Cours 42 : Secure Shell

Dans ce cours nous verrons le fonctionnement du protocole Secure Shell (SSH) qui est utilisé pour se connecter à un appareil et le configurer par la ligne de commande (CLI). Une option pour se connecter à un appareil et le configurer est par le moyen d'un port console. Il est aussi aussi possible de se connecter à un appareil à distance avec l'adresse IP par le moyen de SSH.

Dans ce cours nous verrons le fonctionnement de console port security, puis nous verrons ce qu'est la couche 2 commutateur de gestion IP qui ne route pas les paquets et ne construit pas de table de routage. Il est tout de même possible de configurer et gérer une adresse IP de gestion pour ces appareils pour pouvoir y accéder à distance.
Nous verrons ensuite le fonctionnement de Telnet qui est un protocole similaire à SSH.
Et nous verrons en dernier temps le fonctionnement de SSH.

Tout d'abord voyons le fonctionnement de console port security. Par défaut aucun mot de passe n'est requis pour accéder au CLI d'un appareil Cisco IOS par le port console, il est possible de configurer un mot de passe sur la ligne de console. Après cela l'utilisateur devra entrer un mot de passe pour accéder au CLI par le port console.
Pour cela il faut configurer l'appareil avec les commandes suivantes :

```
R1(config)#line console 0

R1(config-line)#password ccna

R1(config-line)#login

R1(config-line)#end
R1#exit


R1 con0 is now available

Press RETURN to get started.

User Access Verification

Password:

R1>
```

- `line console 0` : sert à se connecter à la ligne de console puisqu'il n'y a qu'une seule ligne de console ou en d'autre terme qu'il n'est possible qu'il y ait qu'une seule connexion à la fois le numéro sera donc toujours 0 (à part dans le cas ou il est possible de connecter plusieurs utilisateurs en ligne de console)
- `password ccna` : sert à configurer le mot de passe « ccna »
- `login` : indique à l'appareil qu'il est requis que l'utilisateur entre le mot de passe configuré pour accéder au CLI par le port console.

Avec cette configuration un mot de passe est à présent requis pour pouvoir se connecter à la ligne de commande de l'appareil.
On remarque que lorsque l'on écrit le mot de passe lors de la connexion il n'apparaît pas en clair sur celui ci, cela permet à ce qu'il ne soit pas visible pour plus de sécurité.

Alternativement il est possible de configurer la ligne de console pour que l'utilisateur se connecte avec l'un des noms utilisateurs configurés sur l'appareil.
On utilise pour cela les commande suivante :

```
R1(config)#username jeremy secret ccnp

R1(config)#line console 0

R1(config-line)#login local

R1(config-line)#end
R1#exit



R1 con0 is now available


Press RETURN to get started.

User Access Verification

Username: jeremy
Password:
R1>
```

- `username jeremy secret ccnp` : sert à créer l'utilisateur jeremy et pour mot de passe associé ccnp
- `line console 0` : sert à configurer la ligne de console comme expliqué auparavant
- `login local` : indique à l'appareil qu'il est requis que l'utilisateur entre le nom d'utilisateur et mot de passe configurés localement pour accéder au CLI par le port console.

On peut aussi configurer l'appareil avec les commandes suivantes pour configurer son interface :

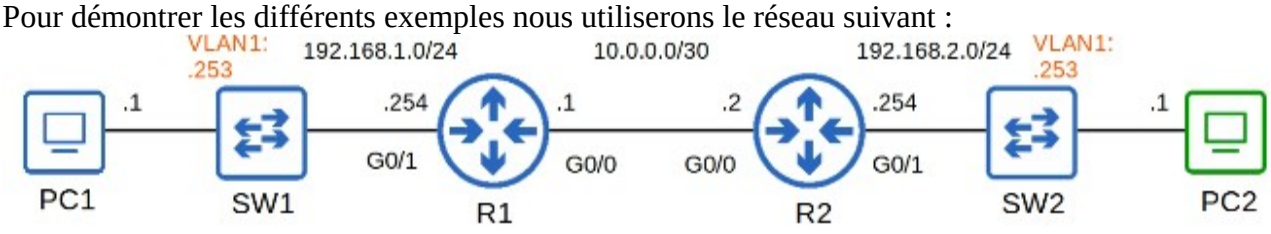
```
line con 0
 exec-timeout 3 30
 password ccna
 logging synchronous
 login local
```

- `exec-timeout 3 30` : permet de déconnecter l'utilisateur après 3 minutes et 30 secondes d'inactivité.

Avec cette configuration il est requis pour l'appareil d'utiliser un nom d'utilisateur et un mot de passe pour se connecter au CLI puisque la commande login local est lancé en priorité par rapport à une connexion direct en utilisant un simple mot de passe.

Voyons à présent le fonctionnement de la couche 2 du Switch et de la gestion des IP. Cette couche ne fais pas fonctionner le routage et ne construit pas de table de routage. Elles ne font pas de routage pour les IP. Il est possible d'assigner une adresse IP à un SVI pour permettre de se connecter à distance au CLI du Switch (en utilisant Telnet ou SSH)

Pour démontrer les différents exemples nous utiliserons le réseau suivant :



On commence par configurer les adresses IP et les vlan avec les commandes suivantes :

```
SW1(config)#interface vlan1
SW1(config-if)#ip address 192.168.1.253 255.255.255.0
SW1(config-if)#no shutdown
SW1(config-if)#exit

SW1(config)#ip default-gateway 192.168.1.254
```

On configure tout d'abord l'adresse IP sur le SVI de la même manière qu'un switch Multicouche et on active l'interface si nécessaire.
On configure ensuite la passerelle du switch par défaut. Dans ce cas le PC2 n'est pas dans la même LAN que SW1. Si SW1 n'a pas de passerelle par défaut il ne pourra pas communiquer avec le PC2.

Voyons le fonctionnement de Telnet, il s'agit d'un protocole plus très utilisé puisque pas très sécurisé mais il est bien de le connaître avant de voir SSH.
Telnet (Teletype Network) est un protocole utilisé pour accéder à distance au CLI d'un hôte distant.
Telnet a été développé en 1969 et a été largement remplacé par SSH qui est plus sécurisé.
SSH a quant à lui été développé en 1995, Telnet envoie des données en texte clair dans cryptage.
Lorsque l'on utilise Telnet on peut capturer les paquets avec Wireshark :

```
348 09:38:22.133251 10.0.0.1          10.0.0.2          TELNET    66 Telnet Data ...
> Frame 348: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface -, id 0
> Ethernet II, Src: 0c:54:cc:2a:0d:00 (0c:54:cc:2a:0d:00), Dst: 0c:54:cc:62:0c:00 (0c:54:cc:62:0c:00)
> Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.0.2
> Transmission Control Protocol, Src Port: 23, Dst Port: 28772, Seq: 681, Ack: 33, Len: 12
v Telnet
    Data: \r\n
    Data: Password:

350 09:38:23.416474 10.0.0.2          10.0.0.1          TELNET    60 Telnet Data ...
> Frame 350: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface -, id 0
> Ethernet II, Src: 0c:54:cc:62:0c:00 (0c:54:cc:62:0c:00), Dst: 0c:54:cc:2a:0d:00 (0c:54:cc:2a:0d:00)
> Internet Protocol Version 4, Src: 10.0.0.2, Dst: 10.0.0.1
> Transmission Control Protocol, Src Port: 28772, Dst Port: 23, Seq: 33, Ack: 693, Len: 4
v Telnet
    Data: ccnp
```

On voit que dans la capture on peut identifier le mot de passe qui n'est pas crypté et qui est utilisé par Telnet, le mot de passe entré est ici ccnp.
Le serveur Telnet auquel l'appareil essaie de se connecter écoute pour Telnet sur le port TCP 23.
Voici les commandes utilisés pour configurer Telnet sur le SW1 :

```
SW1(config)#enable secret ccna
SW1(config)#username jeremy secret ccna
SW1(config)#access-list 1 permit host 192.168.2.1

SW1(config)#line vty 0 15

SW1(config-line)#login local

SW1(config-line)#exec-timeout 5 0

SW1(config-line)#transport input telnet

SW1(config-line)#access-class 1 in
```

- `enable secret ccna` : avec cette commande si un mot de passe n'est pas configuré, il ne sera pas possible de se connecter en mode privilèged exec mode en utilisant Telnet.
- `username jeremy secret ccna` : sert à configurer un nom d'utilisateur/mot de passe
- `access-list 1 permit host 192.168.2.1` : permet de configurer un ACL pour limiter quelle appareil peut se connecter à la ligne VTY
- `line vty 0 15` : l'accès Telnet/SSH est configuré sur la ligne VTY. Il y a 16 lignes disponible, donc jusqu'à 16 utilisateurs peuvent se connecter en même temps (VTY est l'acronyme de Virtual TeleType)
- `login local` : sert à ce que l'utilisateur se connecte uniquement par le moyen d'une connexion avec un nom d'utilisateur/mot de passe local.
- `exec-timeout 5 0` : sert à configurer le délai de déconnexion à 5 minute en période d'inactivité.
- `transport input telnet` : permet une connexion uniquement par Telnet

il y a d'autres possibilité de commande pour autoriser d'autres type de connexion par exemple :
- `transport input ssh` : permet de n'autoriser que les connexion SSH
- `transport input telnet ssh` : permet d'autoriser les deux
- `transport input all` : permet tout type de connexion
- `transport input none` : ne permet aucune connexion
- `access-class 1 in` : Applique les ACL sur la ligne VTY,

on peut aussi configurer l'ACL sur la ligne VTY avec la commande : `access-class` qui applique une ACL sur la ligne VTY, la commande : `ip access-group` applique une ACL sur une interface.

```
R2#ping 192.168.1.253
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.253, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 10/11/16 ms

R2#telnet 192.168.1.253
Trying 192.168.1.253 ...
% Connection refused by remote host
```

Lorsque l'on essaye de ping le SW1 avec R2 on peut voir que le ping fonctionne, cependant lorsque l'on tente de se connecter en utilisant Telnet on obtient un message d'erreur qui indique que la connexion est refusé cela est dû à l'ACL configuré sur la ligne VTY. Avec cette ACL seulement le PC 2 peut se connecter en Telnet au SW1, la connexion fonctionne bien avec PC2 :

```
C:\Users\user>telnet 192.168.1.253
Connecting To 192.168.0.1…

User Access Verification

Username: jeremy
Password:
SW1>
```

La ligne VTY est configuré de la manière suivante sur l'appareil

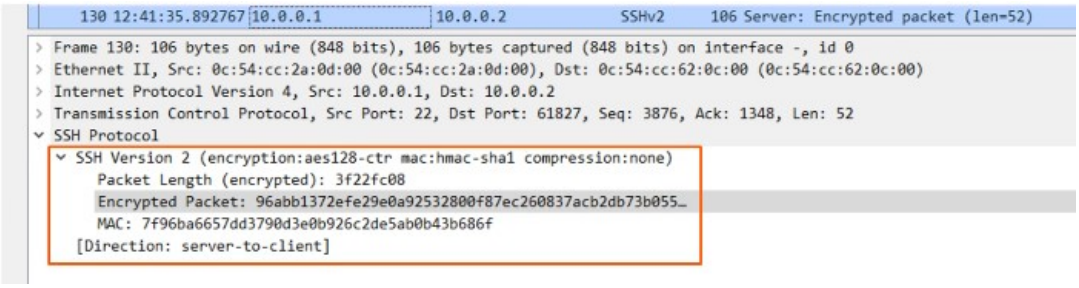
Ici jusqu'à 5 connexion en simultanés sont permises.

```
line vty 0 4
 access-class 1 in
 exec-timeout 5 0
 login local
 transport input telnet
line vty 5 15
 access-class 1 in
 exec-timeout 5 0
 login local
 transport input telnet
```

Voyons à présent le fonctionnement de SSH.
SSH (Secure Shell) a été développé en 1995 pour remplacer les protocoles moins sécurisés comme Telnet.
Voici une définition de Shell donnée par Wikipédia :
« L'interface en ligne de commande (CLI, de l'anglais « command line interface ») permet à l'utilisateur d'interagir avec le système à partir de commandes qui sont adaptées au mode texte et qui permettent, entre autres, l'exécution d'applications affichées à l'origine (dans un système moderne, l'environnement graphique est aussi pris en compte) dans un environnement en mode texte (TUI pour text user interface) ;
La coque logicielle de type graphique fournit à l'utilisateur un environnement graphique (GUI, pour graphical user interface), généralement un environnement de bureau ou un écran d'accueil.. »

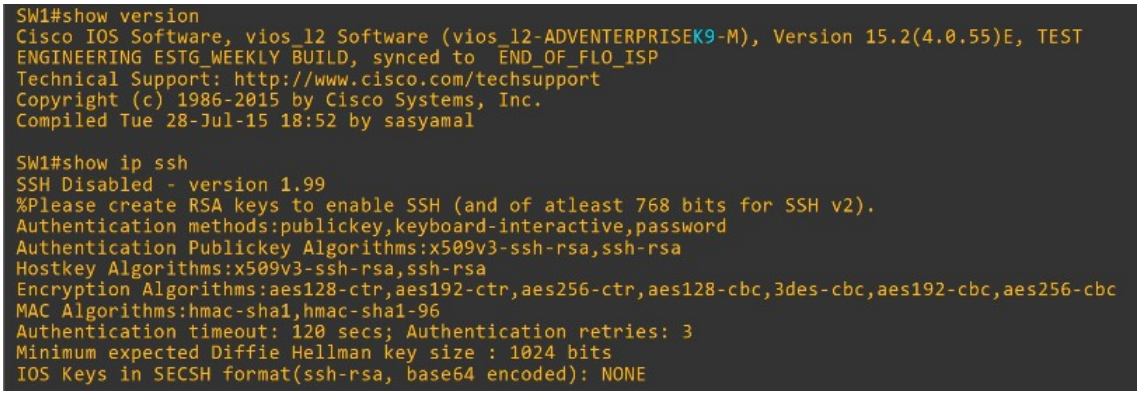
Donc à chaque fois qu'un utilisateur se connecte à une ligne de commande il utilise un Shell.
SSHv2 a la version majeur de révision de SSHv1 et a été publié en 2006.
La version 2 est plus sécurisé et devrait être utilisé le plus souvent possible.
Si un appareil supporte les versions 1 et 2, il dit de lancer la « version 1.99 », ça n'est pas une version de SSH mais cela signifie juste que l'appareil supporte les versions 1 et 2.
SSH fournit des fonctionnalités de sécurité comme le cryptage des données et l'authentification.

Voici l'exemple d'une capture Wireshark par SSH



Le paquet crypté est composé de caractère, seulement le serveur SSH et le client ont la clé pour déchiffrer le paquet. Le serveur SSH écoute le trafique sur le port 22.

Avant de configurer SSH, il faut vérifier si l'OS de l'appareil supporte SSH.



Pour cela on lance les commandes :
- `show version` : pour afficher la version de l'OS, on peut voir K9 à la fin (en bleu), les machines et OS qui supportent SSH ont K9 dans leurs noms. Cisco exporte des NPE (No payload Encryption) des images IOS aux pays qui ont des restrictions sur le chiffrement des technologies. Les images IOS NPE ne supportent pas les fonctionnalités de chiffrage comme SSH.

- `show ip ssh` : permet aussi d'afficher si SSH est supporté par l'appareil. Dans notre cas on peut voir la version de SSH, mais que SSH est désactivé.
On peut voir ici le message : « please create RSA keys to enable SSH (and of atleast 768 bits for SSH v2). Il s'agit de clés cryptographique qui permettent des fonctions essentiels dans la configuration de SSH.

Une fois qu'à été vérifié si l'appareil supporte SSH, on commence par configurer SSH, pour cela il faut générer une pair de clé publique RSA et une pair de clé privée RSA.
Les clés sont utilisés pour le cryptage et le décryptage des données, l'authentification, etc...
Voici les commandes à utiliser pour faire cela :

```
SW1(config)#ip domain name jeremysitlab.com

SW1(config)#crypto key generate rsa
The name for the keys will be: SW1.jeremysitlab.com
Choose the size of the key modulus in the range of 360 to 4096 for your
  General Purpose Keys. Choosing a key modulus greater than 512 may take
  a few minutes.

How many bits in the modulus [512]: 2048
% Generating 2048 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 1 seconds)

SW1(config)#
*Feb 21 04:22:35.778: %SSH-5-ENABLED: SSH 1.99 has been enabled

SW1(config)#do show ip ssh
SSH Enabled - version 1.99
Authentication methods:publickey,keyboard-interactive,password
Encryption Algorithms:aes128-ctr,aes192-ctr,aes256-ctr,aes128-cbc,3des-cbc,aes192-cbc,aes256-cbc
MAC Algorithms:hmac-sha1,hmac-sha1-96
Authentication timeout: 120 secs; Authentication retries: 3
Minimum expected Diffie Hellman key size : 1024 bits
IOS Keys in SECSH format(ssh-rsa, base64 encoded): SW1.jeremysitlab.com
[output omitted]
```

- `ip domain name jeremysitlab.com` : sert à configurer le nom de domaine. Le FQDN de l'appareil est utilisé pour nommer la clé RSA. Le FQDN est l'acronyme de : Fully Qualified Domain Name (Hostname + Nom de domaine)
- `crypto key generate rsa` : sert à générer la clé RSA, ici le nom de la clé est configuré automatiquement c'est SW1.jeremysitlab.com qui est aussi le FQDN de SW1
on choisie ensuite la taille du modulus ou taille de clé en bit. Ici 2048 bits.
On peut aussi utiliser la commande : `crypto key generate rsa modulus` suivie de la longueur de clé pour configurer directement en 1 commande la longueur de clé généré.

Une fois la clé généré un message apparaît indiquant que SSH est à présent activé.

Lorsque l'on vérifie le statut de SSH avec la commande : `show ip ssh` on peut voir à présent qu'il est activé.

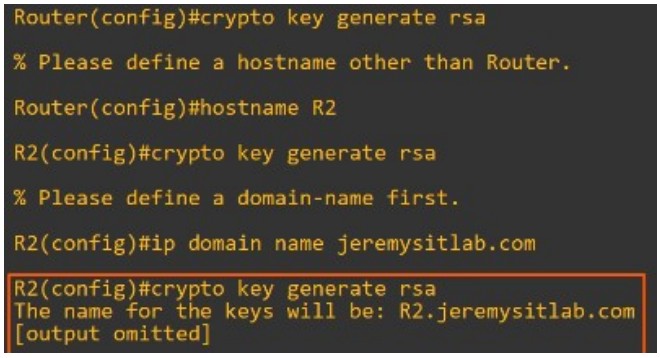
A présent que SSH est activé voyons les commandes à utiliser pour le configurer :

```
SW1(config)#enable secret ccna
SW1(config)#username jeremy secret ccna
SW1(config)#access-list 1 permit host 192.168.2.1
SW1(config)#ip ssh version 2
SW1(config)#line vty 0 15
SW1(config-line)#login local
SW1(config-line)#exec-timeout 5 0
SW1(config-line)#transport input ssh
SW1(config-line)#access-class 1 in
```

- `enable secret ccna` : avec cette commande si un mot de passe n'est pas configuré, il ne sera pas possible de se connecter en mode privilèged exec mode en utilisant Telnet.
- `username jeremy secret ccna` : sert à configurer un nom d'utilisateur/mot de passe local, ici jeremy et mot de passe ccna
- `access-list 1 permit host 192.168.2.1` : permet de configurer une ACL pour qu'il autorise seulement l'hôte : 192.168.2.1
- `ip ssh version 2` : est optionnel mais recommandé permet de restreindre SSH seulement à la version 2.
- `line vty 0 15` : permet de configurer toutes les lignes VTY tout comme Telnet.
- login local : permet d'activer l'authentification local, il n'est pas possible d'utiliser `login` pour SSH seulement `login local` fonctionne.
- `exec-timeout 5 0` : permet de configurer le exec timeout
- `transport input ssh` : est la meilleur pratique et permet de limiter les lignes de connexion VTY seulement pour SSH.
- `access-class 1 in` : est optionnel mais recommandé permet d'appliquer l'ACL pour restreindre la ligne de connexion VTY.

Voici les différentes étapes de configuration :
1) configurer le nom d'hôte
2) Configurer le DNS nom de domaine
3) Générer la pair de clé RSA
4) Configurer et activer le mot de passe, et nom utilisateur/motdepasse
5) Activer SSHv2 (seulement) ça n'est pas obligatoire mais recommandé.
6) Configurer lignes VTY

Pour générer une clé il est obligatoire de configurer un hostname et un nom de domaine d'abord
Comme on peut le voir :



Depuis un PC on peut utiliser la commande suivante pour se connecter par SSH :
`ssh -l username ip-address` OU `ssh username@ip-adress`