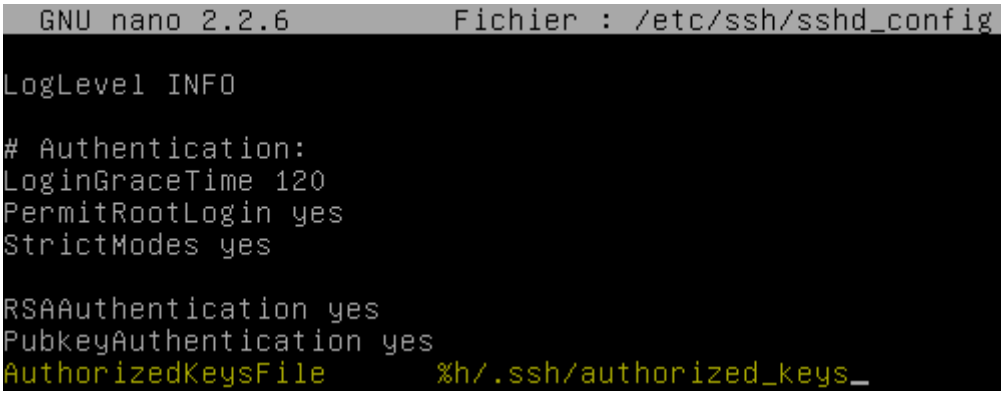
- Nous nous déconnectons de l'utilisateur « **bastien** » :

```
bastien@debian:~$ exit
déconnexion
Connection to 192.168.1.108 closed.
root@debian:~# _
```

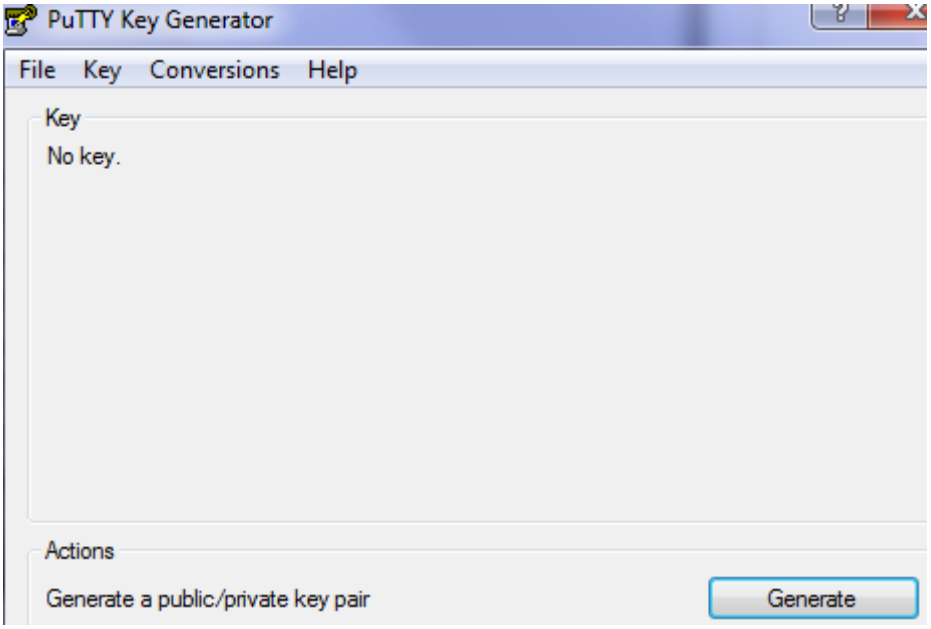
- Ensuite, une fois que nous sommes reconnecté en tant que « **root** », nous éditons le fichier « **sshd_config** » qui se situe dans le dossier « **/etc/ssh** » :

```
root@debian:~# nano /etc/ssh/sshd_config
```

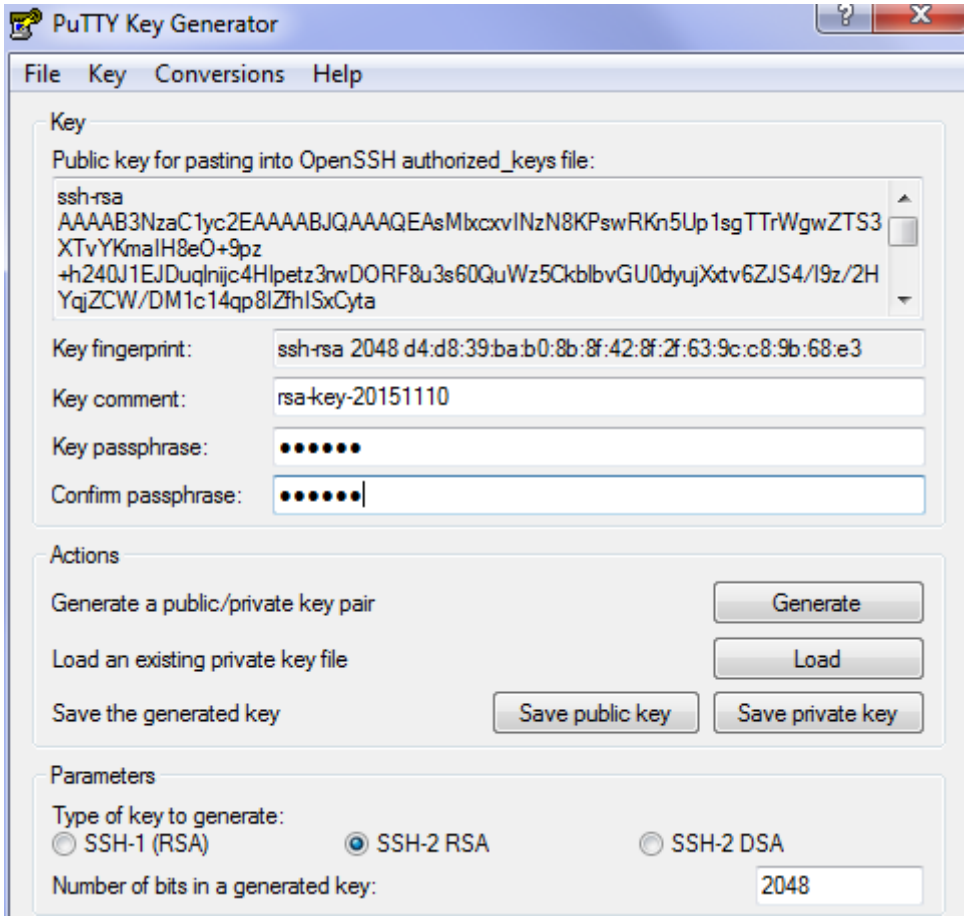
- Nous décommentons la ligne « **AuthorizedKeysFile** » pour autoriser l'authentification par clef :



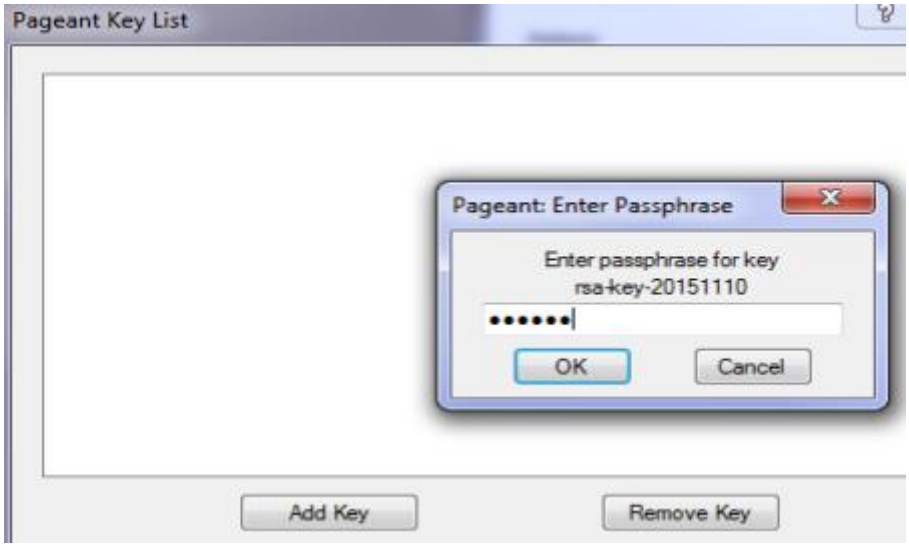
- Maintenant, nous nous rendons dans la recherche de programmes Windows et nous ouvrons l'utilitaire « **PuTTYgen** » qui ouvre l'outil de génération de clef « **PuTTY Key Generator** » et nous cliquons sur « **Generate** » pour générer une clef (Cela peut prendre quelques minutes) :



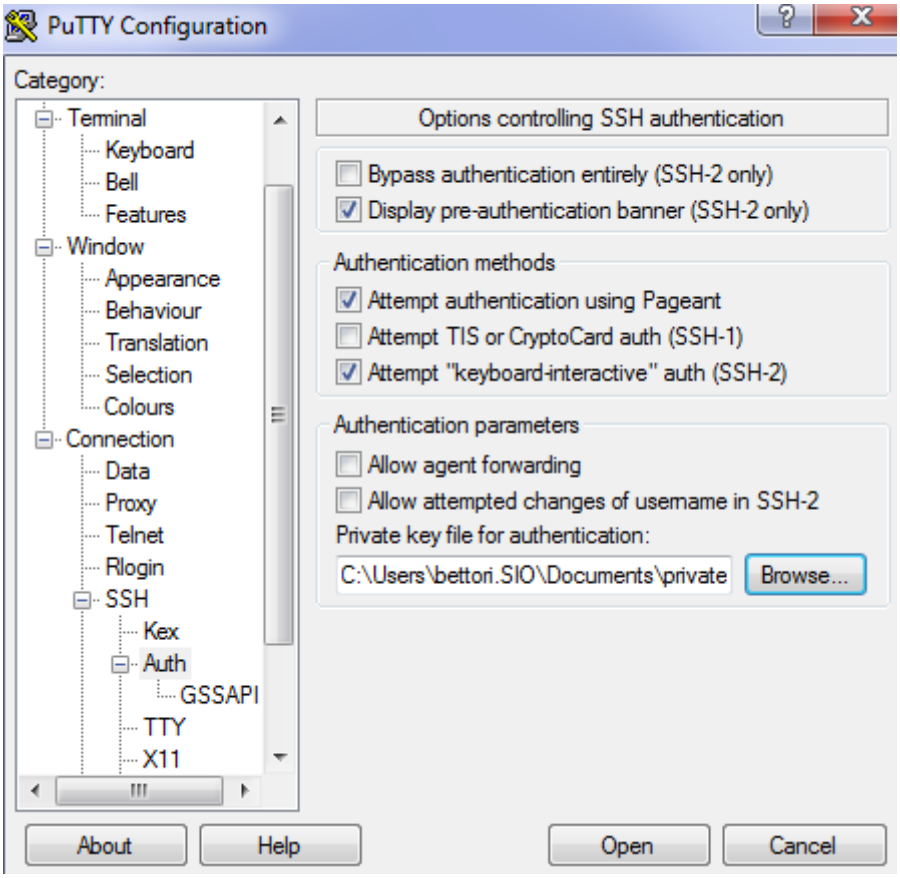
- Une fois que la clef est générée, nous remplissons les champs de saisie « **key passphrase** » et « **Confirm passphrase** » pour la sécuriser et nous sauvegardons les 2 clefs en cliquant sur « **Save public key** » et « **Save private key** » et nous les copions par FTP dans le dossier « **.ssh** » :



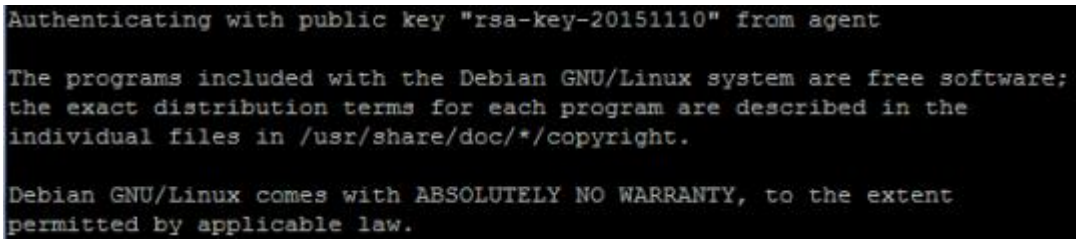
- Ensuite, nous ouvrons « **Pageant** » pour « **Putty** », nous saisissons le passe de la clef **RSA** et nous cliquons sur « **Add Key** » :



- Nous revenons sur « **Putty** » et nous nous rendons dans « **SSH** » et « **Auth** » pour parcourir l'emplacement de la clef :



- Nous nous connectons en tant qu'utilisateur sur « **Putty** » :



Nous pouvons constater que nous sommes connectés avec la clef que nous avons générée et enregistrée.

VIII) Conclusion

En conclusion, nous pouvons dire que le serveur **SSH** est opérationnel et il permet aux utilisateurs de se connecter par authentification de clef ainsi que le transfert sécurisé de fichiers via le protocole **SCP** (**S**ecure **C**opy **P**rotocol).