PaSSHport Documentation

LibrIT

juil. 08, 2023

Table des matières

Votre admin sys quitte votre société. Êtes-vous sûrs que tous ses accès ssh sont révoqués ? Et les stagiaires ? Les prestataires ?... Réglons ça.

CHAPITRE 1

Table des matières :

1.1 Introduction à PaSSHport

1.1.1 Qu'est-ce que PaSSHport ?

PaSSHport est un logiciel qui vous permet de contrôler les accès SSH des briques de votre IT : serveurs Linux/Unix, switchs réseaux, routeurs, points d'accès Wi-Fi, ainsi que n'importe quelle brique accessible via SSH. En quelques mots : qui a accès à quoi ?

PaSSHport a été écrit avec les objectifs suivants :

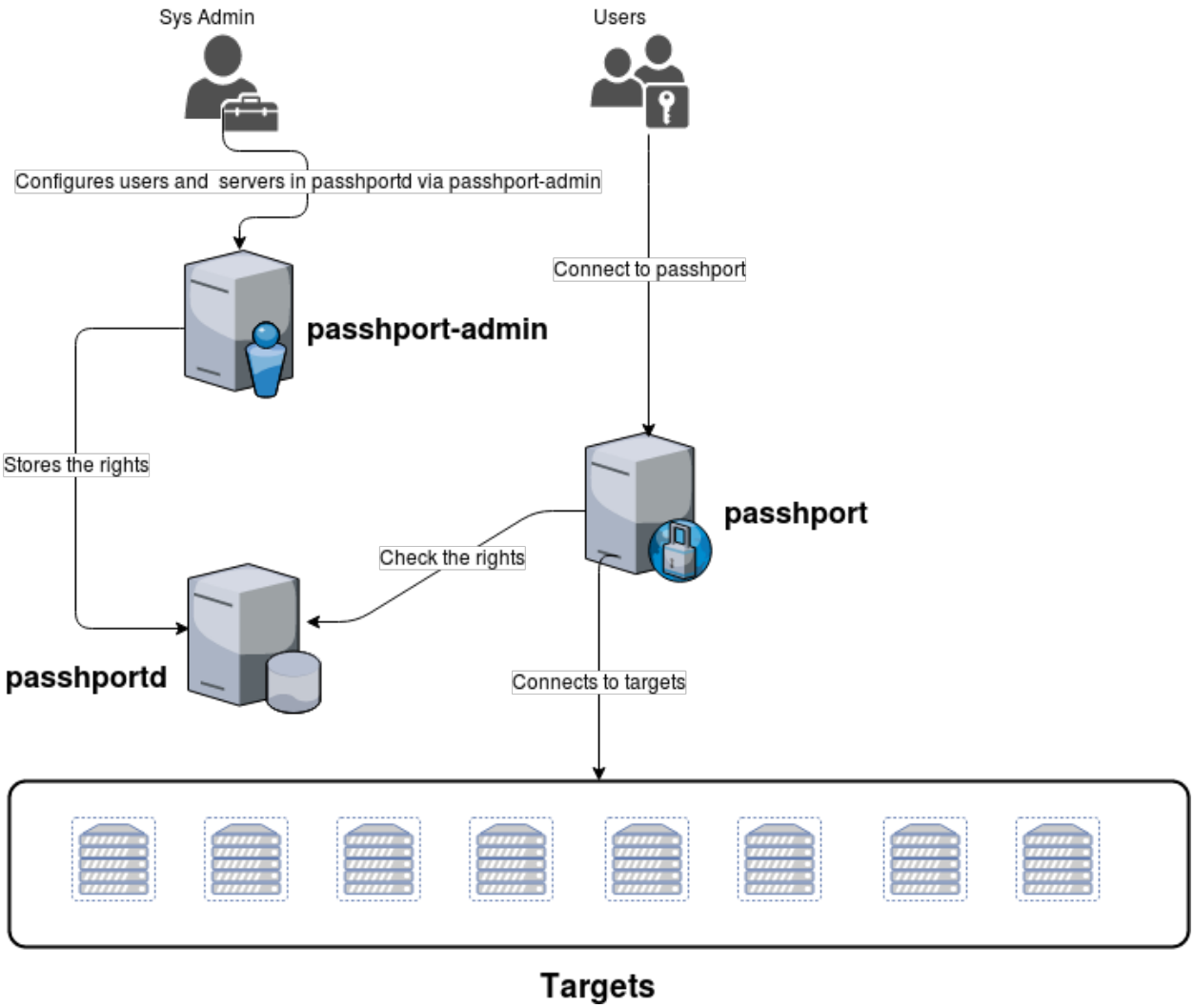
- Même principes que SSHgate
- Deux types d'objets : targets et users (Nous verrons ça un peu plus bas)
- Les objets peuvent être groupés
- Enregistrer toutes les sessions des utilisateurs
- Être entièrement configurable et utilisable depuis la ligne de commandes
- Possibilité de faire des copies sécurisées (scp)
- Des communications entre les composants basées sur une API REST, pour une intégration plus aisée dans un IT automatisé
- Utiliser des technologies OpenSource

Pour comprendre les composants principaux, et comment ils interagissent entre eux, veuillez lire ce qui suit.

1.1.2 Composants

PaSSHport est composé de 3 programmes :

- passhportd : le démon qui enregistre, et vérifie la configuration
- passhport : le script qui reçoit les connexions (il ne remplace PAS un serveur SSH). Il faut le voir comme le shell sur lequel les utilisateurs arrivent quand ils se connectent à PaSSHport.
- passhport-admin : le script qui permet de configurer passhportd. Les adminSys l'utilisent pour, par exemple, ajouter un *user*, une *target*, un *usergroup*, un *targetgroup*, et combiner ces derniers pour configurer les accès



Et maintenant, passons à l'installation. . .

1.2 Installation et configuration

Ce chapitre décrit l'installation de PaSSHport sur différentes distributions Linux, et certains principes de configuration de base.

1.2.1 Installation sur Debian 8, 9 ou 10

Les paragraphes suivant expliquent comment installer et faire tourner PaSSHport sur Debian 8 (Jessie), 9 (Stretch) ou 10 (Buster). On part d'une installation de minimal de Debian (disponible ici), **avec les paquets openssh-server et curl** installés.

La manière simple, et automatisée

Le script d'installation est disponible pour relecture ici.

Vous pouvez le lancer directement depuis la ligne de commande (assurez-vous une fois de plus que curl est bien installé : `apt install curl`) :

```
root@debian:~# bash <(curl -s https://raw.githubusercontent.com/librit/passhport/
↪master/tools/passhport-install-script-debian.sh)
```

Une fois l'installation terminée, rendez-vous au chapitre Première configuration.

La manière longue, manuelle

Si vous voulez comprendre ce qui se passe sur votre système, quand vous installez PaSSHport, suivez les instruction suivantes, qui sont (grossièrement) les commandes, une à une, du script d'installation automatisé sus-mentionné.

Tout d'abord, on update les dépots :

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

On installe python3-pip, et d'autre packages dont nous aurons besoin plus tard pour ce tuto (il y a environ 100Mo à récupérer sur les dépôts, donc soyer patient) :

```
root@debian:~# apt install python3-pip git openssl virtualenv libpython3-dev
```

Ensuite, nous aurons besoin d'ajouter un utilisateur appelé « passhport », et exécuter quelques commande en tant que lui :

```
root@debian:~# useradd --home-dir /home/passhport --shell /bin/bash --create-home␣
↪passhport
root@debian:~# su - passhport
passhport@debian:~$
```

On récupère les sources de PaSSHport sur github :

```
passhport@debian:~$ git clone http://github.com/LibrIT/passhport.git
Clonage dans 'passhport'...
remote: Counting objects: 2713, done.
remote: Compressing objects: 100% (50/50), done.
remote: Total 2713 (delta 19), reused 0 (delta 0), pack-reused 2661
Réception d'objets: 100% (2713/2713), 482.76 KiB | 396.00 KiB/s, fait.
Résolution des deltas: 100% (1633/1633), fait.
passhport@debian:~$
```

On crée un environnement «virtuel-env» pour cette utilisateur passhport :

```
passhport@debian:~$ virtualenv -p python3 passhport-run-env
```

Maintenant que nous avons notre «virtual-env», on installe les modules python nécessaires au fonctionnement de PaSSHport :

```
passhport@debian:~$ /home/passhport/passhport-run-env/bin/pip install -r /home/
↪passhport/passhport/requirements.txt
```

Commençons les choses sérieuses…

PaSSHport a besoin d'écrire des logs, il faut donc créer un répertoire dans «/var/log», et en attribuer la propriété à l'utilisateur «passhport» :

```
root@debian:~# mkdir -p /var/log/passhport/
root@debian:~# chown passhport:passhport /var/log/passhport/
```

On crée aussi le répertoire qui contiendra la configuration, et on copie les différents fichiers de configuration dedans :

```
root@debian:~# mkdir /etc/passhport
root@debian:~# cp /home/passhport/passhport/passhport/passhport.ini /etc/passhport/.
root@debian:~# cp /home/passhport/passhport/passhport-admin/passhport-admin.ini /etc/
↪passhport/.
root@debian:~# cp /home/passhport/passhport/passhportd/passhportd.ini /etc/passhport/.
```

Nous devons faire quelques modifications dans ces fichiers de configurations, surtout si on souhaite séparer les composants sur différentes machines. On remplace ici l'adresse d'écoute par défaut (localhost), avec la *vrai* IP du serveur.

D'abord passhportd :

```
root@debian:~# vim /etc/passhport/passhportd.ini
```

On change la directive « LISTENING_IP », par l'adresse IP du serveur :

```
# Passhportd configuration file. You should copy it to
# /etc/passhport/passhportd.ini if you want to do modifications
[SSL]
SSL = True
SSL_CERTIFICAT = /home/passhport/certs/cert.pem
SSL_KEY = /home/passhport/certs/key.pem

[Network]
LISTENING_IP = 192.168.122.56
PORT = 5000

[Database]
SQLALCHEMY_TRACK_MODIFICATIONS = True
SQLALCHEMY_DATABASE_DIR = /var/lib/passhport/
SQLALCHEMY_MIGRATE_REPO = /var/lib/passhport/db_repository
# For SQLite
SQLALCHEMY_DATABASE_URI = sqlite:////var/lib/passhport/app.db

[Environment]
# SSH Keyfile path
SSH_KEY_FILE = /home/passhport/.ssh/authorized_keys
# Python and passhport paths
PASSHPORT_PATH = /home/passhport/passhport/passhport/passhport
PYTHON_PATH = /home/passhport/passhport-run-env/bin/python3
```

On change le paramètre suivante dans /etc/passhport/passhport.ini et /etc/passhport/passhport-admin.ini :

```
PASSHPORTD_HOSTNAME = 192.168.122.56
```

Nous aurons besoin d'une clef SSH. On en génère donc une RSA de 4096 bits (la taille de la clef peut être plus grande) :

```
root@debian:~# su - passhport
passhport@debian:~$ ssh-keygen -t rsa -b 4096 -N "" -f "/home/passhport/.ssh/id_rsa"
Generating public/private rsa key pair.
Your identification has been saved in /home/passhport/.ssh/id_rsa.
Your public key has been saved in /home/passhport/.ssh/id_rsa.pub.
The key fingerprint is:
```

(suite sur la page suivante)

(suite de la page précédente)

```
SHA256:0o6jkepqr2Phz0AKmLGRZh6PeVexP2gf5CGNPd+ksQ passhport@debian
The key's randomart image is:
+---[RSA 4096]----+
| .    ....       |
|oo . o .+ +      |
|* + o ...= *     |
|.O   o oo + E    |
|=.    LibrIT .   |
|+.   .Rocks = .  |
|o.. o o .  . o   |
| =o. o .         |
|++B+.            |
+----[SHA256]-----+
passhport@debian:~$
```

Cette clé sera utilisé par passhport pour se connecter aux différents serveurs. On peut aussi générer une clef ECDSA si on veut :

```
passhport@debian:~$ ssh-keygen -t ecdsa -b 521 -N "" -f "/home/passhport/.ssh/id_ecdsa
↪"
```

Une fois encore en tant que root, on crée le répertoire qui contiendra la base de données (parce qu'on utilise SQLite pour ce tuto) :

```
root@debian:~# mkdir -p /var/lib/passhport
root@debian:~# chown -R passhport:passhport /var/lib/passhport/
```

… on a alors 3 paramètres à changer dans le fichier de conf de passhportd (en tant que root, on édite «/etc/passhport/passhportd.ini») :

```
SQLALCHEMY_DATABASE_DIR         = /var/lib/passhport/
SQLALCHEMY_MIGRATE_REPO         = /var/lib/passhport/db_repository
SQLALCHEMY_DATABASE_URI         = sqlite:////var/lib/passhport/app.db
```

On peut maintenant créer la base de données, et vérifier que celle-ci a été correctement créé :

```
root@debian:~# su - passhport
passhport@debian:~$ /home/passhport/passhport-run-env/bin/python /home/passhport/
↪passhport/passhportd/db_create.py
passhport@debian:~$ ls -la /var/lib/passhport/
total 172
drwxr-xr-x  3 passhport passhport   4096 févr. 28 16:10 .
drwxr-xr-x 25 root      root        4096 févr. 28 15:37 ..
-rw-r--r--  1 passhport passhport 159744 févr. 28 16:10 app.db
drwxr-xr-x  4 passhport passhport   4096 févr. 28 16:10 db_repository
passhport@debian:~$
```

On va maintenant créer un certificat pour sécuriser les échanges avec l'API. D'abord, on crée le répertoire dans lequel se trouveront la clé privée et le certificat. Il faut aussi attribué les droits rwx à l'utilisateur «passhport» seulement :

```
passhport@debian:~$ mkdir /home/passhport/certs
passhport@debian:~$ chmod 700 /home/passhport/certs
```

On crée la clé RSA :

```
[passhport@centos-7 ~]$ openssl genrsa -out "/home/passhport/certs/key.pem" 4096
```

Il y a un fichier de configuration pour OpenSSL fourni avec les sources de PaSSHport, pour générer un certificat minimal SSL correcte. Le fichier est :

`/home/passhport/passhport/tools/openssl-for-passhportd.cnf`

On l'édite, et on ajoute de nom DNS dont on se servira pour joindre l'API. Pour ce tuto, on utilisera deux noms d'hôtes :

```
[req]
distinguished_name      = req_distinguished_name
req_extensions          = v3_req
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always,issuer

[v3_req]
subjectAltName          = @alternate_names
basicConstraints        = CA:TRUE
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always,issuer

[req_distinguished_name]

[ alternate_names ]
DNS.1 = 127.0.0.1
DNS.2 = localhost
DNS.3 = passhport.librit.fr
DNS.4 = entry.passhport.org
```

On génère le certificat avec la commande suivante (on peut faire un copié/collé des lignes suivantes). Par contre, il faut bien entendu adapter la ligne du sujet (-subj) à votre installation :

```
openssl req -new -key "/home/passhport/certs/key.pem" \
-config "/home/passhport/passhport/tools/openssl-for-passhportd.cnf" \
-out "/home/passhport/certs/cert.pem" \
-subj "/C=FR/ST=Ile De France/L=Ivry sur Seine/O=LibrIT/OU=DSI/CN=passhport.librit.fr
↪" \
-x509 -days 365 -sha256 \
-extensions v3_req
```

Une fois exécuté, vous aurez un certificat à côté d'une clé :

```
passhport@debian:~$ ls -la /home/passhport/certs/
total 16
drwx------ 2 passhport passhport 4096 févr. 28 18:00 .
drwxr-xr-x 8 passhport passhport 4096 févr. 28 17:46 ..
-rw-r--r-- 1 passhport passhport 2171 févr. 28 18:00 cert.pem
-rw------- 1 passhport passhport 3243 févr. 28 16:11 key.pem
passhport@debian:~$
```

En tant que root, on crée deux liens symboliques vers les deux principaux, passhportd et passhport-admin, pour ne plus avoir à besoin de taper les chemin complet :

```
root@debian:~# ln -s /home/passhport/passhport/tools/passhportd.sh /usr/bin/passhportd
root@debian:~# ln -s /home/passhport/passhport/tools/passhport-admin.sh /usr/bin/
↪passhport-admin
```

On peut créer un service systemd, et activer *passhportd* au démarrage :

```
root@debian:~# cp /home/passhport/passhport/tools/passhportd.service /etc/systemd/
↪system/passhportd.service
root@debian:~# systemctl daemon-reload
root@debian:~# systemctl enable passhportd
```

Il n'y a plus qu'à démarrer le démon passhportd :

```
root@debian:~# systemctl start passhportd
```

On peut maintenant vérifier que passhportd tourne correctement, en "curlant" l'IP qu'on a précédemment configurée dans */etc/passhport/passhportd.ini*, sur le port 5000 :

```
root@debian:~# curl -s --insecure https://192.168.122.56:5000
passhportd is running, gratz!
root@debian:~#
```

Well done ! Vous avez installé PaSSHport. Vous pouvez maintenant lire le chapitre Première utilisation.

1.2.2 Installation sur CentOS 7

Les paragraphes suivant expliquent comment installer et faire tourner PaSSHport sur CentOS 7. On part d'une installation de minimal de CentOS 7 (disponible ici).

La manière simple, et automatisée

Le script d'installation est disponible pour relecture ici.

Vous pouvez le lancer directement depuis la ligne de commande :

```
root@centos7:~# bash <(curl -s https://raw.githubusercontent.com/librit/passhport/
↪master/tools/passhport-install-script-centos7.sh)
```

Une fois l'installation terminée, rendez-vous au chapitre Première configuration.

La manière longue, manuelle

Si vous voulez comprendre ce qui se passe sur votre système, quand vous installez PaSSHport, suivez les instruction suivantes, qui sont (grossièrement) les commandes, une à une, du script d'installation automatisé sus-mentionné.

On installe le dépot EPEL :

```
yum install epel-release
```

On installe python34-pip, et d'autre packages dont nous aurons besoin plus tard pour ce tuto (il y a environ 100Mo à récupérer sur les dépôts, donc soyez patient) :

```
root@centos7:~# yum install python34-pip git openssl python34-devel gcc libffi-devel
```

On mets à jour pip :

```
root@centos7:~# pip3 install -U pip
```

Now, install virtual-env and a mandatory lib using pip :

```
root@centos7:~# pip3 install virtualenv pathlib2
```

Ensuite, nous aurons besoin d'ajouter un utilisateur appelé « passhport », et exécuter quelques commande en tant que lui :

```
root@centos7:~# useradd --home-dir /home/passhport --shell /bin/bash --create-home␣
↪passhport
root@centos7:~# su - passhport
passhport@centos7:~$
```

On crée un environnement «virtuel-env» pour cette utilisateur passhport :

```
passhport@centos7:~$ virtualenv -p python3 passhport-run-env
```

On récupère les sources de PaSSHport sur github :

```
passhport@centos7:~$ git clone http://github.com/LibrIT/passhport.git
Clonage dans 'passhport'...
remote: Counting objects: 2713, done.
remote: Compressing objects: 100% (50/50), done.
remote: Total 2713 (delta 19), reused 0 (delta 0), pack-reused 2661
Réception d'objets: 100% (2713/2713), 482.76 KiB | 396.00 KiB/s, fait.
Résolution des deltas: 100% (1633/1633), fait.
passhport@centos7:~$
```

Maintenant que nous avons notre «virtual-env», on installe les modules python nécessaires au fonctionnement de PaSSHport :

```
passhport@centos7:~$ /home/passhport/passhport-run-env/bin/pip install -r /home/
↪passhport/passhport/requirements.txt
```

Commençons les choses sérieuses…

PaSSHport a besoin d'écrire des logs, il faut donc créer un répertoire dans «/var/log», et en attribuer la propriété à l'utilisateur «passhport» :

```
root@centos7:~# mkdir -p /var/log/passhport/
root@centos7:~# chown passhport:passhport /var/log/passhport/
```

On crée aussi le répertoire qui contiendra la configuration, et on copie les différents fichiers de configuration dedans :

```
root@centos7:~# mkdir /etc/passhport
root@centos7:~# cp /home/passhport/passhport/passhport/passhport.ini /etc/passhport/.
root@centos7:~# cp /home/passhport/passhport/passhport-admin/passhport-admin.ini /etc/
↪passhport/.
root@centos7:~# cp /home/passhport/passhport/passhportd/passhportd.ini /etc/passhport/
↪.
```

Nous devons faire quelques modifications dans ces fichiers de configurations, surtout si on souhaite séparer les composants sur différentes machines. On remplace ici l'adresse d'écoute par défaut (localhost), avec la *vrai* IP du serveur.

D'abord passhportd :

```
root@centos7:~# vim /etc/passhport/passhportd.ini
```

On change la directive « LISTENING_IP », par l'adresse IP du serveur :

```
# Passhportd configuration file. You should copy it to
# /etc/passhport/passhportd.ini if you want to do modifications
[SSL]
SSL = True
SSL_CERTIFICAT = /home/passhport/certs/cert.pem
SSL_KEY = /home/passhport/certs/key.pem

[Network]
LISTENING_IP = 192.168.122.56
PORT = 5000

[Database]
SQLALCHEMY_TRACK_MODIFICATIONS = True
SQLALCHEMY_DATABASE_DIR = /var/lib/passhport/
SQLALCHEMY_MIGRATE_REPO = /var/lib/passhport/db_repository
# For SQLite
SQLALCHEMY_DATABASE_URI = sqlite:////var/lib/passhport/app.db

[Environment]
# SSH Keyfile path
SSH_KEY_FILE = /home/passhport/.ssh/authorized_keys
# Python and passhport paths
PASSHPORT_PATH = /home/passhport/passhport/passhport/passhport
PYTHON_PATH = /home/passhport/passhport-run-env/bin/python3
```

On change le paramètre suivante dans /etc/passhport/passhport.ini et /etc/passhport/passhport-admin.ini :

```
PASSHPORTD_HOSTNAME = 192.168.122.56
```

Nous aurons besoin d'une clef SSH. On en génère donc une RSA de 4096 bits :

```
root@centos7:~# su - passhport
passhport@centos7:~$ ssh-keygen -t rsa -b 4096 -N "" -f "/home/passhport/.ssh/id_rsa"
Generating public/private rsa key pair.
Your identification has been saved in /home/passhport/.ssh/id_rsa.
Your public key has been saved in /home/passhport/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:0o6jkepqr2Phz0AKmLGRZh6PeVexP2gf5CGNPd+ksQ passhport@centos7
The key's randomart image is:
+---[RSA 4096]----+
| .    ....       |
|oo . o .+ +      |
|* + o ...= *     |
|.O   o oo + E    |
|=.    LibrIT .   |
|+.   .Rocks = .  |
|o.. o o .  . o   |
| =o. o .         |
|++B+.            |
+----[SHA256]-----+
passhport@centos7:~$
```

Cette clé sera utilisé par passhport pour se connecter aux différents serveurs. On peut aussi générer une clef ECDSA si on veut :

```
passhport@centos7:~$ ssh-keygen -t ecdsa -b 521 -N "" -f "/home/passhport/.ssh/id_
↪ecdsa"
```

Une fois encore en tant que root, on crée le répertoire qui contiendra la base de données (parce qu'on utilise SQLite

pour ce tuto) :

```
root@centos7:~# mkdir -p /var/lib/passhport
root@centos7:~# chown -R passhport:passhport /var/lib/passhport/
```

… on a alors 3 paramètres à changer dans le fichier de conf de passhportd (en tant que root, on édite «/etc/passhport/passhportd.ini») :

```
SQLALCHEMY_DATABASE_DIR        = /var/lib/passhport/
SQLALCHEMY_MIGRATE_REPO        = /var/lib/passhport/db_repository
SQLALCHEMY_DATABASE_URI        = sqlite:////var/lib/passhport/app.db
```

On peut maintenant créer la base de données, et vérifier que celle-ci a été correctement créé :

```
root@centos7:~# su - passhport
passhport@centos7:~$ /home/passhport/passhport-run-env/bin/python /home/passhport/
↪passhport/passhportd/db_create.py
passhport@centos7:~$ ls -la /var/lib/passhport/
total 172
drwxr-xr-x  3 passhport passhport   4096 févr. 28 16:10 .
drwxr-xr-x 25 root      root        4096 févr. 28 15:37 ..
-rw-r--r--  1 passhport passhport 159744 févr. 28 16:10 app.db
drwxr-xr-x  4 passhport passhport   4096 févr. 28 16:10 db_repository
passhport@centos7:~$
```

On va maintenant créer un certificat pour sécuriser les échanges avec l'API. D'abord, on crée le répertoire dans lequel se trouveront la clé privée et le certificat. Il faut aussi attribué les droits rwx à l'utilisateur «passhport» seulement :

```
passhport@centos7:~$ mkdir /home/passhport/certs
passhport@centos7:~$ chmod 700 /home/passhport/certs
```

On crée la clé RSA :

```
[passhport@centos-7 ~]$ openssl genrsa -out "/home/passhport/certs/key.pem" 4096
```

Il y a un fichier de configuration pour OpenSSL fourni avec les sources de PaSSHport, pour générer un certificat minimal SSL correcte. Le fichier est :

`/home/passhport/passhport/tools/openssl-for-passhportd.cnf`

On l'édite, et on ajoute de nom DNS dont on se servira pour joindre l'API. Pour ce tuto, on utilisera deux noms d'hôtes :

```
[req]
distinguished_name      = req_distinguished_name
req_extensions          = v3_req
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always,issuer

[v3_req]
subjectAltName          = @alternate_names
basicConstraints        = CA:TRUE
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always,issuer

[req_distinguished_name]

[ alternate_names ]
```

(suite sur la page suivante)

(suite de la page précédente)

```
DNS.1 = localhost
DNS.2 = passhport.librit.fr
DNS.3 = entry.passhport.org
```

On génère le certificat avec la commande suivante (on peut faire un copié/collé des lignes suivantes). Par contre, il faut bien entendu adapter la ligne du sujet (-subj) à votre installation :

```
openssl req -new -key "/home/passhport/certs/key.pem" \
-config "/home/passhport/passhport/tools/openssl-for-passhportd.cnf" \
-out "/home/passhport/certs/cert.pem" \
-subj "/C=FR/ST=Ile De France/L=Ivry sur Seine/O=LibrIT/OU=DSI/CN=passhport.librit.fr
↪" \
-x509 -days 365 -sha256 \
-extensions v3_req
```

Une fois exécuté, vous aurez un certificat à côté d'une clé :

```
passhport@centos7:~$ ls -la /home/passhport/certs/
total 16
drwx------ 2 passhport passhport 4096 févr. 28 18:00 .
drwxr-xr-x 8 passhport passhport 4096 févr. 28 17:46 ..
-rw-r--r-- 1 passhport passhport 2171 févr. 28 18:00 cert.pem
-rw------- 1 passhport passhport 3243 févr. 28 16:11 key.pem
passhport@centos7:~$
```

En tant que root, on crée deux liens symboliques vers les deux principaux, passhportd et passhport-admin, pour ne plus avoir à besoin :

```
root@centos7:~# ln -s /home/passhport/passhport/tools/passhportd.sh /usr/bin/
↪passhportd
root@centos7:~# ln -s /home/passhport/passhport/tools/passhport-admin.sh /usr/bin/
↪passhport-admin
```

On peut créer un service systemd, et activer *passhportd* au démarrage :

```
root@centos7:~# cp /home/passhport/passhport/tools/passhportd.service /etc/systemd/
↪system/passhportd.service
root@centos7:~# systemctl daemon-reload
root@centos7:~# systemctl enable passhportd
```

Il n'y a plus qu'à démarrer le démon passhportd :

```
root@centos7:~# systemctl start passhportd
```

On peut maintenant vérifier que passhportd tourne correctement, en "curlant" l'IP qu'on a précédemment configurée dans */etc/passhport/passhportd.ini*, sur le port 5000 :

```
root@centos7:~# curl -s --insecure https://192.168.122.56:5000
passhportd is running, gratz!
root@centos7:~#
```

Well done ! Vous avez installé PaSSHport. Vous pouvez maintenant lire le chapitre Première utilisation.

1.2.3 Utiliser PostgreSQL comme base de données

Installer le module python psycopg2 et psycopg2-binary

Si vous n'avez pas utilisé la version paqueté de PaSSHport (dep/rpm), procédez comme suit. Si vous avez utilisé la version packageé, aller directement à la section *Configuration de PostgreSQL*.

Avant d'installer les librairie Python, assurez-vous d'avoir le binaire *pg_config* dans votre variable d'environnemen *$PATH* , ainsi que d'autre outils liés.

Pour Debian, installez *postgresql*

```
# apt install postgresql
```

Pour CentOS, installez *postgresql* :

```
# yum install postgresql
```

Si vous voulez utilisez PostgreSQL comme base de données, vous devez installer le module python *psycopg2* et *psycopg2-binary*.

En tant qu'utilisateur passhport, on installe *psycopg2* et *psycopg2-binary* :

```
$ /home/passhport/passhport-run-env/bin/pip install psycopg2 psycopg2-binary
```

Configuration de PostgreSQL

On créé un utilisateur passhport dans PostgreSQL (la méthode peut varier selon votre distribution) :

```
# su - postgres
$ createuser -D -S -R passhport && createdb -O passhport "passhport"
```

On ajoute un mot de passe à cet utilisateur :

```
$ psql
psql (9.2.18)
Type "help" for help.

postgres=# ALTER USER "passhport" WITH PASSWORD 'MySUpErp45sw0rD';
ALTER ROLE
postgres=# \q
$
```

Configuration de passhportd

On modifie le fichier de configuration *passhportd.ini*(`/etc/passhport/passhportd.ini`). On change le paramètre `SQLALCHEMY_DATABASE_URI` :

```
SQLALCHEMY_DATABASE_URI         = postgresql://passhport:MySUpErp45sw0rD@localhost/
↪passhport
```

En tant qu'utilisateur passhport (celui du system, pas celui de PostgreSQL), on initialise la base de données :

```
$ /home/passhport/passhport-run-env/bin/python /home/passhport/passhport/passhportd/
↪db_create.py
```

On lance alors *passhportd* (arrêtez le process s'il tourne) :

```
$ /home/passhport/passhport-run-env/bin/python /home/passhport/passhport/passhportd/
↪passhportd
```

PaSSHport utilise maintenant la base de données PostgreSQL.

1.2.4 Utiliser MySQL comme base de données

Installer le module python PyMySQL

Si vous n'avez pas utilisé la version paqueté de PaSSHport (dep/rpm), procédez comme suit. Si vous avez utilisé la version packagée, aller directement à la section *Configuration de MySQL*.

Si vous voulez utilisez MySQL comme base de données, vous devez installer le module python *PyMySQL*.

En tant qu'utilisateur passhport, on installe PyMySQL :

```
$ /home/passhport/passhport-run-env/bin/pip install PyMySQL
```

Configuration de MySQL

Créez une base de données *passhport* dans MySQL (la méthode peut varier selon votre distribution) :

```
# mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 2
Server version: 5.5.56-MariaDB MariaDB Server

Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE passhport;
Query OK, 1 row affected (0.00 sec)
```

Créez ensuite un utilisateur qui aura tous les droits sur la base de données *passhport* :

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON passhport.* TO passhport@localhost␣
↪IDENTIFIED BY 'iwetoh3oochieshaRei4';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> Bye

#
```

Configuration de passhportd

On modifie le fichier de configuration *passhportd.ini*(`/etc/passhport/passhportd.ini`). On change le paramètre `SQLALCHEMY_DATABASE_URI` :

```
SQLALCHEMY_DATABASE_URI         = mysql+pymysql://
↪passhport:iwetoh3oochieshaRei4@localhost/passhport
```

En tant qu'utilisateur passhport (celui du system, pas celui de MySQL), on initialise la base de données :

```
$ /home/passhport/passhport-run-env/bin/python /home/passhport/passhport/passhportd/
↪db_create.py
```

On relance passhportd (en tant que root) :

```
# systemctl restart passhportd
```

PaSSHport utilise désormais MySQL.

1.2.5 Ajouter les capacités de complétion de bash pour passhport-admin

Vous trouverez dans les sources un fichier *bash_completion*, qui permet d'utiliser la touche [TAB] pour auto-compléter les arguments de la commande *passhport-admin*.

En tant que root, copiez le fichier fourni (*/home/passhport/passhport/tools/passhport-admin.bash_completion*) dans le répertoire */etc/bash_completion.d/*, puis sourcez-le :

```
# cp /home/passhport/passhport/tools/passhport-admin.bash_completion /etc/bash_
↪completion.d/passhport-admin
# . /etc/bash_completion.d/passhport-admin
```

Vous pouvez maintenant avoir ce genre d'interaction :

```
# passhport-admin [TAB][TAB]
target user targetgroup usergroup
# passhport-admin t[TAB]
# passhport-admin target[TAB]
target targetgroup
# passhport-admin targetg[TAB]
# passhport-admin targetgroup [TAB][TAB]
list search show create edit adduser rmuser
addtarget rmtarget addusergroup rmusergroup
addtargetgroup rmtargetgroup delete


# passhport-admin user show [TAB][TAB]
john rachel alfred bruce kim jared
# passhport-admin user show j[TAB]
john jared
# passhport-admin user show ja[TAB]
# passhport-admin user show jared
```

Fin.

1.2.6 Ajouter un serveur WSGI devant PaSSHport

PaSSHport est basé sur Flask, le serveur embarqué ne peut gérer qu'une seule connexion à la fois. Il faut donc éviter de l'utiliser tel-quel en environnement de production...

Afin de gérer plus de requêtes, on duplique le nombre de processus PaSSHport pour chaque connexion. Apache permet de faire ça avec WSGI... Et c'est finalement assez simple à activer.

Installation

Su Debian :

```
apt install apache2 libapache2-mod-wsgi-py3
```

Configuration

Créez un nouveau vhost apache avec ce contenu :

```
Listen 5000
<VirtualHost *:5000>
    ServerName passhport

    SSLEngine               on
    SSLCertificateFile      /home/passhport/certs/cert.pem
    SSLCertificatekeyFile   /home/passhport/certs/key.pem

    WSGIDaemonProcess passhport user=passhport group=passhport threads=5
    WSGIScriptAlias / /home/passhport/passhport/tools/passhportd.wsgi
    <Directory /home/passhport/ >
        WSGIProcessGroup passhport
        WSGIApplicationGroup %{GLOBAL}
        # passhportd don't provides authentication, please filter by IP
        Require ip 127.0.0.1/8 ::1/128
    </Directory>
</VirtualHost>
```

Activez

Tout d'abord, tuez le processus passhportd existant

```
pkill passhportd
```

Désactivez le site par défaut, et activez celui-ci :

```
a2dissite 000-default
a2enmod ssl
a2ensite passhport.conf

systemctl restart apache2
```

et voilà.

1.2.7 Renouveler le certificat TLS de passhport

Quelques explications

Si vous avez installé PaSSHport il y a 1 an, vous rencontrez peut-être le message suivant lorsque vous essayez de vous connecter :

```
# ssh passhport@passhport.example.com
No such user in PaSSHport database.
tip: it can be a SSL certificate misconfiguration.
Connection to passhport.example.com closed
#
```

Ceci est généraleument du au fait que passhport (le script) n'arrive pas à se connecter à passhportd, et la cause la plus probable est l'expiration du certificat TLS généré lors de l'installation de PaSSHport.

```
passhport@passhport-srv:~$ openssl x509 -in /home/passhport/certs/cert.pem -noout -
↪text | grep Validity -A 2
      Validity
          Not Before: Sep 11 10:48:55 2020 GMT
          Not After : Sep 11 10:48:55 2021 GMT
passhport@passhport-srv:~$
```

Comme on peut le voir ci-dessus, le certificat n'est valable que pour 1 an. Ce dernier a été généré lors de l'installation de PaSSHport.

Renouvellement du certificat avec OpenSSL

Pour renouveler le certificat, utilisez la commande openssl comme suit :

```
root@passhport:~# openssl req -new -key "/home/passhport/certs/key.pem" \
-config "/home/passhport/passhport/tools/openssl-for-passhportd.cnf" \
-out "/home/passhport/certs/cert.pem" \
-subj "/C=FR/ST=Ile De France/L=Ivry sur Seine/O=LibrIT/OU=DSI/CN=passhport.librit.fr
↪" \
-x509 \
-days 365 \
-sha256 \
-extensions v3_req
root@passhport:~#
```

Ceci générera un certificat auto-signé, comme celui généré lors de l'installation. Il sera valide pour 1 an. Bien sur, vous pouvez changer ces valeurs selon vos besoins.

Redémarrer passhportd

Il suffit de redémarrer passhportd :

```
root@passhport:~# systemctl restart passhportd.service
root@passhport:~#
```

Vous devriez être désormais en mesure d'utiliser de nouveau PaSSHport.

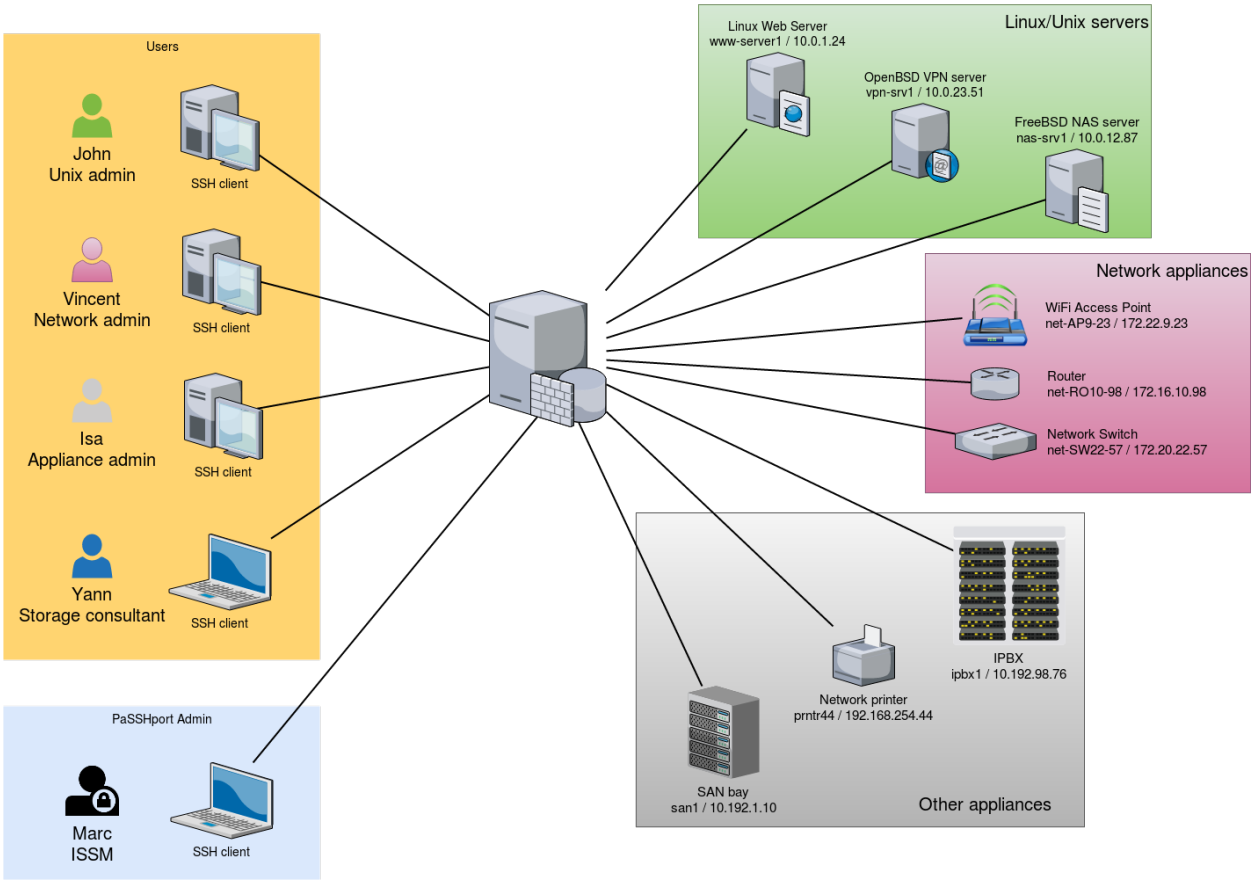
1.3 Premiers pas

Vous avez donc maintenant une toute nouvelle installation de PaSSHport, mais vous ne savez pas quoi faire ensuite…

1.3.1 Exemple d'architecture

Pour ce tutoriel, nous utiliserons les informations suivantes :

PaSSHport example architecture



— 1 serveur PaSSHport

Nous utiliserons une installation monolithique de PaSSHport : passhportd, passhport et passhport-admin sont sur la même machine.

3 utilisateurs

— John, un administrateur linux/unix, qui doit accéder à l'ensemble des serveurs linux/unix
— Vincent, un administrateur réseau, qui a besoin d'accéder à tous les périphériques réseaux
— Alice, une administratrice généraliste, qui a besoin d'accéder à tous les périphériques tiers
— Yann, un consultant qui est ici pour une mission temporaire sur l'infrastructure de stockage, qui a besoin d'accéder au server NAS et à la bas SAN

1 administrateur PaSSHport

— Marc, le RSSI, qui configure PaSSHport, et qui contrôle les droits d'accès

3 type de targets (serveur cible)

— Serveurs Linux/Unix :

— 1 serveur web, Linux, www-server / 10.0.1.24
— 1 serveur VPN, OpenBSD, vpn-srv1 / 10.0.23.51
— 1 serveur NAS, FreeBSD, nas-srv1 / 10.0.12.87

— Périphérique réseau

— 1 point d'accès WiFi, net-AP9-23 / 172.22.9.23
— 1 routeur, net-RO10-98 / 172.16.10.98
— 1 commutateur réseau (switch), net-SW22-57 / 172.20.22.57

— Autre périphériques

— 1 IPBX (serveur voix sur IP), ipbx1 / 10.192.98.76

— 1 imprimante réseau, prntr44 / 192.168.254.44
— 1 bay SAN, san1 / 10.192.1.10

1.3.2 Configurer les targets

Tout d'abord, nous allons enregistrer les targets dans PaSSHport

On se connecte au serveur PaSSHport, et on ajout le serveur linux. On peut le faire en tant qu'utilisateur passhport :

```
passhport@passhport-server:~$ passhport-admin target create www-server 10.0.1.24
OK: "www-server" -> created
passhport@passhport-server:~$
```

On peut vérifier que la target a été correctement enregistrée :

```
passhport@passhport-server:~$ passhport-admin target list
www-server
passhport@passhport-server:~$
```

Maintenant, ajoutons les autres serveurs Linux/Unix :

```
passhport@passhport-server:~$ passhport-admin target create vpn-srv1 10.0.23.51
OK: "vpn-srv1" -> created
passhport@passhport-server:~$ passhport-admin target create nas-srv1 10.0.23.51
OK: "nas-srv1" -> created
passhport@passhport-server:~$
```

On fait la même chose avec les périphériques réseaux, et les autres :

```
passhport@passhport-server:~$ passhport-admin target create net-AP9-23 172.22.9.23
OK: "net-AP9-23" -> created
passhport@passhport-server:~$ passhport-admin target create net-RO10-98 172.16.10.98
OK: "net-RO10-98" -> created
passhport@passhport-server:~$ passhport-admin target create net-SW22-57 172.20.22.57
OK: "net-SW22-57" -> created
passhport@passhport-server:~$ passhport-admin target create ipbx1 10.192.98.76
OK: "ipbx1" -> created
passhport@passhport-server:~$ passhport-admin target create prntr44 192.168.254.44
OK: "prntr44" -> created
passhport@passhport-server:~$ passhport-admin target create san1 10.192.1.10
OK: "san1" -> created
passhport@passhport-server:~$
```

Nous avons désormais l'ensemble de nos targets configurées dans PaSSHport.

1.3.3 Target avec un login spécifique

Nous voulons être capable de nous connecter à la baie SAN, en tant un autre utilisateur que root, car Yann (le prestataire) ne doit accéder à cette baie en tant que root, mais en tant qu'utilisateur "admin" :

```
root@passhport-server:~# passhport-admin target create
Name: admin@san1
Hostname: 10.192.1.10
Login (default is root): admin
Port: 22
```

(suite sur la page suivante)

(suite de la page précédente)

```
SSH Options:
Comment: SAN bay, login as admin user, not root.
OK: "admin@san1" -> created
root@passhport-server:~#
```

La baie SAN sera désormais accessible à travers deux targets : "san1" et "admin@san1".

1.3.4 Configuration des groupes de targets (targetgroup)

Nous groupons les targets que nous venons de créer dans trois groupes : unices, network, et autres.

Nous créons les groupes :

```
passhport@passhport-server:~$ passhport-admin targetgroup create unices
OK: "unices" -> created
passhport@passhport-server:~$ passhport-admin targetgroup create network
OK: "network" -> created
passhport@passhport-server:~$ passhport-admin targetgroup create others
OK: "others" -> created
passhport@passhport-server:~$
```

Maintenant nous mettons les targets dans les groupes correspondant :

```
passhport@passhport-server:~$ passhport-admin targetgroup addtarget www-server unices
OK: "www-server" added to "unices"
passhport@passhport-server:~$
```

Je suis paresseux, je vais donc "scripter" un peu tout ça :

```
passhport@passhport-server:~$ for UNICE in vpn-srv1 nas-srv1; do passhport-admin␣
↪targetgroup addtarget ${UNICE} unices; done
OK: "vpn-srv1" added to "unices"
OK: "nas-srv1" added to "unices"
passhport@passhport-server:~$ for NETAPPLIANCE in net-AP9-23 net-RO10-98 net-SW22-57;␣
↪do passhport-admin targetgroup addtarget ${NETAPPLIANCE} network; done
OK: "net-AP9-23" added to "network"
OK: "net-RO10-98" added to "network"
OK: "net-SW22-57" added to "network"
passhport@passhport-server:~$ for OTHERAPPLIANCE in ipbx1 prntr44 san1; do passhport-
↪admin targetgroup addtarget ${OTHERAPPLIANCE} others; done
OK: "ipbx1" added to "others"
OK: "prntr44" added to "others"
OK: "san1" added to "others"
passhport@passhport-server:~$
```

Nous créons un dernier groupe, qui contiendra l'ensemble des targets (je vais encore le scripter) :

```
passhport@passhport-server:~$ passhport-admin targetgroup create all-targets
OK: "all-targets" -> created
passhport@passhport-server:~$ for TARGET in `passhport-admin target list`; do␣
↪passhport-admin targetgroup addtarget ${TARGET} all-targets; done
OK: "ipbx1" added to "all-targets"
OK: "nas-srv1" added to "all-targets"
OK: "net-AP9-23" added to "all-targets"
OK: "net-RO10-98" added to "all-targets"
```

(suite sur la page suivante)

(suite de la page précédente)

```
OK: "net-SW22-57" added to "all-targets"
OK: "prntr44" added to "all-targets"
OK: "san1" added to "all-targets"
OK: "vpn-srv1" added to "all-targets"
OK: "www-server" added to "all-targets"
passhport@passhport-server:~$
```

On en a terminé avec les targets et les targetgroups (au moins pour le moment)...

1.3.5 Configuration des utilisateurs

Nous partons du préalable que nos utilisateurs on déjà crée leur clé publique SSH (rsa, dsa ou ecdsa), et qu'il nous on donné leur clé publique. Nous avons les clés suivantes :

Alice, une clé RSA de 2048 bits :

```
ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQC8JMsMgyRUEMoq31rPTIWpWKgGFQ7fxt5Kray8yzCPga2pohMLstjJeHpWjkVhH8FhRUv
↪V/42j3izeRH5liXFwotxzfpqTijTxAfj/
↪60IadcUSf5dE8WAiREarrV82ieU5eNZ4FoCH4W0xPS8pEYJDv6hQ8TFHYQCrwHloA3HgzEJgQSFWaS3niMDfNbgbJEOVhXuT2l7
↪tp+9FEAqGCH3kTuFhFnWCgguQxDxH4XiIj7n2w79ARPzMbn2vTtd+6N0or7 alice@myfirm.com
```

John, une clé ECDSA de 521 bits :

```
ecdsa-sha2-nistp521␣
↪AAAAE2VjZHNhLXNoYTItbmlzdHA1MjEAAAAIbmlzdHA1MjEAAACFBAHTlnhl23T9NiHn06wWaDpT1aJqEY0aOW7E4dfu7kQJsmH
↪yJYbKwwPQEaHpiQoHMaBfsgA2BYS5cNVcrOpLk8nHgKSJGEcdYipbZZxqDrLaeX3lBA== john@myfirm.
↪com
```

Marc, une clé RSA de 4096 bits :

```
ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAACAQDFOU5Saf+epkm79BeSniE7VtYMexJeL6BvXUsKUb7m8W4gnD3YTBW93uykO/
↪6ovi9TfYdm+4nKQ9gUGUgzNyD8o7zW8w6wKogoL24UbJKmkZOCU1IgHJSt1QYIs/qHQZ2MR6S6K2f/
↪1J1joYINPtGpQJ475OZfYQbP79fEdRdylupC8L+fvxkka4C0Uxj0I1VjDCVJCjO0md5oXzN75I2aw+RFWuiiL5P/
↪gHRu+2iff2rdhebJZs4ux8u76LQLzYsG9a85Xlagw6N7/aXWnUZ/9gqoF/
↪qVUHfS8ggesTwEJyNnY7EpPcKRUcwnlonn5CIS++Yo8iqjLd93RjFxShUqXlw9Cct4hdh/clW/
↪QYsJRMfN9860mZ9v9dEitM2X1w8HCCD5NAHGqRRrtONM99kZRxmkCQ/
↪Tb+jXvJ+VAl4qffuPPdxY+Bev7wygm4rVnjF2Ac5ioWb4Zd+zIb712VTQDQlRxsu73yWtHSodeSgPpgCWTjCwW/
↪841QbPGkclnE6DKIwQ/
↪vxC0ggSXouc5G6j0gHu90eQ24XL6Gurqr2C11w9saRyzrYRRlS0Ihkp3rMSteVcvrb1Qi4UGmJCHHSBhvP8jRFH4mbdkSGyzsxt
↪60fdEELQyX+kNFQ2VoCw== marc@myfirm.com
```

Vincent, une clé ECDSA de 521 bits :

```
ecdsa-sha2-nistp521␣
↪AAAAE2VjZHNhLXNoYTItbmlzdHA1MjEAAAAIbmlzdHA1MjEAAACFBAHJk+qDLEi283+rUmSek3eEF4PqXYMmQlPTj352w0XO75F
↪xVc+ypwOb2vv6pcjVsvuHTwHgXR2ElyfE8gGV7mITyXMdDyoWP5N8Ly3s7njNChSL9z3NiG38lg3E4Vg10nbmmoZZCA3WCffV4u
↪vincent@myfirm.com
```

Et Yann, une clé RSA de 2048 bits :

```
ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQCs9YpOfP9vgViYa1SSntrydEBLGyWGAr9nvEjqHcMwHQb9JEmhIjvk1ctb8+Kns3/
↪52F0hBrxic6k6UPvvvjbtJX33muFv5dd0k1W4lLcYe4ONTFwLOqCph4Is5r9lbZ5KXxhN/8YC/
↪08jBJow0CoYdc+Yr7MlA51+tEQFwPbuB5vHMUteye0IgmaH9MLzXes/j5BUhnBjDscWVQSvNHY4/
↪PKtHvIdvoI1uKAplstuHI6CDqnb0aJ5P9wME3P1lhRwcVDTm48/
↪AMcfmpp5s+DwOmyDGfGXf+hE0cu7ulAkwHBhR6ciJJg1pz4DqraglxyVyrt+PFq6KDeV/7WwoNEP␣
↪yann@otherfirm.com
```

(suite sur la page suivante)

(suite de la page précédente)

Avec ces clés, nous pouvons ajouter les utilisateurs, des manières suivantes…

— De manière interactive :

```
passphport@passhport-server:~$ passhport-admin user create
Email (user name): alice@myfirm.com
SSH Key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQC8JMsMgyRUEMoq31rPTIWpWKgGFQ7fxt5Kray8yzCPga2pohMLstjJeHpWjkVhH8FhRUv
↪V/42j3izeRH5liXFwotxzfpqTijTxAfj/
↪60IadcUSf5dE8WAiREarrV82ieU5eNZ4FoCH4W0xPS8pEYJDv6hQ8TFHYQCrwHloA3HgzEJgQSFWaS3niMDfNbgbJEOVhXuT2l7
↪tp+9FEAqGCH3kTuFhFnWCgguQxDxH4XiIj7n2w79ARPzMbn2vTtd+6N0or7 alice@myfirm.com
Comment: Alice is the general applicance admin
OK: "alice@myfirm.com" -> created
passhport@passhport-server:~$
```

— Sur une seule ligne, en un coup :

```
passhport@passhport-server:~$ passhport-admin user create john@myfirm.com "ecdsa-sha2-
↪nistp521␣
↪AAAAE2VjZHNhLXNoYTItbmlzdHA1MjEAAAAIbmlzdHA1MjEAAACFBAHTlnhl23T9NiHn06wWaDpT1aJqEY0aOW7E4dfu7kQJsmE
↪yJYbKwwPQEaHpiQoHMaBfsgA2BYS5cNVcrOpLk8nHgKSJGEcdYipbZZxqDrLaeX3lBA== john@myfirm.
↪com" --comment="John is the Unices admin. He rocks."
OK: "john@myfirm.com" -> created
passhport@passhport-server:~$
```

Nous ajoutons les autres :

```
passhport@passhport-server:~$ passhport-admin user create marc@myfirm.com "ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAACAQDFOU5Saf+epkm79BeSniE7VtYMexJeL6BvXUsKUb7m8W4gnD3YTBW93uykO/
↪6ovi9TfYdm+4nKQ9gUGUgzNyD8o7zW8w6wKogoL24UbJKmkZOCU1IgHJSt1QYIs/qHQZ2MR6S6K2f/
↪1J1joYINPtGpQJ475OZfYQbP79fEdRdylupC8L+fvxkka4C0Uxj0I1VjDCVJCjO0md5oXzN75I2aw+RFWuiiL5P/
↪gHRu+2iff2rdhebJZs4ux8u76LQLzYsG9a85Xlagw6N7/aXWnUZ/9gqoF/
↪qVUHfS8ggesTwEJyNnY7EpPcKRUcwnlonn5CIS++Yo8iqjLd93RjFxShUqXlw9Cct4hdh/clW/
↪QYsJRMfN9860mZ9v9dEitM2X1w8HCCD5NAHGqRRrtONM99kZRxmkCQ/
↪Tb+jXvJ+VAl4qffuPPdxY+Bev7wygm4rVnjF2Ac5ioWb4Zd+zIb712VTQDQlRxsu73yWtHSodeSgPpgCWTjCwW/
↪841QbPGkclnE6DKIwQ/
↪vxC0ggSXouc5G6j0gHu90eQ24XL6Gurqr2C11w9saRyzrYRRlS0Ihkp3rMSteVcvrb1Qi4UGmJCHHSBhvP8jRFH4mbdkSGyzsxt
↪60fdEELQyX+kNFQ2VoCw== marc@myfirm.com"
OK: "marc@myfirm.com" -> created
passhport@passhport-server:~$ passhport-admin user create vincent@myfirm.com "ecdsa-
↪sha2-nistp521␣
↪AAAAE2VjZHNhLXNoYTItbmlzdHA1MjEAAAAIbmlzdHA1MjEAAACFBAHJk+qDLEi283+rUmSek3eEF4PqXYMmQlPTj352w0XO75E
↪xVc+ypwOb2vv6pcjVsvuHTwHgXR2ElyfE8gGV7mITyXMdDyoWP5N8Ly3s7njNChSL9z3NiG38lg3E4Vg10nbmmoZZCA3WCffV4u
↪vincent@myfirm.com" --comment="Vincent is the network admin."
OK: "vincent@myfirm.com" -> created
passhport@passhport-server:~$ passhport-admin user create yann@otherfirm.com "ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQCs9YpOfP9vgViYa1SSntrydEBLGyWGAr9nvEjqHcMwHQb9JEmhIjvk1ctb8+Kns3/
↪52F0hBrxic6k6UPvvvjbtJX33muFv5dd0k1W4lLcYe4ONTFwLOqCph4Is5r9lbZ5KXxhN/8YC/
↪08jBJow0CoYdc+Yr7MlA51+tEQFwPbuB5vHMUteye0IgmaH9MLzXes/j5BUhnBjDscWVQSvNHY4/
↪PKtHvIdvoI1uKAplstuHI6CDqnb0aJ5P9wME3P1lhRwcVDTm48/
↪AMcfmpp5s+DwOmyDGfGXf+hE0cu7ulAkwHBhR6ciJJg1pz4DqraglxyVyrt+PFq6KDeV/7WwoNEP␣
↪yann@otherfirm.com" --comment="Yann is an external consultant, for a temporary␣
↪mission bout storage infrastructure."
OK: "yann@otherfirm.com" -> created
passhport@passhport-server:~$
```

Comme vous pouvez le voir ci-dessus, nous avons oublié de mettre un commentaire pour l'utilisateur "marc@myfirm.com". Ajoutons-en un :

```
passhport@passhport-server:~$ passhport-admin user edit marc@myfirm.com --newcomment=
↪"Marc is the ISSM. He access all."
OK: "marc@myfirm.com" -> edited
passhport@passhport-server:~$
```

Les utilisateurs sont maintenant créer. Mettons les dans des groupes d'utilisateur (usergroup)...

1.3.6 Configuration des usergroups

Même si dans ce tutoriel, nous n'avons qu'un utilisateur pour chaque type de machine à administrer, il est généralement conseillé de créer un groupe pour chaque type de compétence.

Ajoutons ces groupes :

```
passhport@passhport-server:~# passhport-admin usergroup create unices_admins
OK: "unices_admins" -> created
passhport@passhport-server:~# passhport-admin usergroup create network_admins
OK: "network_admins" -> created
passhport@passhport-server:~# passhport-admin usergroup create appliance_admins
OK: "appliance_admins" -> created
passhport@passhport-server:~# passhport-admin usergroup create super_admins
OK: "super_admins" -> created
passhport@passhport-server:~#
```

Nous ajoutons chaque utilisateur au groupe lui correspondant :

```
passhport@passhport-server:~$ passhport-admin usergroup adduser john@myfirm.com␣
↪unices_admins
OK: "john@myfirm.com" added to "unices_admins"
passhport@passhport-server:~$ passhport-admin usergroup adduser vincent@myfirm.com␣
↪network_admins
OK: "vincent@myfirm.com" added to "network_admins"
passhport@passhport-server:~$ passhport-admin usergroup adduser alice@myfirm.com␣
↪appliance_admins
OK: "alice@myfirm.com" added to "appliance_admins"
passhport@passhport-server:~$ passhport-admin usergroup adduser marc@myfirm.com super_
↪admins
OK: "marc@myfirm.com" added to "super_admins"
passhport@passhport-server:~$
```

1.3.7 Mise en ralation des groupe

Nous pouvons maintenant connecter chaque usergroup à son targetgroup :

```
passhport@passhport-server:~# passhport-admin targetgroup addusergroup unices_admins␣
↪unices
OK: "unices_admins" added to "unices"
passhport@passhport-server:~# passhport-admin targetgroup addusergroup network_admins␣
↪network
OK: "network_admins" added to "network"
passhport@passhport-server:~# passhport-admin targetgroup addusergroup appliance_
↪admins others
```

(suite sur la page suivante)

(suite de la page précédente)

```
OK: "appliance_admins" added to "others"
passhport@passhport-server:~# passhport-admin targetgroup addusergroup super_admins␣
↪all-targets
OK: "super_admins" added to "all-targets"
passhport@passhport-server:~#
```

1.3.8 Configuration spécifique pour le prestataire Yann :

Comme Yann n'est ici que pour une courte mission, et qu'il a besoin d'accéder à différentes targets qui n'ont pas vocations à être groupée au sein d'un targetgroup, nous connectons directement l'utilisateur à ses targets :

```
passhport@passhport-server:~$ passhport-admin target adduser yann@otherfirm.com nas-
↪srv1
OK: "yann@otherfirm.com" added to "nas-srv1"
passhport@passhport-server:~# passhport-admin target adduser yann@otherfirm.com␣
↪admin@san1
OK: "yann@otherfirm.com" added to "admin@san1"
passhport@passhport-server:~#
```

Comme vous pouvez le voir ci-dessus, nous n'avons pas donné à Yann l'accès a la baie SAN san1 en tant que root, mais en tant qu'utilisateur admin, grâce à la target admin@san1 précédemment configurée.

1.3.9 Vérification des droits d'accès :

On peut vérifier ce que nous avons configuré avec la sous-commande "show" de passhport-admin :

```
passhport@passhport-server:~$ passhport-admin user show marc@myfirm.com
Email: marc@myfirm.com
SSH key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAACAQDFOU5Saf+epkm79BeSniE7VtYMexJeL6BvXUsKUb7m8W4gnD3YTBW93uykO/
↪6ovi9TfYdm+4nKQ9gUGUgzNyD8o7zW8w6wKogoL24UbJKmkZOCU1IgHJSt1QYIs/qHQZ2MR6S6K2f/
↪1J1joYINPtGpQJ475OZfYQbP79fEdRdylupC8L+fvxkka4C0Uxj0I1VjDCVJCjO0md5oXzN75I2aw+RFWuiiL5P/
↪gHRu+2iff2rdhebJZs4ux8u76LQLzYsG9a85Xlagw6N7/aXWnUZ/9gqoF/
↪qVUHfS8ggesTwEJyNnY7EpPcKRUcwnlonn5CIS++Yo8iqjLd93RjFxShUqXlw9Cct4hdh/clW/
↪QYsJRMfN9860mZ9v9dEitM2X1w8HCCD5NAHGqRRrtONM99kZRxmkCQ/
↪Tb+jXvJ+VAl4qffuPPdxY+Bev7wygm4rVnjF2Ac5ioWb4Zd+zIb712VTQDQlRxsu73yWtHSodeSgPpgCWTjCwW/
↪841QbPGkclnE6DKIwQ/
↪vxC0ggSXouc5G6j0gHu90eQ24XL6Gurqr2C11w9saRyzrYRRlS0Ihkp3rMSteVcvrb1Qi4UGmJCHHSBhvP8jRFH4mbdkSGyzsxt
↪60fdEELQyX+kNFQ2VoCw== marc@myfirm.com
Comment: Marc is the ISSM. He access all.
Accessible target list: ipbx1 nas-srv1 net-AP9-23 net-RO10-98 net-SW22-57 prntr44␣
↪san1 vpn-srv1 www-server

Details in access:
Accessible directly:
Accessible through usergroups:
super_admins: www-server ; vpn-srv1 ; nas-srv1 ; net-AP9-23 ; net-RO10-98 ; net-SW22-
↪57 ; ipbx1 ; prntr44 ; san1 ;
Accessible through targetgroups:
passhport@passhport-server:~$
```

Comme on peut le voir, la sous commande "show" montre comment un utilisateur à accès à une target. Nous pouvons voir ci-dessus que Marc a accès à toutes targets configurées, parce que nous l'avons placé dans le usergroup "super_admins".

Voici l'exemple de Yann :

```
passphport@passhport-server:~$ passhport-admin user show yann@otherfirm.com
Email: yann@otherfirm.com
SSH key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQCs9YpOfP9vgViYa1SSntrydEBLGyWGAr9nvEjqHcMwHQb9JEmhIjvk1ctb8+Kns3/
↪52F0hBrxic6k6UPvvvjbtJX33muFv5dd0k1W4lLcYe4ONTFwLOqCph4Is5r9lbZ5KXxhN/8YC/
↪08jBJow0CoYdc+Yr7MlA51+tEQFwPbuB5vHMUteye0IgmaH9MLzXes/j5BUhnBjDscWVQSvNHY4/
↪PKtHvIdvoI1uKAplstuHI6CDqnb0aJ5P9wME3P1lhRwcVDTm48/
↪AMcfmpp5s+DwOmyDGfGXf+hE0cu7ulAkwHBhR6ciJJg1pz4DqraglxyVyrt+PFq6KDeV/7WwoNEP␣
↪yann@otherfirm.com
Comment: Yann is an external consultant, for a temporary mission bout storage␣
↪infrastructure.
Accessible target list: nas-srv1 san1

Details in access:
Accessible directly: nas-srv1 ; san1 ;
Accessible through usergroups:
Accessible through targetgroups:
passhport@passhport-server:~$
```

On peut voir ci-dessus que Yann à un accès direct aux target, sans passer par des usergroup, ou des targetgroup.

1.3.10 Connectons-nous !

Mettons nous maintenant dans la peau de John : je me connect à PaSSHport, en utilisant la clé id_ecdsa que j'ai envoyé à l'administrateur de PaSSHport :

```
john@my-desktop:~$ ssh passhport@passhport-server
Welcome john@myfirm.com.
Here is the list of servers you can access:
1  www-server  10.0.1.24
2  vpn-srv1    10.0.23.51
3  nas-srv1    10.0.12.87
Type the number, name or hostname of the server you want to connect to :
```

En tant que John, je constate que je peux accéder à trois serveurs : www-server, vpn-srv1, et nas-srv1. Je peux maintenant accéder à chacun de ces serveurs, selon les méthodes suivantes :

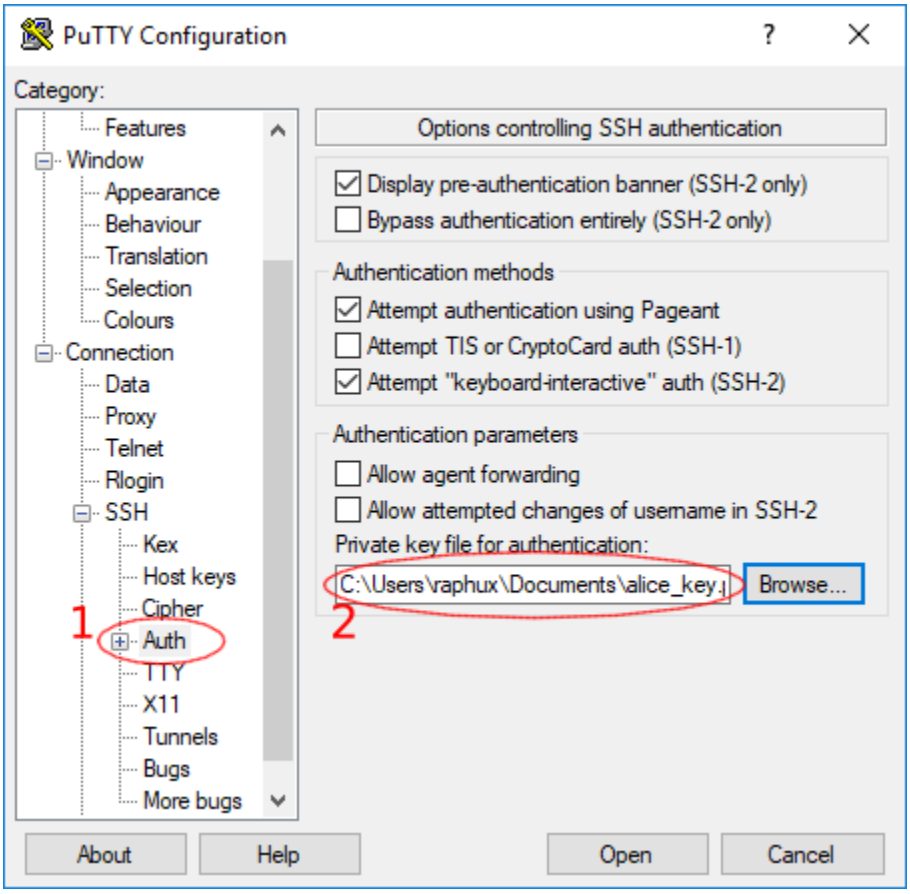
- Le numéro situé dans la première colonne ;
- le nom du serveur (www-server. . .) ;
- l'adresse IP.

```
john@my-desktop:~$ ssh passhport@passhport-server
Welcome john@myfirm.com.
Here is the list of servers you can access:
1  www-server  10.0.1.24
2  vpn-srv1    10.0.23.51
3  nas-srv1    10.0.12.87
Type the number, name or hostname of the server you want to connect to : 1
Linux www-server 4.9.0-3-amd64 #1 SMP Debian 4.9.30-2+deb9u3 (2017-08-06) x86_64
root@www-server:~#
```

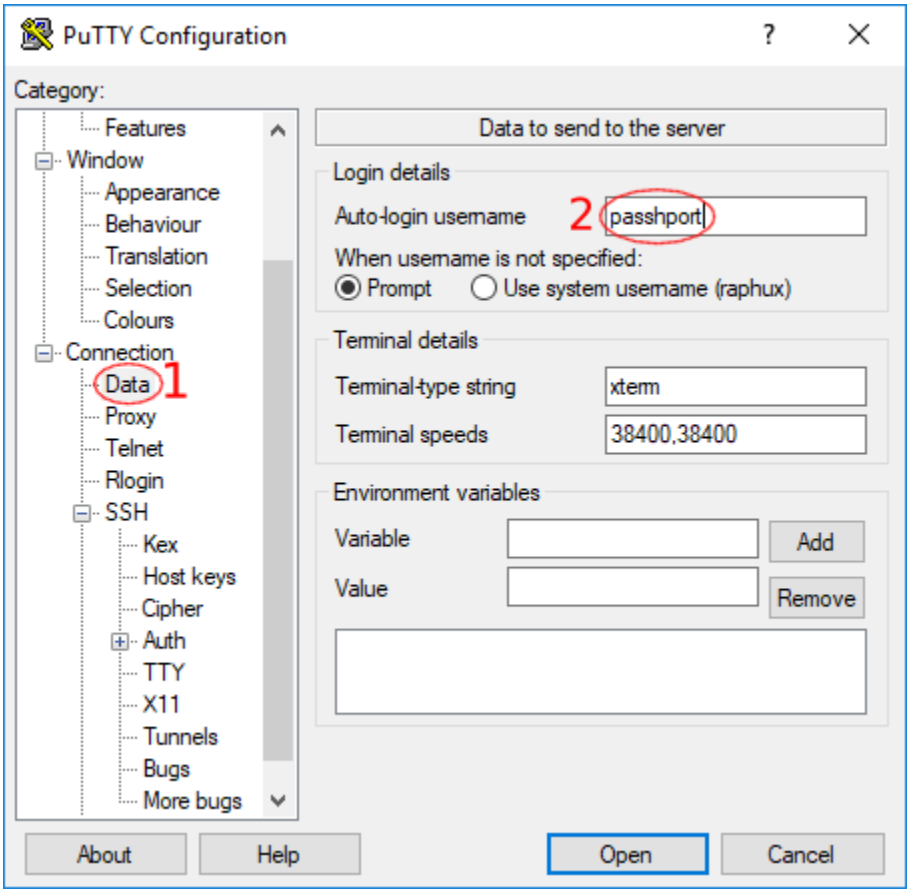
John est maintenant sur le serveur www-server.

Je suis maintenant dans la peau d'Alice, dont le poste de travail est sous Windows. Je vais utiliser l'outil Putty pour me connecter à PaSSHport. Voilà comment le configurer :

On lance Putty (téléchargable ici), et sur la partie gauche, on clique sur *Connection -> SSH -> Auth*, pour selectionner la clé PPK qu'Alice a créé (avec l'outil puttygen par exemple) :



Ensuite nous allons dans *Connection -> SSH -> Data* , et mettons le username à *passhport* :

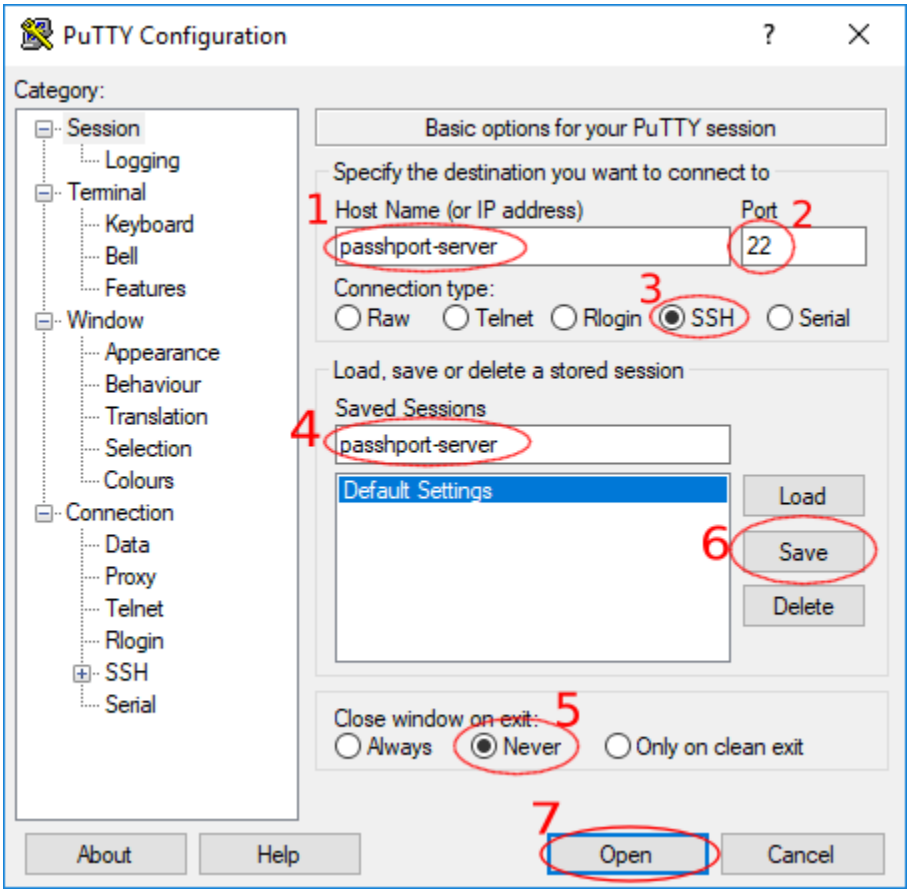


Pour finir, on retourne à la racine de l'arbre de configuration, dans la section *Session* :

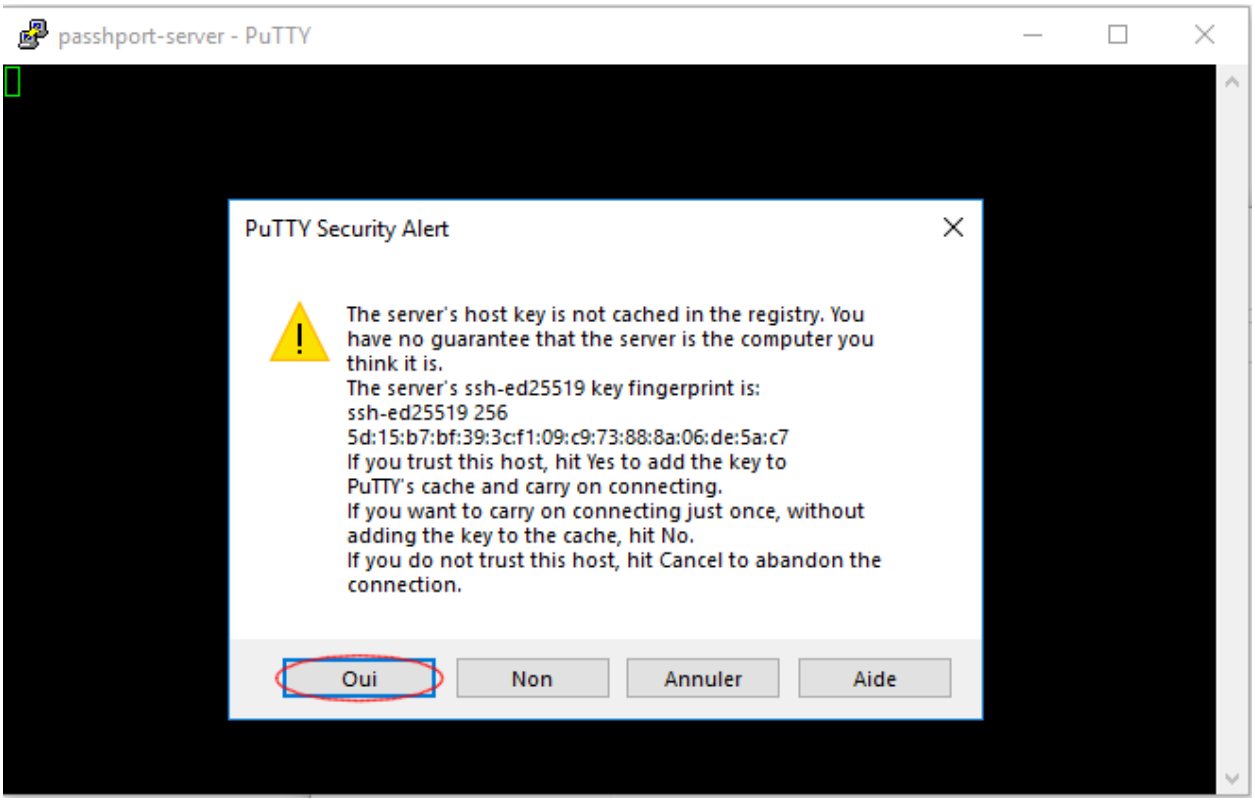
- on entre le nom d'hôte ou l'IP du serveur PaSSHport
- on entre le port du serveur SSH (généralement 22)
- on sélectionne *SSH* comme type de connexion
- on entre le nom sous lequel sauver la configuration que nous venons de mettre en place

Pour débugger une connexion défaillante, il est souvent utile de sélectionner l'option *Never* dans la section *Close window on exit*, et ce, afin de voir la raison de déconnexion.

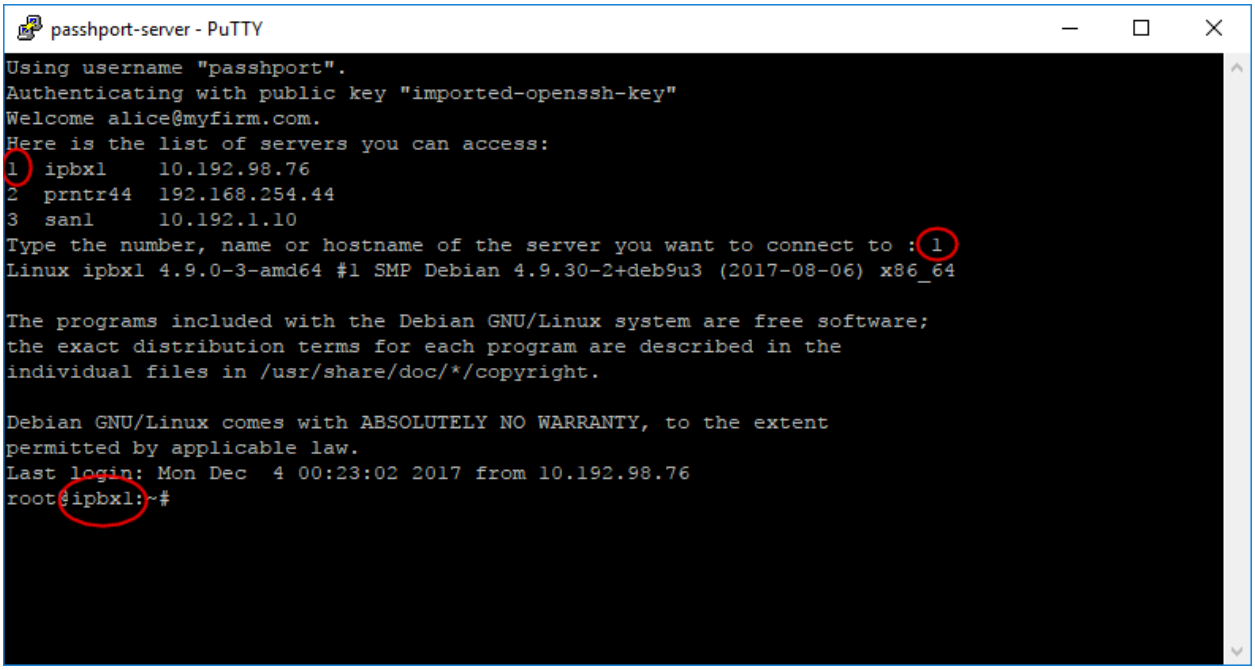
On sauve, et on clique le bouton *Open*, pour lancer la connexion !



Si c'est la première fois qu'on se connecte sur le serveur PaSSHport, une fenêtre indiquant qu'il faut vérifier l'empreinte du serveur, et si on doit l'accepter... On l'accepte en cliquant le bouton *Oui* :



Nous avons ensuite le prompt de PaSSHport, et comme nous voulons nous connecter à l'IPBX, on sélection l'entrée *1* :



On est désormais sur notre serveur cible.

Dernier exemple intéressant, Yann, qui accède au serveur nas-srv1 et san1. Il utilise un ordinateur portable sous linux :

```
yann@my-laptop:~$ ssh passhport@passhport-server
Welcome yann@otherfirm.com.
Here is the list of servers you can access:
1  nas-srv1    10.0.12.87
2  admin@san1  10.192.1.10  SAN bay, login as admin user, not root.
Type the number, name or hostname of the server you want to connect to :
```

Il peut désormais accéder au deux serveurs relatifs à sa mission.

1.3.11 Suppression d'un utilisateur

Yann a désormais fini sa mission, et a quitté la société. On peut révoquer ses droits de deux manières :

— en enlevant les liaisons de son compte (que ce soit une target, un targetgroup, ou un usergroup) ;
— en supprimant l'utilisateur.

Il parfois plus judicieux d'utiliser la première méthode, quand on sait par exemple que l'utilisateur est susceptible de revenir plus tard, et ne pas avoir à le recréer (et devoir récupérer sa clef, et ses informations...). Voici comment procéder :

Premièrement, on liste ses droits :

```
passhport@passhport-server:~$ passhport-admin user show yann@otherfirm.com
Email: yann@otherfirm.com
SSH key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQCs9YpOfP9vgViYa1SSntrydEBLGyWGAr9nvEjqHcMwHQb9JEmhIjvk1ctb8+Kns3/
↪52F0hBrxic6k6UPvvvjbtJX33muFv5dd0k1W4lLcYe4ONTFwLOqCph4Is5r9lbZ5KXxhN/8YC/
↪08jBJow0CoYdc+Yr7MlA51+tEQFwPbuB5vHMUteye0IgmaH9MLzXes/j5BUhnBjDscWVQSvNHY4/
↪PKtHvIdvoI1uKAplstuHI6CDqnb0aJ5P9wME3P1lhRwcVDTm48/
↪AMcfmpp5s+DwOmyDGfGXf+hE0cu7ulAkwHBhR6ciJJg1pz4DqraglxyVyrt+PFq6KDeV/7WwoNEP␣
↪yann@otherfirm.com
Comment: Yann is an external consultant, for a temporary mission bout storage␣
↪infrastructure.
Accessible target list: admin@san1 nas-srv1

Details in access:
Accessible directly: nas-srv1 ; admin@san1 ;
Accessible through usergroups:
Accessible through targetgroups:
passhport@passhport-server:~$
```

On peut voir qu'il a accès aux targets nas-srv1 et admin@san1, directement. Révoquons ces droits :

```
passhport@passhport-server:~$ passhport-admin target rmuser yann@otherfirm.com␣
↪admin@san1
OK: "yann@otherfirm.com" removed from "admin@san1"
passhport@passhport-server:~$ passhport-admin target rmuser yann@otherfirm.com nas-
↪srv1
OK: "yann@otherfirm.com" removed from "nas-srv1"
passhport@passhport-server:~$
```

Yann n'aura désormais plus accès à ces targets :

```
passhport@passhport-server:~$ passhport-admin user show yann@otherfirm.com
Email: yann@otherfirm.com
SSH key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQCs9YpOfP9vgViYa1SSntrydEBLGyWGAr9nvEjqHcMwHQb9JEmhIjvk1ctb8+Kns3/
↪52F0hBrxic6k6UPvvvjbtJX33muFv5dd0k1W4lLcYe4ONTFwLOqCph4Is5r9lbZ5KXxhN/8YC/
↪08jBJow0CoYdc+Yr7MlA51+tEQFwPbuB5vHMUteye0IgmaH9MLzXes/j5BUhnBjDscWVQSvNHY4/
↪PKtHvIdvoI1uKAplstuHI6CDqnb0aJ5P9wME3P1lhRwcVDTm48/
↪AMcfmpp5s+DwOmyDGfGXf+hE0cu7ulAkwHBhR6ciJJg1pz4DqraglxyVyrt+PFq6KDeV/7WwoNEP␣
↪yann@otherfirm.com
```

(suite sur la page suivante)

(suite de la page précédente)

```
Comment: Yann is an external consultant, for a temporary mission bout storage␣
↪infrastructure.
Accessible target list:

Details in access:
Accessible directly:
Accessible through usergroups:
Accessible through targetgroups:
passhport@passhport-server:~$
```

La seconde option est de supprimer cet utilisateur :

```
passhport@passhport-server:~$ passhport-admin user delete yann@otherfirm.com
Email: yann@otherfirm.com
SSH key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQCs9YpOfP9vgViYa1SSntrydEBLGyWGAr9nvEjqHcMwHQb9JEmhIjvk1ctb8+Kns3/
↪52F0hBrxic6k6UPvvvjbtJX33muFv5dd0k1W4lLcYe4ONTFwLOqCph4Is5r9lbZ5KXxhN/8YC/
↪08jBJow0CoYdc+Yr7MlA51+tEQFwPbuB5vHMUteye0IgmaH9MLzXes/j5BUhnBjDscWVQSvNHY4/
↪PKtHvIdvoI1uKAplstuHI6CDqnb0aJ5P9wME3P1lhRwcVDTm48/
↪AMcfmpp5s+DwOmyDGfGXf+hE0cu7ulAkwHBhR6ciJJg1pz4DqraglxyVyrt+PFq6KDeV/7WwoNEP␣
↪yann@otherfirm.com
Comment: Yann is an external consultant, for a temporary mission bout storage␣
↪infrastructure.
Accessible target list:

Details in access:
Accessible directly:
Accessible through usergroups:
Accessible through targetgroups:
Are you sure you want to delete yann@otherfirm.com? [y/N] y
OK: "yann@otherfirm.com" -> deleted
passhport@passhport-server:~$
```

1.3.12 Conclusion

Vous devriez être désormais capable d'utiliser les fonctions basiques de PaSSHport.

1.4 Utilisation de passhport-admin

Ce chapitre décrit l'utilisation de passhport-admin

Un chapitre par sous-module :

1.4.1 Utilisation de la CLI

Vous pouvez lancer passhport-admin en ligne de commande interactive, en utilisant l'option `-i`.

Vous pouvez configurer passhportd à travers ce mode interactif. Référez-vous à la bonne section de la documentation du module correspondant.

1.4.2 user

Usages

```
passhport-admin user list
passhport-admin user search [<pattern>]
passhport-admin user show [<name>]
passhport-admin user create [((<name> <sshkey>) [--comment=<comment>])]
passhport-admin user edit [(<name> [--newname=<name>] [--newsshkey=<sshkey>] [--
↪newcomment=<comment>])]
passhport-admin user delete [([-f | --force] <name>)]
```

list

passhport-admin target list affiche l'ensemble des users configurées.

Exemple :

```
admin@bastion:~$ passhport-admin user list
admin1@compagny.com
admin2@compagny.com
alice@compagny.com
bob@compagny.com
admin@bastion:~$
```

search

passhport-admin user search [<PATTERN>] cherche dans la liste des *user* un utilisateur dont le nom correspond au modèle <PATTERN>.

Exemple :

```
admin@bastion:~# passhport-admin user search admin
admin1@compagny.comi
admin2@compagny.com
admin@bastion:~#
```

Si aucun modèle (PATTERN), l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~# passhport-admin user search
Pattern: alice
alice@compagny.com
admin@bastion:~#
```

show

passhport-admin target show <*NAME*> affiche toutes les informations au sujet d'un user nommé <NAME>.

Exemple :

```
admin@bastion:~# passhport-admin user show alice@compagny.com
Email: alice@compagny.com
SSH key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAACAQDFOU5Saf+epkm79BeSniE7VtYMexJeL6BvXUsKUb7m8W4gnD3YTBW93uykO/
↪6ovi9TfYdm+4nKQ9gUGUgzNyD8o7zW8w6wKogoL24UbJKmkZOCU1IgHJSt1QYIs/qHQZ2MR6S6K2f/
↪1J1joYINPtGpQJ475OZfYQbP79fEdRdylupC8L+fvxkka4C0Uxj0I1VjDCVJCjO0md5oXzN75I2aw+RFWuiiL5P/
↪gHRu+2iff2rdhebJZs4ux8u76LQLzYsG9a85Xlagw6N7/aXWnUZ/9gqoF/
↪qVUHfS8ggesTwEJyNnY7EpPcKRUcwnlonn5CIS++Yo8iqjLd93RjFxShUqXlw9Cct4hdh/clW/
↪QYsJRMfN9860mZ9v9dEitM2X1w8HCCD5NAHGqRRrtONM99kZRxmkCQ/
↪Tb+jXvJ+VAl4qffuPPdxY+Bev7wygm4rVnjF2Ac5ioWb4Zd+zIb712VTQDQlRxsu73yWtHSodeSgPpgCWTjCwW/
↪841QbPGkclnE6DKIwQ/
↪vxC0ggSXouc5G6j0gHu90eQ24XL6Gurqr2C11w9saRyzrYRRlS0Ihkp3rMSteVcvrb1Qi4UGmJCHHSBhvP8jRFH4mbdkSGyzsxt
↪60fdEELQyX+kNFQ2VoCw== alice@compagny.com
Comment:
Accessible target list:

Details in access:
Accessible directly:
Accessible through usergroups:
Accessible through targetgroups:
admin@bastion:~#
```

Si aucun modèle (PATTERN), l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~# passhport-admin user show
Name: alice@compagny.com
Email: alice@compagny.com
SSH key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAACAQDFOU5Saf+epkm79BeSniE7VtYMexJeL6BvXUsKUb7m8W4gnD3YTBW93uykO/
↪6ovi9TfYdm+4nKQ9gUGUgzNyD8o7zW8w6wKogoL24UbJKmkZOCU1IgHJSt1QYIs/qHQZ2MR6S6K2f/
↪1J1joYINPtGpQJ475OZfYQbP79fEdRdylupC8L+fvxkka4C0Uxj0I1VjDCVJCjO0md5oXzN75I2aw+RFWuiiL5P/
↪gHRu+2iff2rdhebJZs4ux8u76LQLzYsG9a85Xlagw6N7/aXWnUZ/9gqoF/
↪qVUHfS8ggesTwEJyNnY7EpPcKRUcwnlonn5CIS++Yo8iqjLd93RjFxShUqXlw9Cct4hdh/clW/
↪QYsJRMfN9860mZ9v9dEitM2X1w8HCCD5NAHGqRRrtONM99kZRxmkCQ/
↪Tb+jXvJ+VAl4qffuPPdxY+Bev7wygm4rVnjF2Ac5ioWb4Zd+zIb712VTQDQlRxsu73yWtHSodeSgPpgCWTjCwW/
↪841QbPGkclnE6DKIwQ/
↪vxC0ggSXouc5G6j0gHu90eQ24XL6Gurqr2C11w9saRyzrYRRlS0Ihkp3rMSteVcvrb1Qi4UGmJCHHSBhvP8jRFH4mbdkSGyzsxt
↪60fdEELQyX+kNFQ2VoCw== alice@compagny.com
Comment:
Accessible target list:

Details in access:
Accessible directly:
Accessible through usergroups:
Accessible through targetgroups:
admin@bastion:~#
```

create

passhport-admin user create [((<name> <sshkey>) [–comment=<comment>])] crée un nouveau *user*

Argument	Description
<name>	Nom du user à éditer
<sshkey>	La clef SSH du *user* (bien utiliser des apostrophes pour entourer la clé)
–comment	Commentaire concernant le user (optionnel)

Exemple :

```
admin@bastion:~$ passhport-admin user create bob@compagny.com "ecdsa-sha2-nistp521␣
↪AAAAE2VjZHNhLXNoYTItbmlzdHA1MjEAAAAIbmlzdHA1MjEAAACFBAHTlnhl23T9NiHn06wWaDpT1aJqEY0aOW7E4dfu7kQJsmF
↪yJYbKwwPQEaHpiQoHMaBfsgA2BYS5cNVcrOpLk8nHgKSJGEcdYipbZZxqDrLaeX3lBA== bob@mydesktop"
OK: "bob@compagny.com" -> created
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin user create bob@compagny.com "ecdsa-sha2-nistp521␣
↪AAAAE2VjZHNhLXNoYTItbmlzdHA1MjEAAAAIbmlzdHA1MjEAAACFBAHTlnhl23T9NiHn06wWaDpT1aJqEY0aOW7E4dfu7kQJsmF
↪yJYbKwwPQEaHpiQoHMaBfsgA2BYS5cNVcrOpLk8nHgKSJGEcdYipbZZxqDrLaeX3lBA== bob@mydesktop"
OK: "bob@compagny.com" -> created
admin@bastion:~$ passhport-admin user create
Email (user name): john@ext-compagny.com
SSH Key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQCs9YpOfP9vgViYa1SSntrydEBLGyWGAr9nvEjqHcMwHQb9JEmhIjvk1ctb8+Kns3/
↪52F0hBrxic6k6UPvvvjbtJX33muFv5dd0k1W4lLcYe4ONTFwLOqCph4Is5r9lbZ5KXxhN/8YC/
↪08jBJow0CoYdc+Yr7MlA51+tEQFwPbuB5vHMUteye0IgmaH9MLzXes/j5BUhnBjDscWVQSvNHY4/
↪PKtHvIdvoI1uKAplstuHI6CDqnb0aJ5P9wME3P1lhRwcVDTm48/
↪AMcfmpp5s+DwOmyDGfGXf+hE0cu7ulAkwHBhR6ciJJg1pz4DqraglxyVyrt+PFq6KDeV/7WwoNEP␣
↪yann@mylaptop.com
Comment: John is a extern expert.
OK: "john@ext-compagny.com" -> created
admin@bastion:~$
```

edit

passhport-admin user edit [(<name> [–newname=<name>] [–newsshkey=<sshkey>] [–newcomment=<comment>])] édite un user existant.

Argument	Description
<name>	Nom du user à éditer
–newname	Nouveau nom du user que l'on souhaite renommer (optionnel)
–newsshkey	La nouvelle clef SSH du *user* (bien utiliser des apostrophes pour entourer la clé)
–newcomment	Nouveau commentaire concernant le user (optionel)

Exemple :

```
admin@bastion:~$ passhport-admin user edit john@ext-compagny.com --newname=john.
↪doe@ext-compagny.com --newcomment="John is a extern expert, he'll be here until␣
↪january 18th."
OK: "john@ext-compagny.com" -> edited
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif. Un tableau montrant l'ensemble des paramètre de configuration est d'abord affiché, puis, ligne par ligne, chaque argument modifiable est affiché. L'utilisateur peut

conserver chaque paramètre présenté au dessus en appuyant sur "Entrer". La seule exception est pour le champs "comment" : si l'utilisateur souhaite enlever le commentaire, il tape alors "Entrer", puis il lui sera demandé s'il veut supprimer le commentaire, ou non.

Exemple :

```
admin@bastion:~$ passhport-admin user edit
Name of the user you want to modify: john.doe@ext-compagny.com
Email: john.doe@ext-compagny.com
SSH key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQCs9YpOfP9vgViYa1SSntrydEBLGyWGAr9nvEjqHcMwHQb9JEmhIjvk1ctb8+Kns3/
↪52F0hBrxic6k6UPvvvjbtJX33muFv5dd0k1W4lLcYe4ONTFwLOqCph4Is5r9lbZ5KXxhN/8YC/
↪08jBJow0CoYdc+Yr7MlA51+tEQFwPbuB5vHMUteye0IgmaH9MLzXes/j5BUhnBjDscWVQSvNHY4/
↪PKtHvIdvoI1uKAplstuHI6CDqnb0aJ5P9wME3P1lhRwcVDTm48/
↪AMcfmpp5s+DwOmyDGfGXf+hE0cu7ulAkwHBhR6ciJJg1pz4DqraglxyVyrt+PFq6KDeV/7WwoNEP␣
↪yann@mylaptop.com
Comment: John is a extern expert, he'll be here until january 18th.
Accessible target list:

Details in access:
Accessible directly:
Accessible through usergroups:
Accessible through targetgroups:
New name:
New SSH key:
New comment: John is a extern expert, he'll be here until february 2nd
OK: "john.doe@ext-compagny.com" -> edited
admin@bastion:~$
```

Comme montré ci-dessus, seule l'entrée "New comment" a été modifiée. Si une entrée est simplement remplie par "Entrer", la valeur précédente est conservée.

delete

passhport-admin target delete [([-f | –force] <name>)] supprime un user.

Argument	Description
<name>	Nom du user à supprimer
-f ou –force	Si utilisé, aucune confirmation ne sera demandé à l'utilisateur

Exemple :

```
admin@bastion:~$ passhport-admin user delete john.doe@ext-compagny.com
Email: john.doe@ext-compagny.com
SSH key: ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAABAQCs9YpOfP9vgViYa1SSntrydEBLGyWGAr9nvEjqHcMwHQb9JEmhIjvk1ctb8+Kns3/
↪52F0hBrxic6k6UPvvvjbtJX33muFv5dd0k1W4lLcYe4ONTFwLOqCph4Is5r9lbZ5KXxhN/8YC/
↪08jBJow0CoYdc+Yr7MlA51+tEQFwPbuB5vHMUteye0IgmaH9MLzXes/j5BUhnBjDscWVQSvNHY4/
↪PKtHvIdvoI1uKAplstuHI6CDqnb0aJ5P9wME3P1lhRwcVDTm48/
↪AMcfmpp5s+DwOmyDGfGXf+hE0cu7ulAkwHBhR6ciJJg1pz4DqraglxyVyrt+PFq6KDeV/7WwoNEP␣
↪yann@mylaptop.com
Comment: John is a extern expert, he'll be here until february 2nd
Accessible target list:

Details in access:
Accessible directly:
```

(suite sur la page suivante)

(suite de la page précédente)

```
Accessible through usergroups:
Accessible through targetgroups:
Are you sure you want to delete john.doe@ext-compagny.com? [y/N] y
OK: "john.doe@ext-compagny.com" -> deleted
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin user delete
Name: bob@compagny.com
Email: bob@compagny.com
SSH key: ecdsa-sha2-nistp521␣
↪AAAAE2VjZHNhLXNoYTItbmlzdHA1MjEAAAAIbmlzdHA1MjEAAACFBAHTlnhl23T9NiHn06wWaDpT1aJqEY0aOW7E4dfu7kQJsmE
↪yJYbKwwPQEaHpiQoHMaBfsgA2BYS5cNVcrOpLk8nHgKSJGEcdYipbZZxqDrLaeX3lBA== bob@mydesktop
Comment:
Accessible target list:

Details in access:
Accessible directly:
Accessible through usergroups:
Accessible through targetgroups:
Are you sure you want to delete bob@compagny.com? [y/N] y
OK: "bob@compagny.com" -> deleted
admin@bastion:~$
```

1.4.3 target

Usages :

```
passhport-admin target list
passhport-admin target search [<pattern>]
passhport-admin target checkaccess [<pattern>]
passhport-admin target show [<name>]
passhport-admin target create [((<name> <hostname>) [--login=<login>] [--type=<ssh>]␣
↪[--comment=<comment>] [--sshoptions=<sshoptions>] [--port=<port>])]
passhport-admin target edit [(<name> [--newname=<name>] [--newhostname=<hostname>] [--
↪newlogin=<login>] [--newcomment=<comment>] [--newsshoptions=<sshoptions>] [--
↪newport=<port>])]
passhport-admin target (adduser | rmuser) [(<username> <targetname>)]
passhport-admin target (addusergroup | rmusergroup) [(<usergroupname> <targetname>)]
passhport-admin target delete [([-f | --force] <name>)]
```

list

passhport-admin target list affiche l'ensemble des targets configurées.

Exemple :

```
admin@bastion:~$ passhport-admin target list
srv1.compagny.com
srv2.compagny.com
srv3.compagny.com
```

(suite sur la page suivante)

(suite de la page précédente)

```
websrv.ext.client.com
webbackend.ext.client.com
admin@bastion:~$
```

search

passhport-admin target search [<PATTERN>] cherche dans la liste des targets celles qui correspondent au modèle (PATTERN) donné.

Exemple :

```
admin@bastion:~$ passhport-admin target search web
websrv.ext.client.com
webbackend.ext.client.com
admin@bastion:~$
```

Si aucun modèle (PATTERN), l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin target search
Pattern: web
websrv.ext.client.com
webbackend.ext.client.com
admin@bastion:~$
```

checkaccess

passhport-admin target checkaccess [<PATTERN>] verifie que PaSSHport a bien accès à toutes les targets qui correspondent au modèle (<PATTERN>) donné.

Exemple :

```
admin@bastion:~$ passhport-admin target checkaccess web
OK:    132.123.45.67   websrv.ext.client.com
OK:    132.234.56.78   webbackend.ext.client.com
admin@bastion:~$
```

Si aucun modèle (PATTERN), l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin target checkaccess
Pattern: web
OK:    132.123.45.67   websrv.ext.client.com
OK:    132.234.56.78   webbackend.ext.client.com
admin@bastion:~$
```

show

passhport-admin target show <NAME> affiche toutes les informations au sujet d'une target nommée <NAME>.

Exemple :

```
admin@bastion:~$ passhport-admin target show websrv.ext.client.com
Name: websrv.ext.client.com
Hostname: 132.123.45.67
Server Type : ssh
Login: root
Port: 22
SSH options:
Comment:
Attached users:
Usergroup list:
Users who can access this target: admin1@compagny.com admin2@compagny.com
All usergroups:
Member of the following targetgroups: all-targets
admin@bastion:~$
```

Si aucun modèle (PATTERN), l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin target show
Name: websrv.ext.client.com
Name: websrv.ext.client.com
Hostname: 132.123.45.67
Server Type : ssh
Login: root
Port: 22
SSH options:
Comment:
Attached users:
Usergroup list:
Users who can access this target: admin1@compagny.com admin2@compagny.com
All usergroups:
Member of the following targetgroups: all-targets
admin@bastion:~$
```

create

passhport-admin target create [((<name> <hostname>) [–login=<login>] [–type=<ssh>] [–comment=<comment>] [–sshoptions=<sshoptions>] [–port=<port>])] crée une nouvelle target.

Argument	Description
<name>	Nom de la target à supprimer
hostname	Nom d'hôte ou IP de la target
–login	Login à utiliser lors de la connexion à une target (optionnel)
–type	Le type de la target (pour la version entreprise seulement). Peut être *ssh*, *postgresql*, *mysql*, *oracle*. Cette option est utilisé pour savoir quel 'hook' lancer, en fonction justement du type. Si le type est autre chose que *ssh*, le serveur ne sera pas accessible via SSH. Si lla target est un serveur PostGreSQL, et que vous souhaitez lancer le 'hook' correspondant (généralement un proxy pour qui permet d'enregistrer tout ce qu'un utilisateur fait sur une base), utilisez le type *postgresql*. Même explication pour le type *mysql* et *oracle*. Laissez le type par défaut *ssh*, sauf si vous utilisez la version entreprise et que vous savez ce que vous faites.
–comment	Commentaire concernant la target (optionnel)
–sshoptions	Options SSH à utiliser pour se connecter à la target (optionnel)
–port	Port SSH à utiliser pour se connecter à la target (optionnel)

Exemple :

```
admin@bastion:~# passhport-admin target create firewall.compagny.com 87.65.43.219 --
↪login=root --comment="Client 1 web server number 1"
OK: "firewall.compagny.com" -> created
admin@bastion:~#
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~# passhport-admin target create
Name: firewall2.compagny.com
Hostname: 87.65.43.220
Type (default is ssh):
Login (default is root):
Port: 22
SSH Options:
Comment: Client 1 FireWall 2 (Cisco)
OK: "firewall1.compagny.com" -> created
admin@bastion:~#
```

Pensez à copier une clé public de passhport sur la target et à tester avec "checkaccess" les accès à votre nouvelle target.

edit

passhport-admin target edit [(<name> [–newname=<name>] [–newhostname=<hostname>] [–newtype=<ssh>] [–newlogin=<login>] [–newcomment=<comment>] [–newsshoptions=<sshoptions>] [–newport=<port>])] édite une target existante.

Argument	Description
<name>	Nom de la target à éditer
–newname	Nouveau nom de la target que l'on souhaite renommer (optionnel)
–newhostname	Nouveau nom d'hôte ou IP de la target (optionnel)
–newtype	Le type de la target (pour la version entreprise seulement). Peut être *ssh*, *postgresql*, *mysql*, *oracle*. Cette option est utilisé pour savoir quel 'hook' lancer, en fonction justement du type. Si le type est autre chose que *ssh*, le serveur ne sera pas accessible via SSH. Si lla target est un serveur PostGreSQL, et que vous souhaitez lancer le 'hook' correspondant (généralement un proxy pour qui permet d'enregistrer tout ce qu'un utilisateur fait sur une base), utilisez le type *postgresql*. Même explication pour le type *mysql* et *oracle*. Laissez le type par défaut *ssh*, sauf si vous utilisez la version entreprise et que vous savez ce que vous faites.
–newlogin	Nouveau login à utiliser lors de la connexion à une target (optionnel)
–newcomment	Nouveau commentaire concernant la target (optionel)
–newsshoptions	Nouvelles options SSH à utiliser pour se connecter à la target (optionnel)
–newport	Nouveau port SSH à utiliser pour se connecter à la target (optionnel)

Exemple :

```
admin@bastion:~# passhport-admin target edit firewall.compagny.com --
↪newname=firewall1.compagny.com --newcomment="Client 1 FireWall 1 (Cisco)" --
↪newlogin="admin"
OK: "firewall.compagny.com" -> edited
admin@bastion:~#
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif. Un tableau montrant l'ensemble des paramètre de configuration est d'abord affiché, puis, ligne par ligne, chaque argument modifiable est affiché. L'utilisateur peut conserver chaque paramètre présenté au dessus en appuyant sur "Entrer". La seule exception est pour le champs "comment" : si l'utilisateur souhaite enlever le commentaire, il tape alors "Entrer", puis il lui sera demandé s'il veut supprimer le commentaire, ou non.

Exemple :

```
admin@bastion:~# passhport-admin target edit
Name of the target you want to modify: firewall2.compagny.com
Name: firewall2.compagny.com
Hostname: 87.65.43.220
Server Type : ssh
Login: root
Port: 22
SSH options:
Comment: Client 1 FireWall 2 (Cisco)
Attached users:
Usergroup list:
Users who can access this target:
All usergroups:
Member of the following targetgroups:
New name:
```

(suite sur la page suivante)

(suite de la page précédente)

```
New hostname:
New Login: admin
New port:
New SSH options:
New comment:
Remove original comment? [y/N]N
OK: "firewall2.compagny.com" -> edited
admin@bastion:~#
```

Comme montré ci-dessus, seule l'entrée "New Login" a été modifié. Si une entrée est simplement rempli par "Entrer", la valeur précédent est conservée.

adduser

passhport-admin target adduser [(<username> <targetname>)] connecte directement une target avec un user.

Argument	Description
<username>	Nom du user que l'on connecte directement à la target
<targetname>	Nom de la target à laquelle on connecte directement au user

Exemple :

```
admin@bastion:~# passhport-admin target adduser admin1@compagny.com firewall1.
↪compagny.com
OK: "admin1@compagny.com" added to "firewall1.compagny.com"
admin@bastion:~#
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~# passhport-admin target adduser
Username: admin2@compagny.com
Targetname: firewall2.compagny.com
OK: "admin2@compagny.com" added to "firewall2.compagny.com"
admin@bastion:~#
```

rmuser

passhport-admin target adduser [(<username> <targetname>)] supprime le lien direct entre une target et un user.

Argument	Description
<username>	Nom du user que l'on délie de la target
<targetname>	Nom de la target de laquelle on délie le user

Exemple :

```
admin@bastion:~# passhport-admin target rmuser admin1@compagny.com firewall1.compagny.
↪com
OK: "admin1@compagny.com" removed from "firewall1.compagny.com"
admin@bastion:~#
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~# passhport-admin target rmuser
Username: admin2@compagny.com
Targetname: firewall2.compagny.com
OK: "admin2@compagny.com" removed from "firewall2.compagny.com"
admin@bastion:~#
```

addusergroup

passhport-admin target addusergroup [(<usergroupname> <targetname>)] connecte directement une target à un usergroup.

Argument	Description
<usergroupname>	Nom du usergroup à connecter directement à la target
<targetname>	Nom de la target à laquelle on connecte directement le usergroup

Exemple :

```
admin@bastion:~# passhport-admin target addusergroup firewall-admins firewall1.
↪compagny.com
OK: "firewall-admins" added to "firewall1.compagny.com"
admin@bastion:~#
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~# passhport-admin target addusergroup
Usergroupname: firewall-admins
Targetname: firewall2.compagny.com
OK: "firewall-admins" added to "firewall2.compagny.com"
admin@bastion:~#
```

rmusergroup

passhport-admin target delusergroup [(<usergroupname> <targetname>)] supprime le lien direct entre une target et un usergroup.

Argument	Description
<usergroupname>	Nom du usergroup à delier de la target
<targetname>	Nom de la target de laquelle on délie le usergroup

Exemple :

```
admin@bastion:~# passhport-admin target addusergroup firewall-admins firewall1.
↪compagny.com
OK: "firewall-admins" added to "firewall1.compagny.com"
admin@bastion:~#
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~# passhport-admin target addusergroup
Usergroupname: firewall-admins
Targetname: firewall2.compagny.com
OK: "firewall-admins" added to "firewall2.compagny.com"
admin@bastion:~#
```

delete

passhport-admin target delete [([-f | –force] <name>)] supprime une target.

Argument	Description
<name>	Nom de la target à supprimer
-f ou –force	Si utilisé, aucune confirmation ne sera demandé à l'utilisateur

Exemple :

```
admin@bastion:~# passhport-admin target delete firewall1.compagny.com
Name: firewall1.compagny.com
Hostname: firewall1.compagny.com
Server Type : ssh
Login: admin
Port: 22
SSH options:
Comment: Client 1 FireWall 1 (Cisco)
Attached users:
Usergroup list: firewall-admins
Users who can access this target:
All usergroups: firewall-admins
Member of the following targetgroups:
Are you sure you want to delete firewall1.compagny.com? [y/N] y
OK: "firewall1.compagny.com" -> deleted
admin@bastion:~#
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~# passhport-admin target delete
Name: firewall2.compagny.com
Name: firewall2.compagny.com
Hostname: 87.65.43.220
Server Type : ssh
Login: admin
Port: 22
SSH options:
Comment: Client 1 FireWall 2 (Cisco)
Attached users:
Usergroup list: firewall-admins network-admins
Users who can access this target:
All usergroups: firewall-admins network-admins
Member of the following targetgroups:
Are you sure you want to delete firewall2.compagny.com? [y/N] y
OK: "firewall2.compagny.com" -> deleted
admin@bastion:~#
```

1.4.4 usergroup

Usages :

```
passhport-admin usergroup list
passhport-admin usergroup search [<pattern>]
passhport-admin usergroup show [<name>]
passhport-admin usergroup create [(<name> [--comment=<comment>])]
passhport-admin usergroup edit [(<name> [--newname=<name>] [--newcomment=<comment>])]
passhport-admin usergroup (adduser | rmuser) [(<username> <usergroupname>)]
passhport-admin usergroup (addusergroup | rmusergroup) [(<subusergroupname>
↪<usergroupname>)]
passhport-admin usergroup delete [([-f | --force] <name>)]
```

list

passhport-admin usergroup list affiche l'ensemble des usergroups configurées.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup list
admins
database-admins
external
network-admins
admin@bastion:~$
```

search

passhport-admin usergroup search [<PATTERN>] cherche dans la liste des usergroup tous les usergroup dont le nom correspond à PATTERN.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup search ext
external
admin@bastion:~$
```

Si aucun modèle (PATTERN), l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup search
Pattern: admins
admins
database-admins
network-admins
admin@bastion:~$
```

show

passhport-admin usergroup show <NAME> affiche toutes les informations au sujet d'un usergroup nommé <NAME>.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup show admins
Name: admins
Comment:
User list: john@compagny.com vincent@compagny.com
Usergroup list:
All users: john@compagny.com vincent@compagny.com
All usergroups:
admin@bastion:~$
```

Si aucun modèle (PATTERN), l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup show
Name: admins
Name: admins
Comment:
User list: john@compagny.com vincent@compagny.com
Usergroup list:
All users: john@compagny.com vincent@compagny.com
All usergroups:
admin@bastion:~$
```

create

passhport-admin usergroup create [((<name> [–comment=<comment>])] crée un nouveau usergroup.

Argument	Description
<name>	Nom de la usergroup à créer
–comment	Commentaire concernant le usergroup (optionnel)

Exemple :

```
admin@bastion:~$ passhport-admin usergroup create external
OK: "external" -> created
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup delete database-admins
Name: database-admins
Comment:
User list:
Usergroup list:
All users:
All usergroups:
Are you sure you want to delete database-admins? [y/N] y
OK: "database-admins" -> deleted
admin@bastion:~$
```

edit

passhport-admin usergroup edit [(<name> [–newname=<name>] [–newcomment=<comment>])] édite un usergroup existant.

Argument	Description
<name>	Nom du usergroup à éditer
–newname	Nouveau nom du usergroup que l'on souhaite renommer (optionnel)
–newcomment	Nouveau commentaire concernant le usergroup (optionel)

Exemple :

```
admin@bastion:~$ passhport-admin usergroup edit admins --newname=linux-admins
OK: "admins" -> edited
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif. Un tableau montrant l'ensemble des paramètre de configuration est d'abord affiché, puis, ligne par ligne, chaque argument modifiable est affiché. L'utilisateur peut conserver chaque paramètre présenté au dessus en appuyant sur "Entrer". La seule exception est pour le champs "comment" : si l'utilisateur souhaite enlever le commentaire, il tape alors "Entrer", puis il lui sera demandé s'il veut supprimer le commentaire, ou non.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup edit
Name of the usergroup you want to modify: external
Name: external
Comment:
User list:
Usergroup list:
All users:
All usergroups:
New name: external-admins
New comment:
Remove original comment? [y/N]
OK: "external" -> edited
admin@bastion:~$
```

Comme montré ci-dessus, seule l'entrée "New name" a été modifiée. Si une entrée est simplement remplie par "Entrer", la valeur précédente est conservée.

adduser

passhport-admin usergroup adduser [(<username> <usergroupname>)] ajoute un user dans un usergroup.

Argument	Description
<username>	Nom du user ajouter dans un usergroup
<usergroupname>	Nom du usergroup dans lequel on ajoute l'utilisateur

Exemple :

```
admin@bastion:~$ passhport-admin usergroup adduser vincent@compagny.com network-admins
OK: "vincent@compagny.com" added to "network-admins"
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup adduser
Username: yann@ext-compagny.com
Usergroupname: external-admins
OK: "yann@ext-compagny.com" added to "external-admins"
admin@bastion:~$
```

rmuser

passhport-admin usergroup rmuser [(<username> <usergroupname>)] enlève un user d'un usergroup.

Argument	Description
<username>	Nom du user que l'on souhaite enlever d'un usergroup
<usergroupname>	Nom du usergroup duquel on souhaite enlever le user

Exemple :

```
admin@bastion:~$ passhport-admin usergroup rmuser vincent@compagny.com linux-admins
OK: "vincent@compagny.com" removed from "linux-admins"
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup rmuser
Username: yann@ext-compagny.com
Usergroupname: external-admins
OK: "yann@ext-compagny.com" removed from "external-admins"
admin@bastion:~$
```

addusergroup

passhport-admin usergroup addusergroup [(<subusergroupname> <usergroupname>)] ajoute un usergroup dans un autre usergroup.

Argument	Description
<subusergroupname>	Nom du usergroup à ajouter dans un autre usergroup
<usergroupname>	Nom du usergroup dans lequel on ajoute l'autre usergroup

Exemple :

```
admin@bastion:~$ passhport-admin usergroup addusergroup linux-admins admins
OK: "linux-admins" added to "admins"
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup addusergroup
Subusergroupname: network-admins
Usergroupname: admins
OK: "network-admins" added to "admins"
admin@bastion:~$
```

rmusergroup

passhport-admin usergroup delusergroup [(<usergroupname> <usergroupname>)] supprime un usergroup d'un autre usergroup.

Argument	Description
<subusergroupname>	Nom du usergroup que l'on souhaite enlever d'un autre usergroup
<usergroupname>	Nom du usergroup duquel on souhaite enlever l'autre usergroup

Exemple :

```
admin@bastion:~$ passhport-admin usergroup rmusergroup linux-admins admins
OK: "linux-admins" removed from "admins"
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup rmusergroup
Subsergroupname: network-admins
Usergroupname: admins
OK: "network-admins" removed from "admins"
admin@bastion:~$
```

delete

passhport-admin usergroup delete [([-f | –force] <name>)] supprime un usergroup.

Argument	Description
<name>	Nom du usergroup à supprimer
-f ou –force	Si utilisé, aucune confirmation ne sera demandé à l'utilisateur

Exemple :

```
admin@bastion:~$ passhport-admin usergroup delete network-admins
Name: network-admins
Comment:
User list: vincent@compagny.com
Usergroup list:
All users: vincent@compagny.com
All usergroups:
Are you sure you want to delete network-admins? [y/N] y
OK: "network-admins" -> deleted
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin usergroup delete
Name: linux-admins
Name: linux-admins
Comment:
User list: john@compagny.com
Usergroup list:
All users: john@compagny.com
All usergroups:
Are you sure you want to delete linux-admins? [y/N] y
OK: "network-admins" -> deleted
admin@bastion:~$
```

1.4.5 targetgroup

Usages :

```
passhport-admin targetgroup list
passhport-admin targetgroup search [<pattern>]
passhport-admin targetgroup show [<name>]
passhport-admin targetgroup create [(<name> [--comment=<comment>])]
passhport-admin targetgroup edit [(<name> [--newname=<name>] [--newcomment=<comment>
↪])]
passhport-admin targetgroup (adduser | rmuser) [(<username> <targetgroupname>)]
passhport-admin targetgroup (addtarget | rmtarget) [(<targetname> <targetgroupname>)]
passhport-admin targetgroup (addusergroup | rmusergroup) [(<usergroupname>
↪<targetgroupname>)]
passhport-admin targetgroup (addtargetgroup | rmtargetgroup) [(<subtargetgroupname>
↪<targetgroupname>)]
passhport-admin targetgroup delete [([-f | --force] <name>)]
```

list

passhport-admin targetgroup list affiche l'ensemble des targetgroups configurées.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup list
linux-servers
network-appliances
phone-appliance
admin@bastion:~$
```

search

passhport-admin targetgroup search [<PATTERN>] cherche dans la liste des targetgroups, les targetgroups qui correspondent au modèle <PATTERN>.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup search appliance
network-appliances
phone-appliance
admin@bastion:~$
```

Si aucun modèle (PATTERN), l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup search
Pattern: servers
linux-servers
admin@bastion:~$
```

show

passhport-admin targetgroup show <NAME> affiche toutes les informations au sujet d'un targetgroup nommé <NAME>.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup show linux-servers
Name: linux-servers
Comment:
User list:
Target list: linux-7892 linux-7239 linux-1398
Usergroup list:
Targetgroup list:
All users:
All targets: linux-1398 linux-7239 linux-7892
All usergroups:
All targetgroups:
admin@bastion:~$
```

Si aucun modèle (PATTERN), l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup show
Name: linux-servers
Name: linux-servers
Comment:
User list:
Target list: linux-7892 linux-7239 linux-1398
Usergroup list:
Targetgroup list:
All users:
All targets: linux-1398 linux-7239 linux-7892
All usergroups:
All targetgroups:
admin@bastion:~$
```

create

passhport-admin targetgroup create [((<name> [–comment=<comment>])] crée un nouveau targetgroup.

Argument	Description
<name>	Nom de la targetgroup à créer
–comment	Commentaire concernant le targetgroup (optionnel)

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup create linux-servers
OK: "linux-servers" -> created
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup create
Name: phone-appliance
Comment: Phones and IPBX appliances group.
OK: "phone-appliance" -> created
admin@bastion:~$
```

edit

passhport-admin targetgroup edit [(<name> [–newname=<name>] [–newcomment=<comment>])] édite un targetgroup existant.

Argument	Description
<name>	Nom du targetgroup à éditer
–newname	Nouveau nom de la targetgroup (optionnel)
–newcomment	Nouveau commentaire concernant le targetgroup (optionnel)

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup edit linux-servers --newcomment="Linux␣
↪servers group."
OK: "linux-servers" -> edited
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif. Un tableau montrant l'ensemble des paramètre de configuration est d'abord affiché, puis, ligne par ligne, chaque argument modifiable est affiché. L'utilisateur peut conserver chaque paramètre présenté au dessus en appuyant sur "Entrer". La seule exception est pour le champs "comment" : si l'utilisateur souhaite enlever le commentaire, il tape alors "Entrer", puis il lui sera demandé s'il veut supprimer le commentaire, ou non.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup edit
Name of the targetgroup you want to modify: network-appliances
Name: network-appliances
Comment:
User list:
Target list:
Usergroup list:
Targetgroup list:
All users:
```

(suite sur la page suivante)

(suite de la page précédente)

```
All targets:
All usergroups:
All targetgroups:
New name:
New comment: Network appliance group.
OK: "network-appliances" -> edited
admin@bastion:~$
```

Comme montré ci-dessus, seule l'entrée "New comment" a été modifié. Si une entrée est simplement rempli par "Entrer", la valeur précédent est conservée.

adduser

passhport-admin targetgroup adduser [(<username> <targetname>)] connecte un user directement à un targetgroup.

Argument	Description
<username>	Nom du user que l'on connecte directement au targetgroup
<targetname>	Nom du targetgroup auquel on connecte directement un user

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup adduser john@compagny.com linux-servers
OK: "john@compagny.com" added to "linux-servers"
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup adduser
Username: vincent@compagny.com
Targetgroupname: network-appliances
OK: "vincent@compagny.com" added to "network-appliances"
admin@bastion:~$
```

rmuser

passhport-admin targetgroup rmuser [(<username> <targetname>)] supprime le lien direct entre un targetgroup et un user.

Argument	Description
<username>	Nom du user que l'on connecte directement au targetgroup
<targetname>	Nom du targetgroup duquel on déconnecte un user

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup rmuser vincent@compagny.com network-
↪appliances
OK: "vincent@compagny.com" removed from "network-appliances"
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup rmuser
Username: john@compagny.com
Targetgroupname: linux-servers
OK: "john@compagny.com" removed from "linux-servers"
admin@bastion:~$
```

addusergroup

passhport-admin targetgroup addusergroup [(<usergroupname> <targetname>)] connecte directement un targetgroup à un usergroup.

Argument	Description
<usergroupname>	Nom du usergroup à connecter directement à un targetgroup
<targetname>	Nom du targetgroup auquel on connecte directement le usergroup

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup addusergroup linux-admins linux-servers
OK: "linux-admins" added to "linux-servers"
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup addusergroup
Usergroupname: network-admins
Targetgroupname: network-appliances
OK: "network-admins" added to "network-appliances"
admin@bastion:~$
```

rmusergroup

passhport-admin targetgroup delusergroup [(<usergroupname> <targetgroupname>)] supprime le lien direct entre un targetgroup et un usergroup.

Argument	Description
<usergroupname>	Nom du usergroup que l'on souhaite déconnecter du targetgroup
<targetname>	Nom du targetgroup duquel on déconnecte un usergroup

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup rmusergroup linux-admins linux-servers
OK: "linux-admins" removed from "linux-servers"
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup rmusergroup
Usergroupname: network-admins
Targetgroupname: network-appliances
OK: "network-admins" removed from "network-appliances"
admin@bastion:~$
```

addtargetgroup

passhport-admin targetgroup addusergroup [(<subusergroupname> <targetgroupname>)] connecte directement un targetgroup à un autre targetgroup.

Argument	Description
<subtargetgroupname>	Nom du targetgroup à connecter directement à un autre targetgroup
<targetname>	Nom du targetgroup à connecter directement à un autre targetgroup

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup addtargetgroup linux-servers all-servers
OK: "linux-servers" added to "all-servers"
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup addtargetgroup
Subtargetgroupname: network-appliances
Targetgroupname: all-servers
OK: "network-appliances" added to "all-servers"
admin@bastion:~$
```

rmtargetgroup

passhport-admin targetgroup deltargetgroup [(<subtargetgroupname> <targetgroupname>)] supprime le lien direct entre un targetgroup et un autre targetgroup.

Argument	Description
<subtargetgroupname>	Nom du targetgroup que l'on souhaite déconnecter d'un autre targetgroup
<targetname>	Nom du targetgroup duquel on déconnecte l'autre targetgroup

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup rmtargetgroup linux-servers all-servers
OK: "linux-servers" removed from "all-servers"
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup rmtargetgroup
Subtargetgroupname: network-appliances
Targetgroupname: all-servers
OK: "network-appliances" removed from "all-servers"
admin@bastion:~$
```

delete

passhport-admin targetgroup delete [([-f | –force] <name>)] supprime une target.

Argument	Description
<name>	Nom du targetgroup à supprimer
-f ou –force	Si utilisé, aucune confirmation ne sera demandé à l'utilisateur

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup delete linux-servers
Name: linux-servers
Comment: Linux servers group.
User list:
Target list: linux-7892 linux-7239 linux-1398
Usergroup list:
Targetgroup list:
All users:
All targets: linux-1398 linux-7239 linux-7892
All usergroups:
All targetgroups:
Are you sure you want to delete linux-servers? [y/N] y
OK: "linux-servers" -> deleted
admin@bastion:~$
```

Si aucun argument n'est donné, l'utilisateur entre en mode interactif.

Exemple :

```
admin@bastion:~$ passhport-admin targetgroup delete
Name: network-appliances
Name: network-appliances
Comment: Network appliance group.
User list:
Target list:
Usergroup list:
Targetgroup list:
All users:
All targets:
All usergroups:
All targetgroups:
Are you sure you want to delete network-appliances? [y/N] y
OK: "linux-servers" -> deleted
admin@bastion:~$
```

1.5 Coté utilsateur

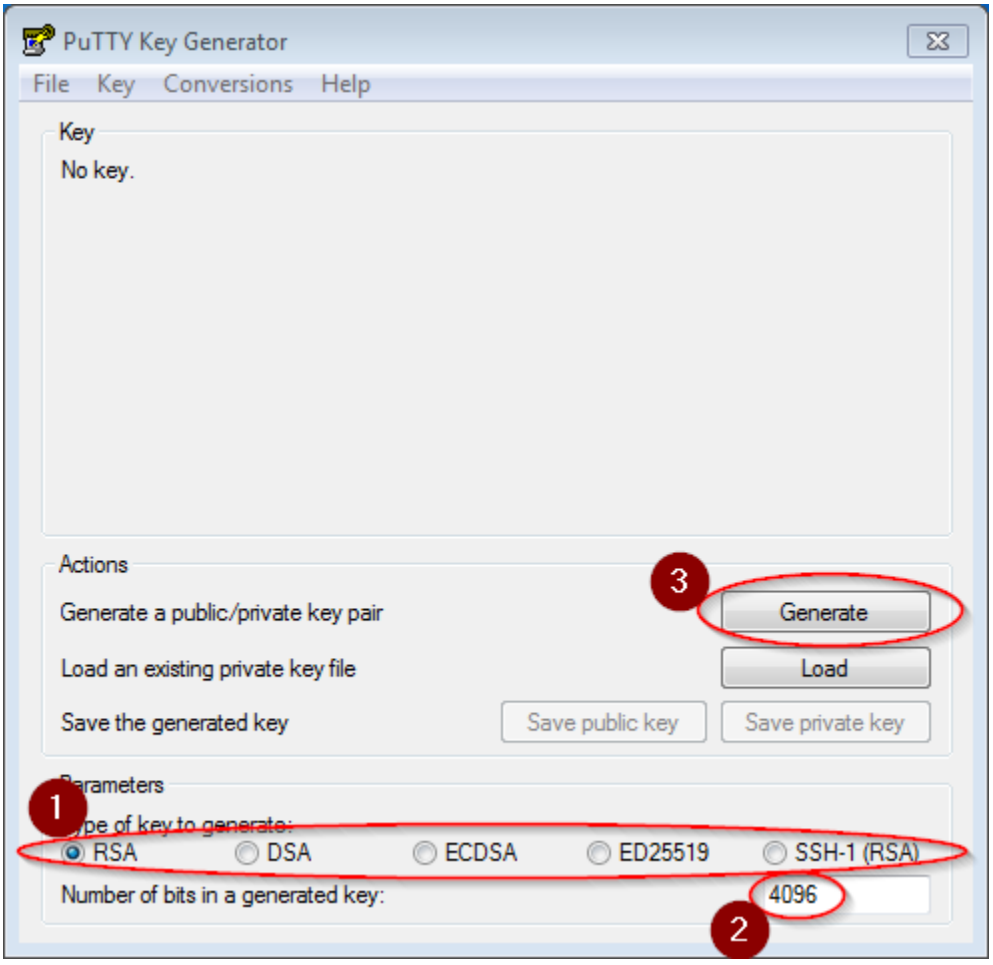
Ce chapitre montre comment se connecter à une target, à travers PaSSHport, depuis la génération d'une paire de clé, jusqu'à SCP.

1.5.1 Générer des clés privées

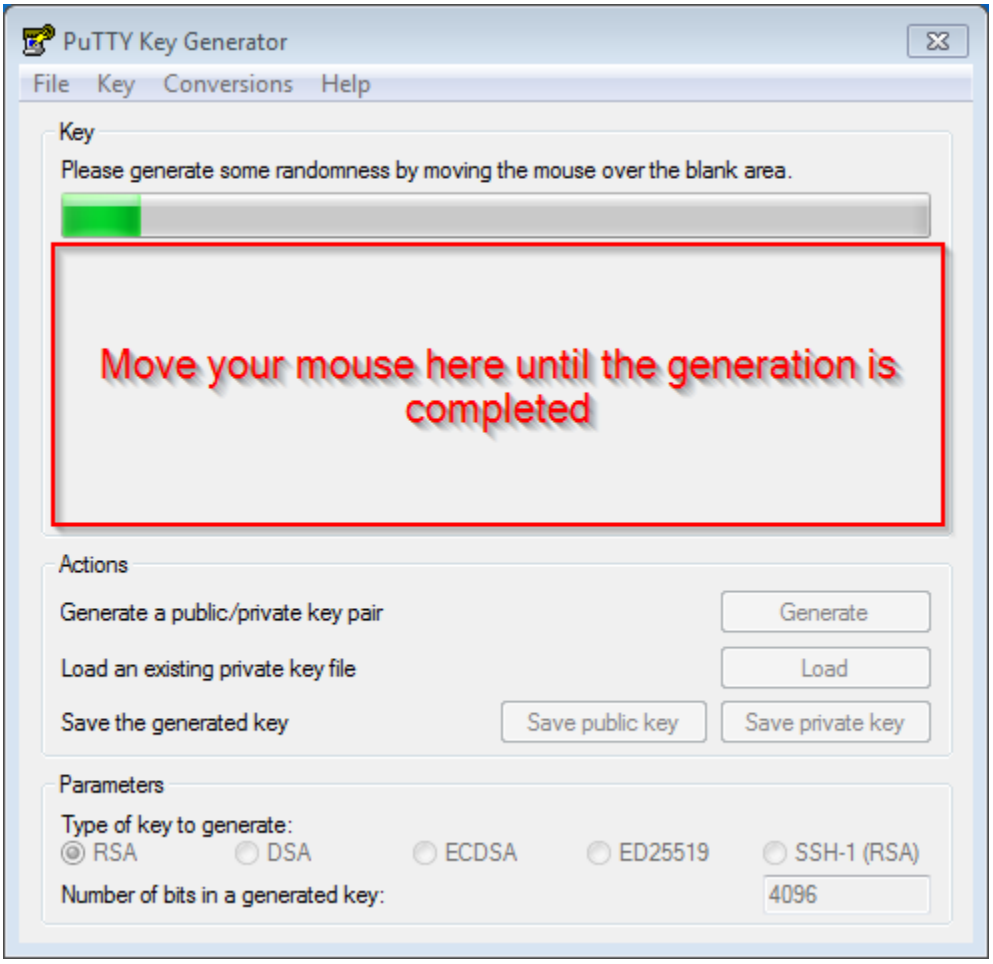
Sur Windows :

Pour générer la clé publique (extraite de la clé privée créée) que vous transmettrez à votre administrateur qui s'occupe de PaSSHport, utilisez *puttygen*, qui vous pouvez télécharger ici (cherchez *puttygen.exe* sur la page).

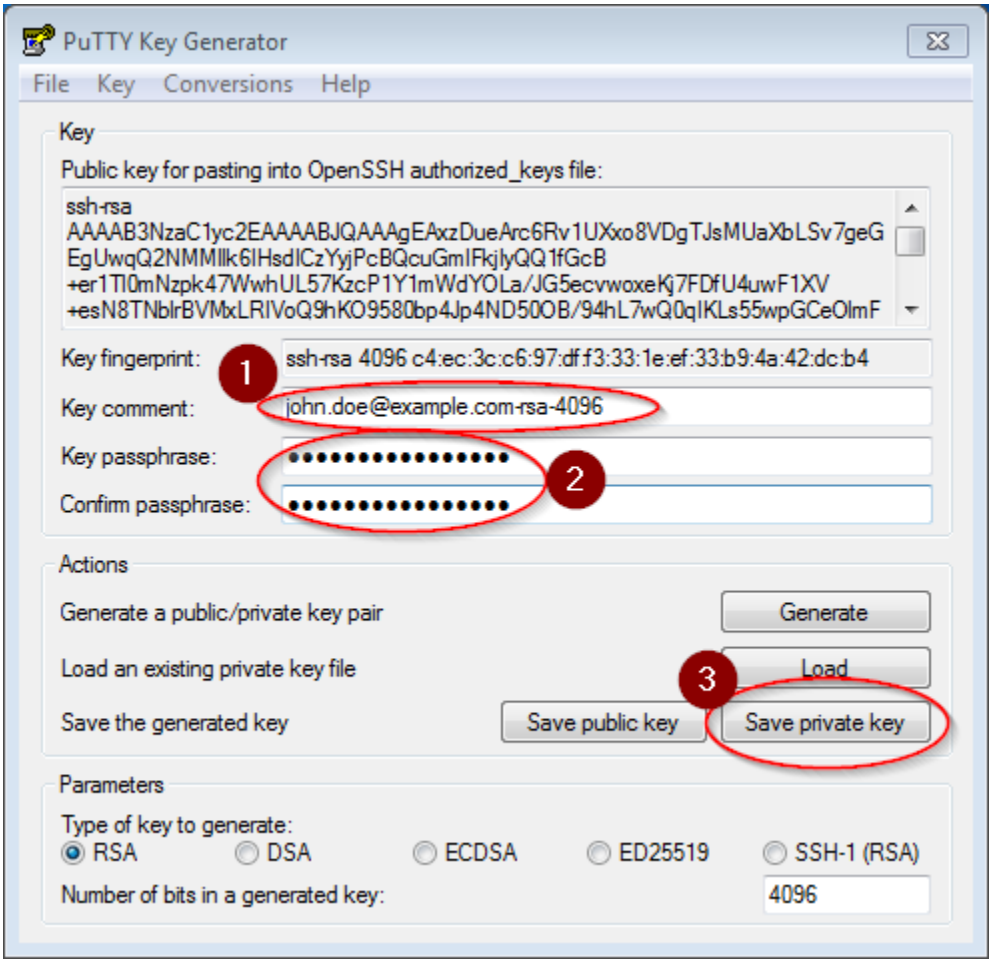
Démarrez puttygen, et sur la fenêtre principale, sélectionnez le type de clé que vous voulez générer (*1*), la longueur de la clé (*2*), et cliquez ensuite sur le bouton *Generate* (*3*). On a sélectionné dans cette exemple que clé de type RSA avec une taille de 4096 bits (as of 2019, une longueur de 2048 bits est considéré comme un minimum pour une clé RSA) :



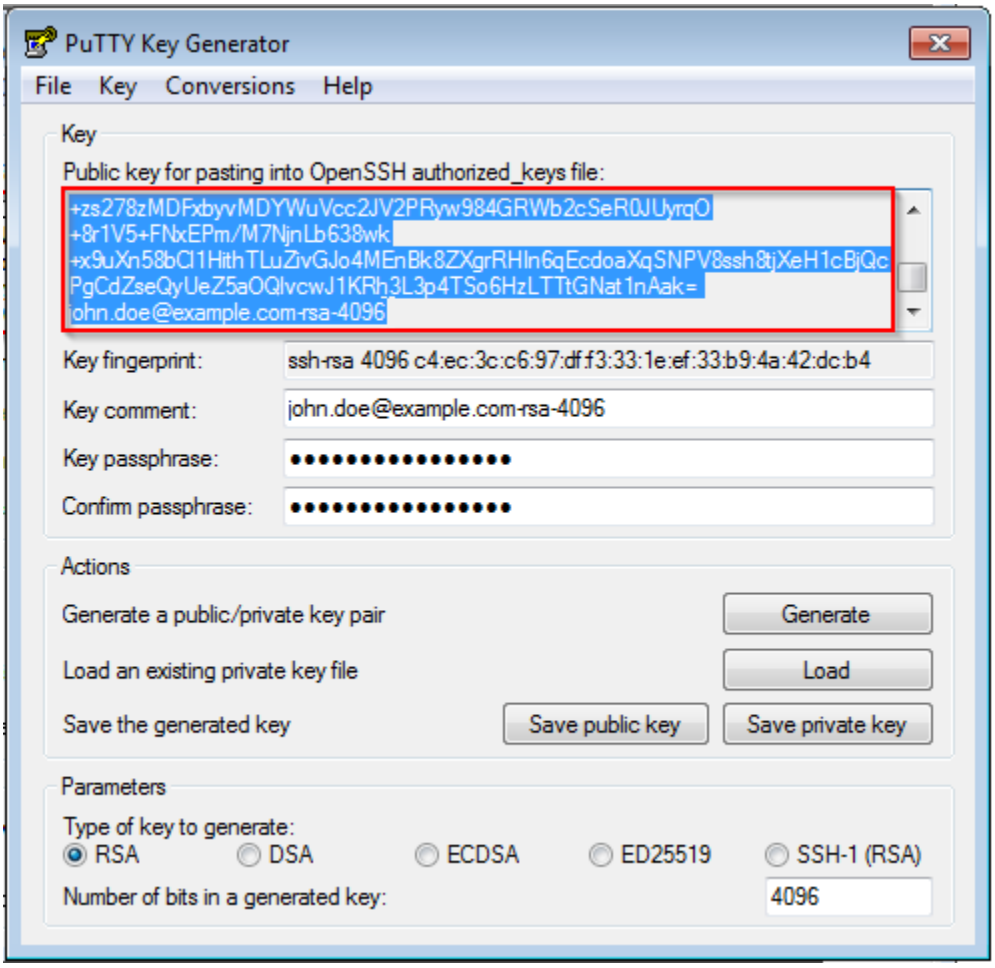
Faites bouger aléatoirement votre curseur dans l'espace blanc, jusqu'à ce que la clé soit générée :



Une fois générée, mettez un commentaire (*1*), un mot de passe fort (*2*), sauvez ensuite votre clé privée (*3*).



Vous devez maintenant envoyez votre clé public RSA à votre administrateur PaSSHport. Sélectionnez votre clé publique comme montré dans cette capture d'écran, et copiez/collez la dans un courriel à destination de votre administrateur PaSSHport :



Vous devez maintenant attendre que votre administrateur PaSSHport ajoute votre clé à votre compte dans PaSSHport.

Sur Linux / Unix :

Ouvrez simplement un terminal, et utilisez la commande *ssh-keygen*. Ici, non générerons une clé RSA d'une longueur de 4096 bits :

```
user@host:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user/.ssh/id_rsa):
Created directory '/home/user/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user/.ssh/id_rsa.
Your public key has been saved in /home/user/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:1r28XcYMIclivAHSSqmzH5Dh1LJ+IMsQMhl2Ds1HtXQ user@passhport-debian9-dev
The key's randomart image is:
+---[RSA 4096]----+
|.=o..oo=.E       |
|* +o+.=.+o . .   |
|.o +.B o  = + .  |
|. . O .  o = . . |
```

(suite sur la page suivante)

(suite de la page précédente)

```
| o + =  S o . .  |
|  o o o.   . . + |
|     o .    o   =|
|      .      o o |
|             . . |
+----[SHA256]-----+
user@host:~$
```

Affichez votre clé nouvellement créée :

```
user@host:~$ cat .ssh/id_rsa.pub
ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAACAQDmcmVG6uGW3BvOkHN7M7ubITihVwL9glc7jilZvzgDJL4CzCXG2VwjdxHaCBfW82HsgoU
↪TEZw/tgUWSSo7TOmldFmEs4TmZc9n0lhCgGT/XtShqtwyYAxeAw419Uc+L/
↪unXKPRtulLjNqdp62GW68CTQ7GzJosDWLYWZfNrhRoMvw6K6j/
↪vLbVcoktY+RNoNdFjYhgPcKzP0p73pvlh9uIKohBkh3vh5pOfVEu6L9J4VvjM3dACScPJORG05N7MB4rJ3FpSy9fgfMwaT99Xm
↪IVKXZxoUjB9z2EBkKYK+Hlj5Oopwgas6AvcrJIdZo1tsbdUYqcbQKoX7TeSwjbxESygFuCLMgs4SuMy8/
↪1+pPIlJQY7XzdCCDzkEp/
↪s12Ca5xPSUpFGdWKKIf1jzZzjS5BeUzm63ldFoN+HHKuU7FRpPNSXrlWNkqkwHnpa1SbhT3yOlu6BdnxMcaRNAeQ+cfxyykUSdc
↪toXNiXpyhrg3RT35Pj96cx7nwg9CrQ== user@passhport-debian9-dev
user@host:~$
```

Et envoyez le contenu à votre administrateur PaSSHport. Vous devez maintenant attendre que votre administrateur PaSSHport ajoute votre clé à votre compte dans PaSSHport.

1.5.2 Se connecter à PaSSHport

Sur Windows

Vous pouvez utiliser le client *Putty* pour vous connecter à PaSSHport, en indiquant votre clé privée.

Sur Linux / *nix

Utilisez votre client standard SSH et connectez vous à PaSSHport en tant qu'utilisateur *passhport* :

```
ssh passhport@bastion.tld
```

Si vous voulez vous connecter directement à une target, vous devez spécifier le nom de la target dans la commande ET utiliser l'option *-t* (pour forcer le rattachement à un terminal, sans quoi vous aurez un comportement incertain) :

```
ssh -t passhport@bastion.tld targetname
```

Si vous voulez lancer directement une commande, vous pouvez utiliser la même syntaxe, en ajoutant la commande à la fin

```
ssh -t passhport@bastion.tld targetname 'cat /proc/cpuinfo'
```

1.5.3 scp à travers PaSSHport

En utilisant la CLI

Pour utiliser SCP à travers PaSSHport, utilisez l'une des syntaxe suivante…

Depuis la target distante vers la machine local :

```
$ scp passhport@my-passhport-server:TARGET_NAME//etc/fstab /tmp/.
```

Depuis la machine locale vers la target distante :

```
$ scp /etc/passwd passhport@my-passhport-server:TARGET_NAME//tmp/.
```

Pour Windows, vous devez utiliser la commande "pscp". Vous pouvez la trouver ici : https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html . Attention, vous devrez avoir votre clé au format "ppk". Vous pouvez utiliser puttygen pour en générer une, ou convertir votre clé privée actuelle. La commande pscp, par défaut, utilise le protocole SFTP, ce qui ne peut pas fonctionner, et mène à l'erreur suivante :

```
FATAL ERROR: Received unexpected end-of-file from server
```

Pour pouvoir utiliser pscp avec PaSSHport, ajouter l'option "-scp" :

```
pscp -scp -i "C:\path\to\sshkeys\my_private_key.ppk"  "C:\path\to\file\totransfer.txt
↪" passhport@my-passhport-server:TARGET_NAME//pah/to/copy
```

Quelques liens :

- Site du projet : <http://www.passhport.org>
- Github : <https://www.github.com/LibrIT/passhport>