Serveur SSH
rédigé pour AlmaLinux 8.5

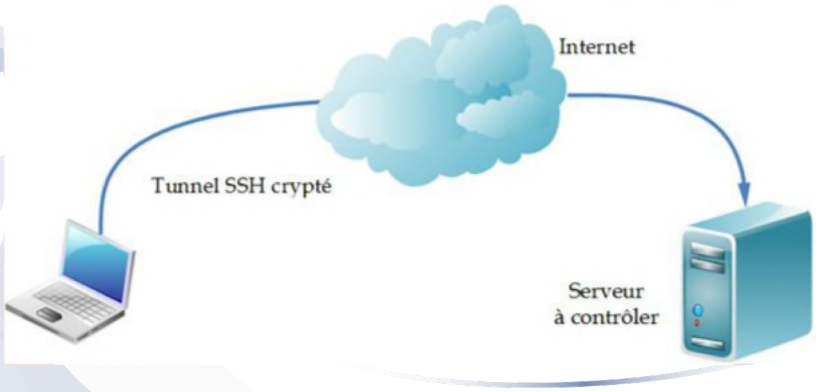
Hainaut Patrick 2022

But de cette présentation

- Apprendre à configurer le service SSH
- Le service SSH, configuré sur notre serveur Linux, nous donnera un accès à distance sécurisé, à ce serveur
- On pourra donc se connecter au serveur à partir d'un poste de travail Windows avec un client SSH tel que Putty ou WinScp

Principe

- A travers Internet (ou le réseau local), on crée un tunnel sécurisé pour ouvrir un « shell » ou pour transférer des petits fichiers



Principe

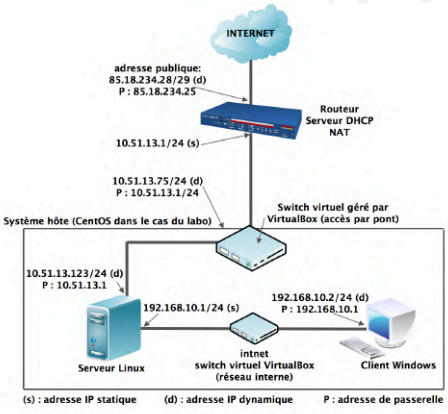
- **Secure Shell (SSH)** est à la fois un programme informatique et un protocole de communication sécurisé
- Le protocole de connexion impose un échange de clés de chiffrement en début de connexion
- Par la suite toutes les trames sont chiffrées. Il devient donc impossible d'utiliser un « sniffer » pour voir ce que fait l'utilisateur
- Le protocole **SSH** a été conçu avec l'objectif de remplacer divers outils tel que telnet (qui transmet en clair, y compris les passwords)

Principe

- Habituellement le protocole SSH utilise le port 22 et est particulièrement utilisé pour ouvrir un shell (console) sur un ordinateur distant
- SSH peut également être utilisé pour « forwarder » des ports TCP d'une machine vers une autre, créant ainsi un tunnel. Cette méthode est couramment utilisée afin de sécuriser une connexion qui ne l'est pas (par exemple le protocole email POP3) en la faisant transférer par le biais du tunnel chiffré SSH
- Rien n'empêche de faire plusieurs sauts entre consoles SSH, c'est-à-dire ouvrir une console sur un serveur, puis, de là, en ouvrir une autre sur un autre serveur

Schéma réseau de notre manipulation

- Nous aurons accès à notre serveur Linux, soit à partir du PC client du réseau local (la machine Windows), soit à partir d'un PC client du réseau étendu (du point de vue du serveur Linux) Ce sera la machine hôte
- La carte réseau cible sera différente suivant le cas; enp0s8 à partir du LAN, enp0s3 à partir du WAN



Installation du serveur SSH

- La version choisie est **OpenSSH**, une version libre de la suite de protocoles de SSH (développée dans le cadre s'OpenBsd)
- Pour l'installer, il suffit d'installer le paquet openssh-server via dnf:

```
dnf install openssh-server
```


Mise en oeuvre

- Pour démarrer le service:

```
systemctl start sshd
```

- Pour le stopper, remplacer start par stop
- Pour le redémarrer, remplacer start par restart (pour relire la config.)
- Pour qu'il soit actif au redémarrage du serveur:

```
systemctl enable sshd
```

Configuration

- La configuration par défaut donne un serveur ssh écoutant sur le port 22 et acceptant les connexions par login/mot de passe
- Nous verrons comment le sécuriser plus loin

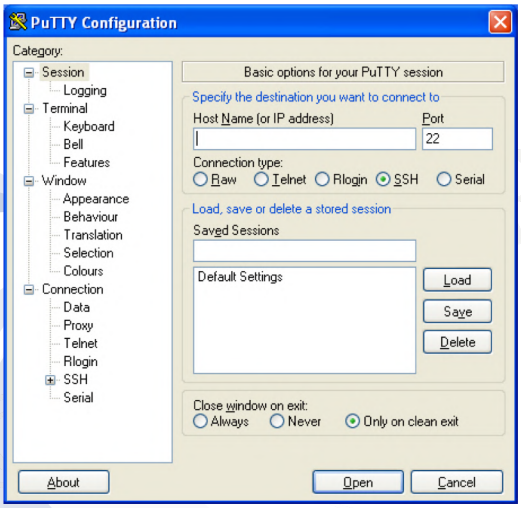
Test sur le PC Client

- Maintenant que notre serveur SSH est actif, on peut se connecter à notre serveur virtuel au moyen d'un client SSH
- Il existe deux clients bien pratiques tournant sous Windows; putty et winscp
- Putty permet d'ouvrir un shell distant sur le serveur virtuel
- Winscp permet le transfert de fichiers sécurisé (adaptation de Secure CoPy)

Test avec Putty sur le PC client

- Télécharger Putty sur le PC client de votre serveur virtuel
- Pas d'installation nécessaire, il suffit de lancer l'exécutable

Test avec Putty sur le PC client



Entrez l'IP du serveur (adr. de passerelle)

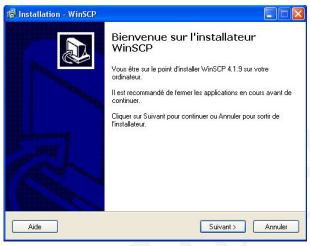
Vérifiez que le type de connexion est bien SSH et le port de destination 22

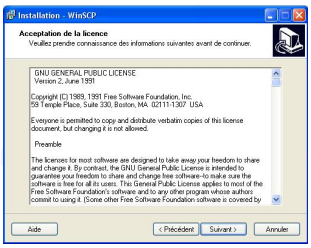
cliquez sur « Open »

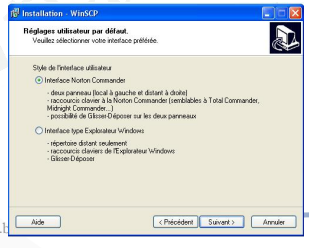
Test avec Putty sur le PC hôte

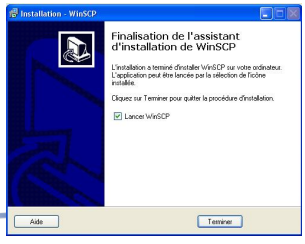
- Même principe qu'avec le PC client, mais l'adresse IP à indiquer est différente
- Laquelle allez-vous mettre ?

Installation de WinScp sur le PC client









Test de WinScp sur le PC client

- Dans « Nom d'hôte », entrez l'adresse IP du serveur
- Entrez aussi le nom d'un utilisateur reconnu sur le serveur et son mot de passe
- Vous pouvez maintenant transférer des fichiers entre les deux ordinateurs

Sécurisation du serveur ssh

- Désactiver la connexion distante au moyen du root:
  - Le compte root peut tout faire sur le serveur, si il tombe dans de mauvaise mains, c'est dramatique …
  - On peut interdire la connexion du root par ssh
  - Pour taper des commandes avec droits root, on utilisera sudo
  - Dans /etc/ssh/sshd_config, passez la variable PermitRootLogin de yes à no

    Redémarrer le service par systemctl restart sshd

Sécurisation du serveur ssh

- Changer le port d'écoute par défaut:
  - Les attaques envers ssh vont cibler le port 22 qui estt le port serveur par défaut
  - On peut le changer par un numéro de port supérieur à 1024 (les ports inférieurs sont des ports réservés)
  - Dans /etc/ssh/sshd_config, activez la variable Port en enlevant le # et changez le numéro de port
    Exemple: Port 2022

    Redémarrer le service par systemctl restart sshd

Gestion des utilisateurs

- On peut aussi aller plus loin en indiquant quels utilisateurs peuvent se connecter en ssh
- Toujours dans /etc/ssh/sshd_config, on emploie des mots clés
- Les mots clés sont examinés dans l'ordre indiqué:
  DenyUsers, **AllowUsers**, **DenyGroups** et **AllowGroups**
- Les arguments sont constitués d'une liste de noms séparés par des espaces

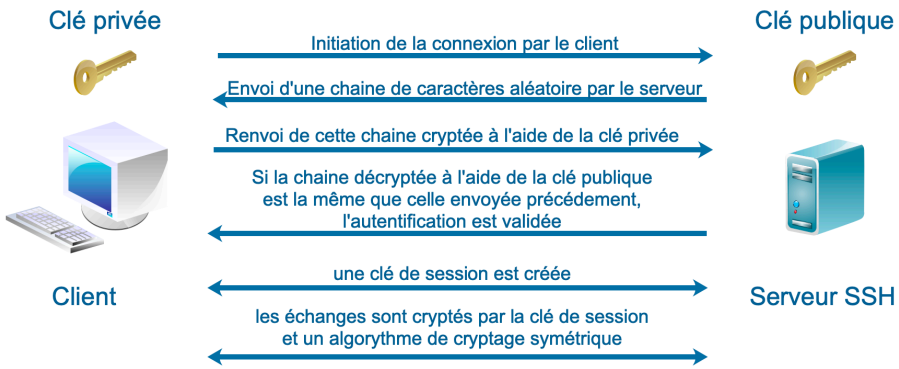
  Ex.: DenyUsers root
  AllowUsers tintin milou

Authentification par un système de clés publique/privée

- Avec l'authentification typique via identifiant-mot de passe, si quelqu'un connaît votre mot de passe, la sécurité est compromise
- Pour être débarrassé du problème, SSH offre l'Authentification par clé publique/privée au lieu des mots de passe « simples »
- De cette manière, il faut être en possession de non plus une mais de deux informations pour se connecter (avoir la clé privée & connaître le mot de passe de cette clé)

Authentification par un système de clés publique/privée

- Principe de fonctionnement:



Authentification par un système de clés publique/privée

- Ceci peut permettre par exemple:
  - À un administrateur de se connecter à des centaines de machines sans devoir connaître des centaines de mots de passe différents
  - De ne pas avoir un mot de passe à saisir toutes les 2 minutes (en utilisant *ssh-agent*)
- A moins que vous n'ayez déjà un couple de clés, vous devez d'abord en créer

Authentification par un système de clés publique/privée

- Le chiffrement asymétrique consiste à transformer une donnée, en utilisant un élément dit « clé publique », en quelque chose d'inintelligible pour tous, sauf pour celui qui possède la « clé privée » qui correspond à la clé publique
- En utilisant la clé privée, on peut faire l'opération inverse, et retrouver les données

Authentification par un système de clés publique/privée

- Le protocole SSH existe en deux versions: SSH1 et SSH2
- SSH1 utilise une paire de clés de type RSA1 pour l'authentification
- SSH2 utilise une paire de clés de type DSA ou RSA (différent de RSA1)

Authentification par un système de clés publique/privée

- RSA est basé sur un calcul de factorisation
- DSA est basé sur un calcul de logarithme discret
- DSA existe principalement parce qu'à l'époque où le gouvernement américain a voulu définir un logarithme standard de signature électronique pour son administration, RSA était encore protégé par un brevet
- Les deux types sont de sécurité équivalentes, RSA étant plus utilisé

Génération des clés sur le serveur

- Nous utiliserons la commande ssh-keygen
  Exemple: ssh-keygen -t rsa -b 4096

  -t permet de définir le type de clés: rsa ou dsa
  -b permet d'en définir la taille

- Vous avez la possibilité de spécifier une passphrase pour protéger la clé privée et l'endroit où se trouveront les clés

- Par défaut ce sera dans le répertoire .ssh créé dans celui de l'utilisateur

Génération des clés sur le serveur

- Deux clés sont générées: id_rsa (privée) et id_rsa.pub (publique)

- La clé publique est à copier sur le serveur dans le fichier ~/.ssh/authorized_keys

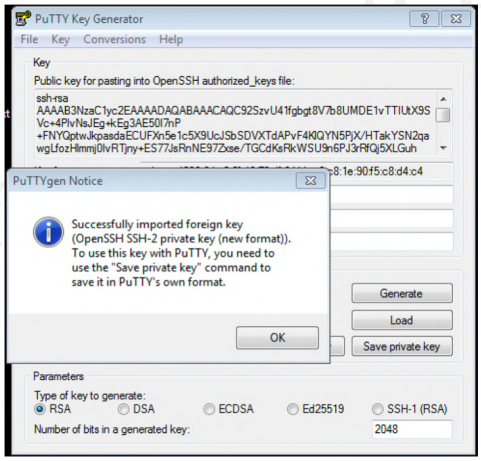
  ~ étant le répertoire de l'utilisateur (/root ou /home/hainautp par exemple)

- La clé privée est à transférer sur le client de manière sécurisée via winscp si c'est un client Windows ou via l'utilitaire ssh-copy-id si c'est un client Linux

  Exemple: ssh-copy-id hainautp@192.168.10.2

Adaptation de la clé privée sous Windows

- Une fois la clé privée transférée sur le client Windows, il faut la convertir pour qu'elle puisse être utilisée par putty ou WinSCP
- Pour ça, on va utiliser putttygen.exe
- On clique sur "Load" pour charger la clé et sur "Save private key" pour la sauver au format .ppk

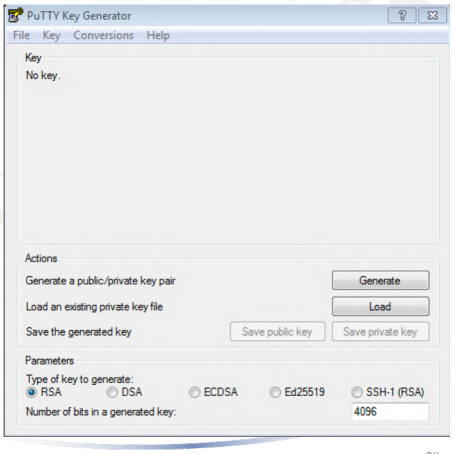


Génération des clés

- Nous venons de voir comment générer les clés sur le serveur, mettre la clé publique au bon endroit sur le serveur et adapter la clé privée sur le client pour qu'elle fonctionne avec putty ou winscp
- On peut générer les clés sur le client et transmettre la clé privée au serveur, c'est ce que nous allons voir dans les dias suivantes
- On choisira une méthode ou l'autre, pas les deux ;-)

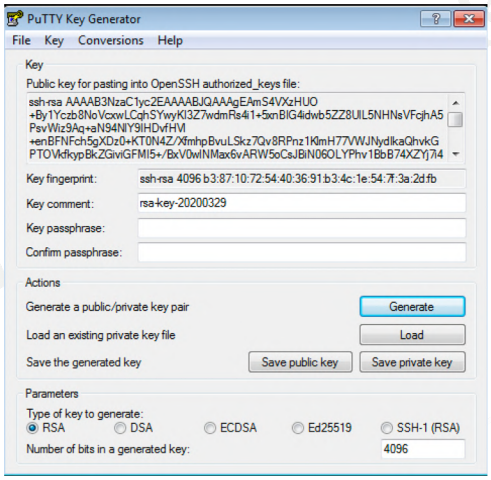
Génération des clés sur le client

- Nous allons utiliser pour cela, de nouveau, puttygen.exe
- Choisissez le type de clés à générer (RSA 4096 bytes dans notre exemple)
- Puis, cliquez sur « Generate »



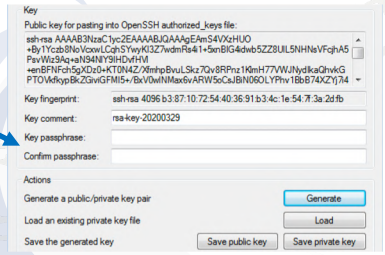
Génération des clés sur le client

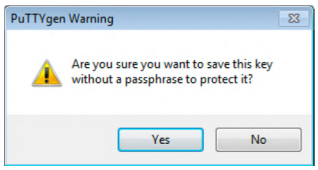
- Pour générer les clés, il faut faire bouger la souris …
- Une fois les clés générées, vous pouvez les sauver
- La clé privée sera automatiquement au format .ppk



Génération des clés sur le client

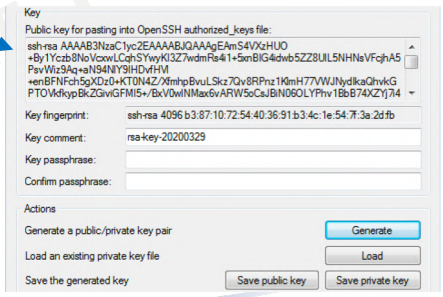
- Pour la clé privée, le système demande confirmation si une passphrase n'est pas entrée pour protéger cette clé
- Le fait de rentrer une passphrase augmente la sécurité mais demande qu'on rentre un mot de passe à chaque utilisation de la clé





Génération des clés sur le client

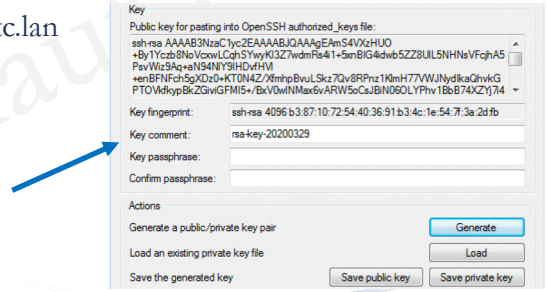
- Pour la clé publique, il faudra modifier sa structure pour qu'elle puisse fonctionner avec openssh-server
- Préférez la clé affichée que celle sauvée par "Save private key"
- Faites un copier-coller vers un fichier texte (tout sera sur une ligne à cause des retour à la ligne différents sous Windows et Linux)



Génération des clés sur le client

- Il faudra ensuite changer le "Key comment" à la fin du fichier texte
- Au lieu de "rsa-key …", vous devez spécifier l'utilisateur avec lequel on va se connecter + @ + le nom DNS du serveur Linux

  Exemple: root@srv.cdatc.lan



Génération des clés sur le client

- Ce qui donne par exemple:

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQC92SzvU41fgbgt8V7b8UMDE1vTTIUtX9SVc+4PlvNs
JEg+kEg3AE50I7nP+FNYQptwJkpasdaECUFXn5e1c5X9UcJSbSDVXTdAPvF4KlQYN5PjX/HTakYSN2qa
wgLfozHlmmj0IvRTjny+ES77JsRnNE97Zxse/TGCdKsRkWSU9n6PJ3rRfQj5XLGuh+tQ2HOJbtSXkF4p
hVJjL1J3iao2CCPY5EpM1XSqA8yaU4NNT0MlF4+bs4WPXW3Z0BmYpfsBd91kFBGIVt43BdeQpJnVIR3h
ip2J2drS2ip/f/qXwdfus9H7GUOzkjI6piFB5R0TOXwcsAr5rh8cpSbU7LePLM7AfDuY9HypJXdcTO/r
Pxa0HnPdkJo6HMjGlJBsML/AxDHF5iYiHBIXMfXcaD57iTnm0Qf35B520IqD9WSSP36DpsydpLRywZB1
soypBC9dU/DFGhDeKhQgL9XxAhK0fopqdj1g3qgmcvdXmo8SIt3LyeRVUM2j8aKuegTavx6L1sx2OjaQ
VBSojSizhSTTlWaNwcRqA5kPEIv5rrIXanJypx6KDCrpNjvWaQu7OrI4vgqfAcgzPXqF3NTwj5Xhl/EH
09xE5pal+4NA7NwHwReh2OebJJFzdM4o6nLSRcKv2DMA8AFjA+rRMmDj6sY+GIlA+a6CremmYIG8YI7k
Dw== root@srv.cdatc.lan
```

- ssh-rsa au début et root@srv.cdatc.lan à la fin ne font pas partie de la clé proprement-dite

Installation de la clé publique sur le serveur distant

- Il faut maintenant copier la clé publique sur le serveur dans le fichier ~/.ssh/authorized_keys, pour qu'il puisse crypter les messages
- Les permissions sur ce fichier doivent être mises en lecture seule
- ~ représente le répertoire de l'utilisateur

Adaptation de sshd_config

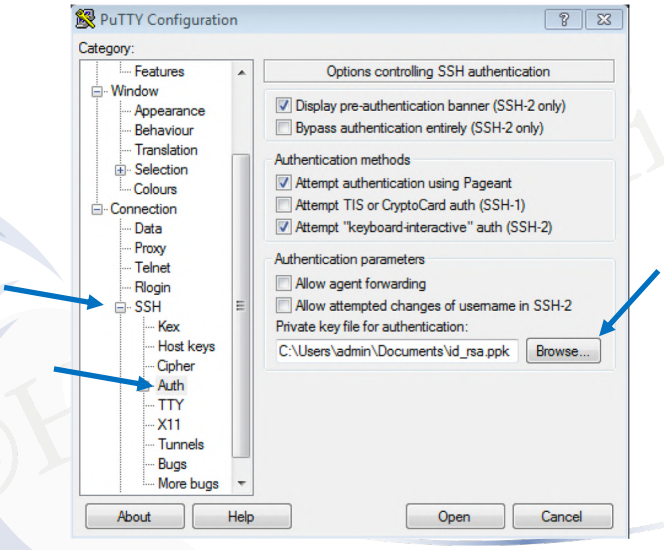
- Pour passer d'une connexion par login/mot de passe à une connexion par clés, il faut modifier quelques lignes dans /etc/ssh/sshd_config

  PubkeyAuthentication no -> PubkeyAuthentication yes

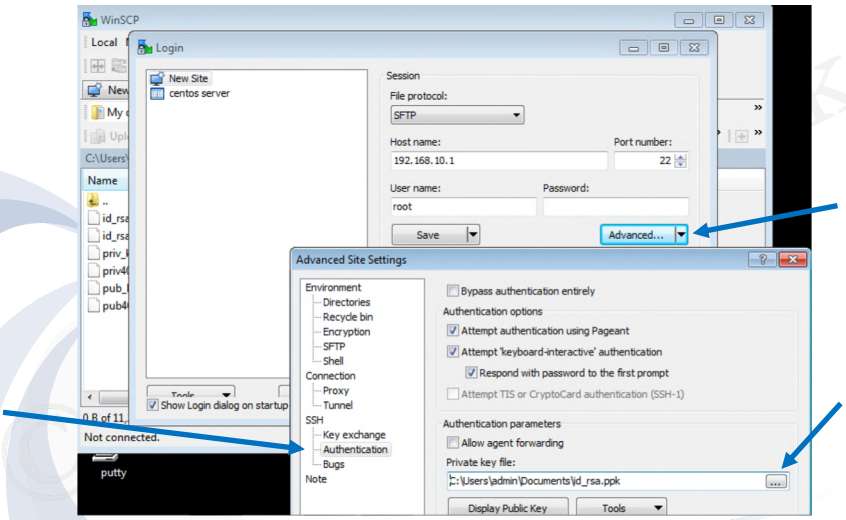
  PasswordAuthentication yes -> PasswordAuthentication no

- N'oubliez pas de redémarrer le service:
  systemctl restart sshd

Utilisation de putty avec la clé privée



Utilisation de Winscp avec la clé privée



Conclusion

- Nous avons, maintenant, la possibilité de contrôler notre serveur à distance, ce qui est vraiment très utile
- Il faut, bien sûr, pour cela, que notre serveur soit « online »
- S'il est « planté », ce ne sera pas possible de le redémarrer à distance, bien qu'il existe des cartes de contrôle qui permettent de le faire, comme les cartes remote insight de chez Compaq (HP maintenant)

