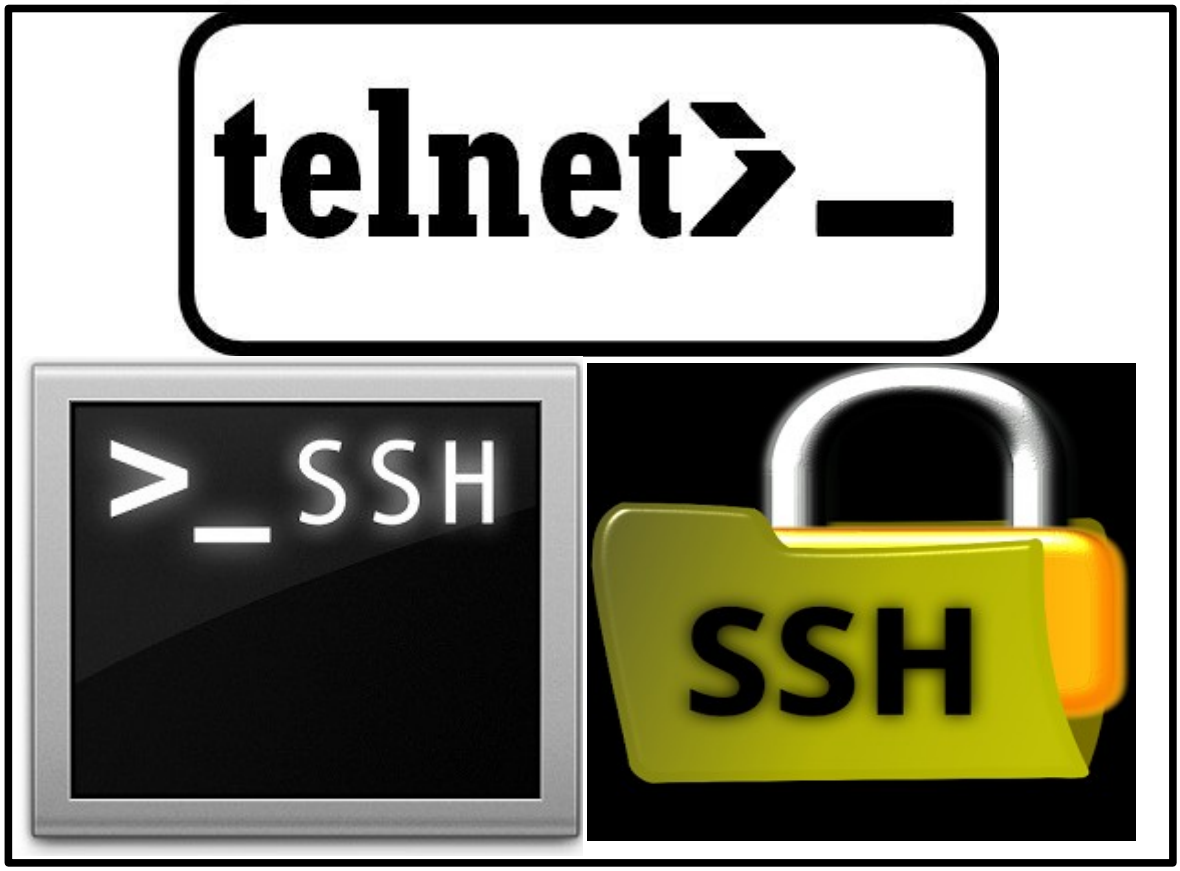
MOUSSAOUI
LEPETIT

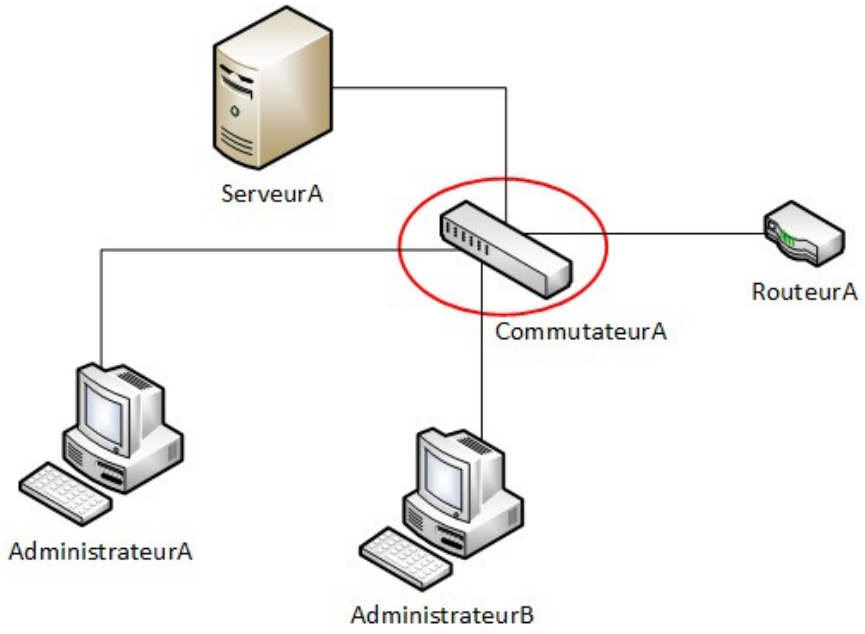
14/12/2015
BTS SIO2 SISR

Sécurité des accès aux matériels et aux serveurs



Sommaire

Schéma d'une liaison sécurisé à distance



Légende

Câble droit ———

Domaine SSH

Introduction

Nous avons toujours eux pour habitude de sécurisé nos éléments réseaux Cisco sans savoir réellement l'intérêt de mettre en place une sécurité, même si le mode administrateur possède un mot de passe crypter en md5.

C'est pourquoi dans ce TP nous allons mieux comprendre l'utilité de mettre en place un système authentification sur des éléments réseaux de couche 2 et 3. Et l'utilité d'avoir une méthode de cryptage RSA.

Malgré que les mots de passe sont cryptées, la connexion via Telnet présente toujours des risques ?

La méthode de cryptage RSA est elle réellement satisfaisante ?

Nous répondrons à ces deux problématique une fois que nous aurons les connaissances nécessaire. Prises au sein du TP.

Plan d'adressage

ServeurA

Adresse IP	Masque de sous-Réseaux	Passerelle
172.32.69.1	255.255.0.0	172.32.69.254

AdministrateurA

Adresse IP	Masque de sous-Réseaux	Passerelle
172.32.69.2	255.255.0.0	172.32.69.254

AdministrateurB

Adresse IP	Masque de sous-Réseaux	Passerelle
172.32.69.3	255.255.0.0	172.32.69.254

CommutateurA

Adresse IP	Masque de sous-Réseaux
172.32.69.253	255.255.0.0

RouteurA

Adresse IP	Masque de sous-Réseaux
172.32.69.254	255.255.0.0

Partie 1 – Mise ne place d'un accès sécurisé sur les matériels d'interconnexion

Dans cette première partie, nous allons essayer de se connecter à distance via le Telnet puis via le SSH et comprendre donc la différence entre ces deux protocoles de communication.

Test de piratage du commutateur

Avant de commencer par définir tous les mots de passe possibles sur le commutateur, on commence par d'abord l'adresser.

```
#On définit son nom d'hôte grâce aux syntaxe Cisco suivante :
Switch>enable
Switch#configure terminal
Switch(config)#hostname CommutateurA


#On entre dans la configuration de l'interface VLAN puis on l'adresse :
CommutateurA(config)#interface vlan 1
CommutateurA(config-if)#ip address 172.32.69.253 255.255.0.0
CommutateurA(config-if)#no shutdown
CommutateurA(config-if)#exit
CommutateurA(config)#
```

Maintenant on peut commencer par configurer la sécurisations de l'accès au commutateur.

```
#On commence par déclarer le mot de passe de l'accès en mode console :
CommutateurA(config)#line con 0
CommutateurA(config-line)#password ciscocon
CommutateurA(config-line)#login
CommutateurA(config-line)#exit
```

```
#Puis on déclarer le mot de passe d'accès par Telnet :
CommutateurA(config)#line vty 0 15
CommutateurA(config-line)#password ciscovty
#Avec cette syntaxe Cisco on autorise la connexion
CommutateurA(config-line)#login
CommutateurA(config-line)#exit
```

Dans notre cas le mot de passe est : ciscovty !

Le mode privilégié (enable) permet d'avoir plus de droit que le mode simple, c'est pourquoi on va définir un mot de passe que seule l'administrateur pourra y accédés. Tout en sachant que l'IOS possède un système de cryptage md5.

```
#On définit donc le mot de passe enable :
CommutateurA(config)#enable secret class
CommutateurA(config)#end
```

Le mot passe ici est : class

Ensuite on applique une **syntaxe Cisco** qui va nous permettre de voir notre configuration :

```
CommutateurA#show running-config
Building
configuration...

Current
configuration : 1380 bytes
!
version
12.2
no
service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname CommutateurA
!
boot-start-marker
boot-end-marker
!
enable secret 5 $1$WPXM$mhZBJr3wMhyHGG5p1.QHM/
```

On peut constater que le mot de passe enable est bien crypter d'où l'algorithme md5 !

```
!
!
ip http server
ip http secure-server
!

line con 0
 password ciscocon
 login
line vty 0 4
 password ciscovty
 login
line vty 5 15
 password ciscovty
 login
!
end

CommutateurA#
```

Par conséquent ce n'est pas le cas pour le mot de passe de l'accès par Telnet. Étant données que l'on a pas encore crypter ce mot de passe.

Ce qui est intéressant c'est que l'accès par console est plus sécurisé que l'accès à distance via le Telnet car il faut être relié directement au commutateur. Mais le seule inconvénient est le déplacement, vue qu'il faut être présent dans le lieu où est placé le commutateur. Ce qui n'est pas toujours possible pour un administrateur.

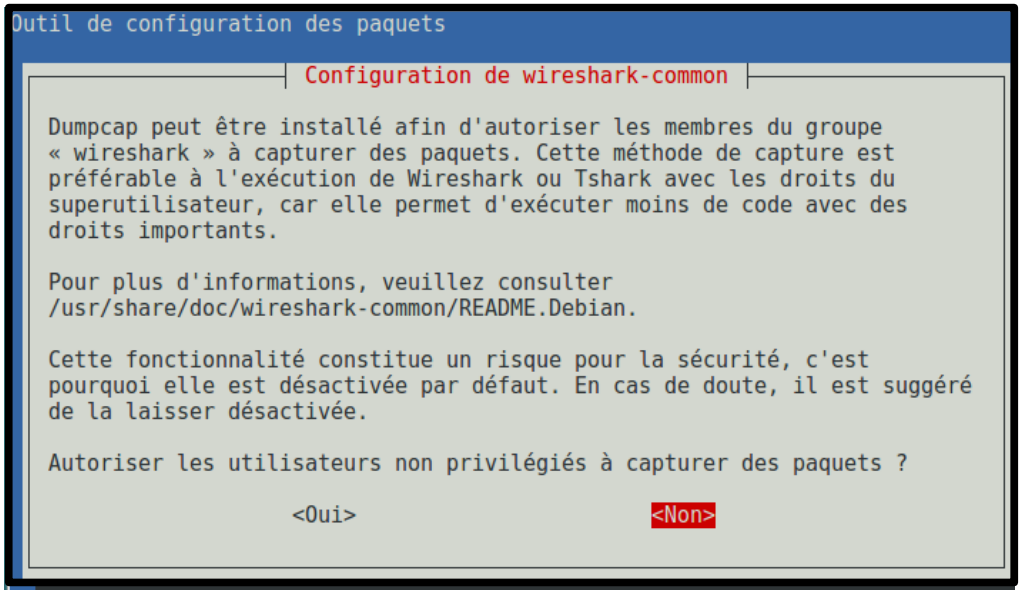
À présent on va administrer le poste A (sous Linux) pour pouvoir appliquer nos teste.

On lui attribut donc son adresse IP par la syntaxe shell : **`nano /etc/network/interfaces`**

```
auto eth0
iface eth0 inet static
        address 172.16.68.115
        netmask 255.255.0.0
        gateway 172.16.253.253
```

Sur notre poste, on doit installer un utilitaire très pratique appelé Wireshark qui va nous permettre d'analyser les trafic du réseau.

On utilise pour cela la syntaxe shell : **`apt-get install wireshark`**



après avoir installer l'utilitaire en question, nous isolons notre poste A pour éviter tout conflit d'adresse IP.

Ensuite, on administre notre poste A pour qu'il soit conforme au plan d'adressage ainsi qu'au schéma.

On commence par lui attribuer sont nom d'hôte via la syntaxe : **`gedit /etc/hostname`**

Par la suite on définit son adressage en rapport avec le schéma, en passant par la même syntaxe que tout à l'heure, à contrario que l'éditeur de texte ne sera pas nano !

Ouvrir *interfaces /etc/network

```
# This file describes the network interfaces availab
# and how to activate them. For more information, se

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
        address 172.32.69.2
        netmask 255.255.0.0
        gateway 172.32.69.254
```

On peut toujours vérifier, **après avoir appliquer un redémarrage de l'interface réseau**, que notre poste à bien reçut son adressage :

```
root@AdministrateurA:/home/administrateur# ifdown eth0
RTNETLINK answers: No such process
root@AdministrateurA:/home/administrateur# ifup eth0
root@AdministrateurA:/home/administrateur# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:27:a1:98:40
          inet adr:172.32.69.2  Bcast:172.32.255.255  Masque:255.255.0.0
          adr inet6: fe80::a00:27ff:fea1:9840/64 Scope:Lien
```

On peut toute fois tester la liaison entre le commutateur et notre poste.

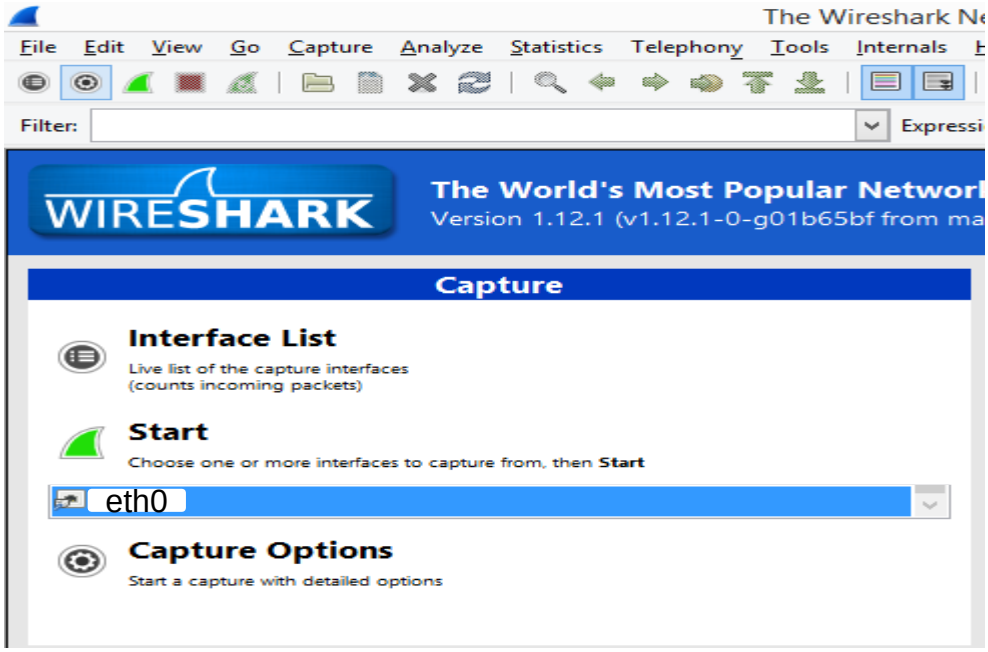
Requête ping à partir du poste A vers le commutateurA :

```
root@AdministrateurA:/home/administrateur# ping 172.32.69.253
PING 172.32.69.253 (172.32.69.253) 56(84) bytes of data.
64 bytes from 172.32.69.253: icmp_seq=1 ttl=255 time=5.49 ms
64 bytes from 172.32.69.253: icmp_seq=2 ttl=255 time=2.90 ms
64 bytes from 172.32.69.253: icmp_seq=3 ttl=255 time=1.67 ms
64 bytes from 172.32.69.253: icmp_seq=4 ttl=255 time=1.51 ms
```

Ensuite, on démarre Wiresharsk via le terminal en mode administrateur :

```
root@AdministrateurA:/home/administrateur# wireshark
Gtk-Message: GtkDialog mapped without a transient parent. This is discouraged.
```

et on lance une capture de trame à partir de l'interface eth0 (donc celle utilisé)



En parallèle on établit une connexion sur le terminal via le telnet :

```
root@AdministrateurA:/home/administrateur# telnet 172.32.69.253
Trying 172.32.69.253...
Connected to 172.32.69.253.
Escape character is '^]'.


User Access Verification

Password:
CommutateurA>ena
CommutateurA>enable
Password:
CommutateurA#
```

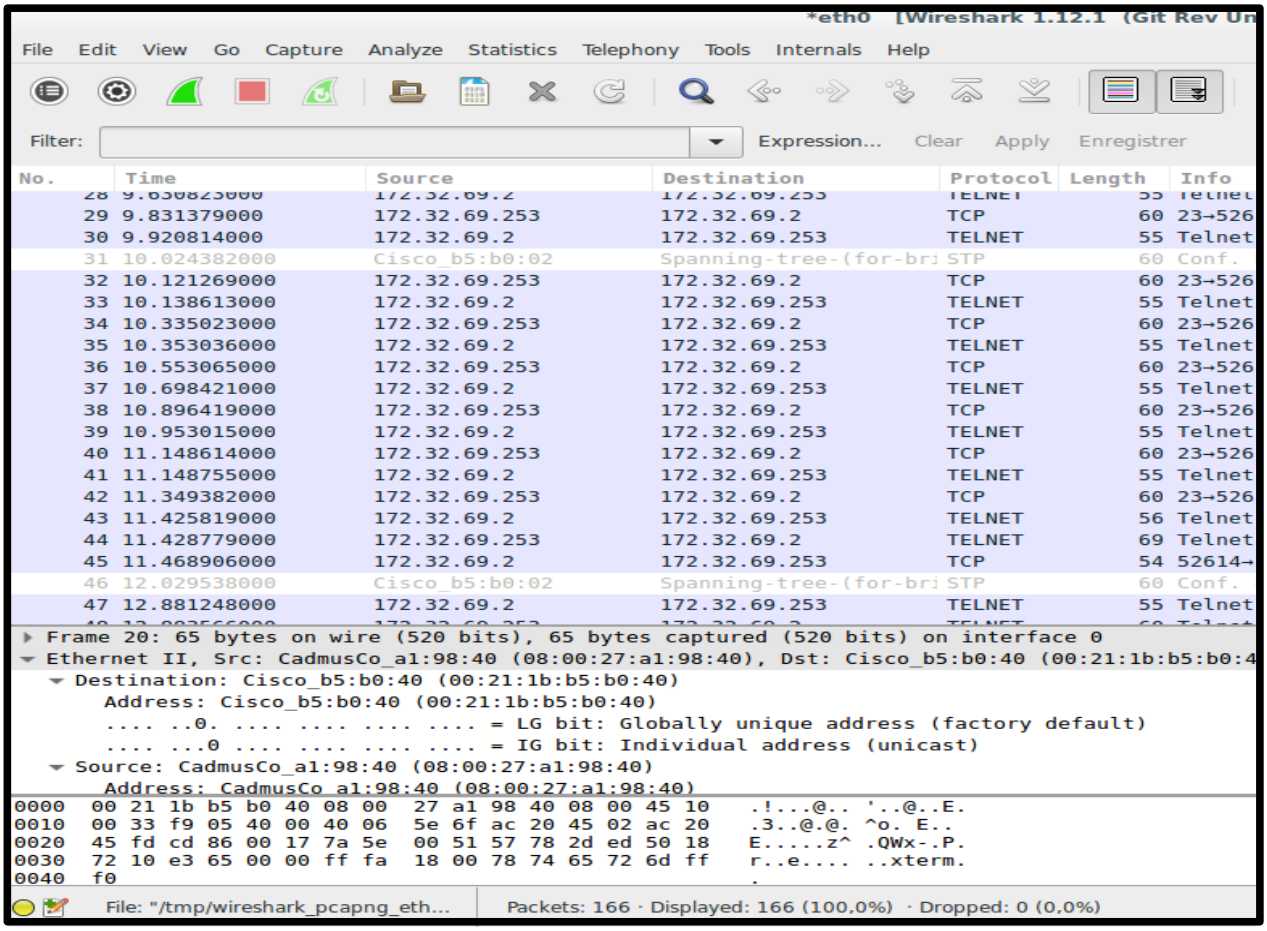
On rentre le mot de passe d'authentification pour l'accès à distance puis le mot de passe du mode privilégié.

Une fois connecté sur le commutateurA, on utilise la commande show pour voir les configurations du commutateurA.

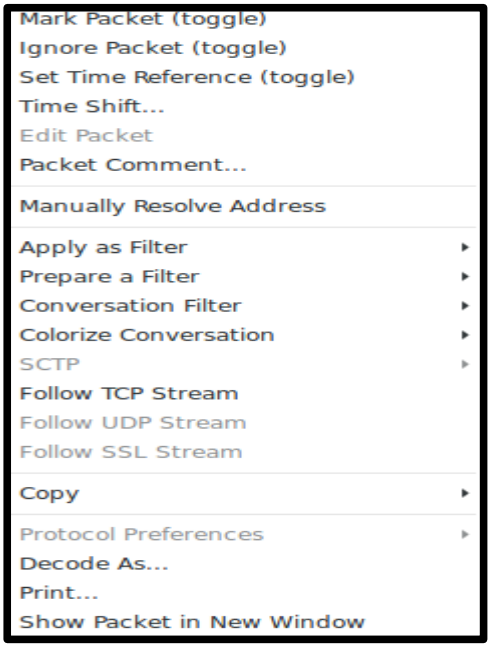
```
CommutateurA#show running-config
Building configuration...

Current configuration : 1381 bytes
!
version 12.2
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname CommutateurA
!
```

Après cela, on ferme la session Telnet et en parallèle en arrête la capture.

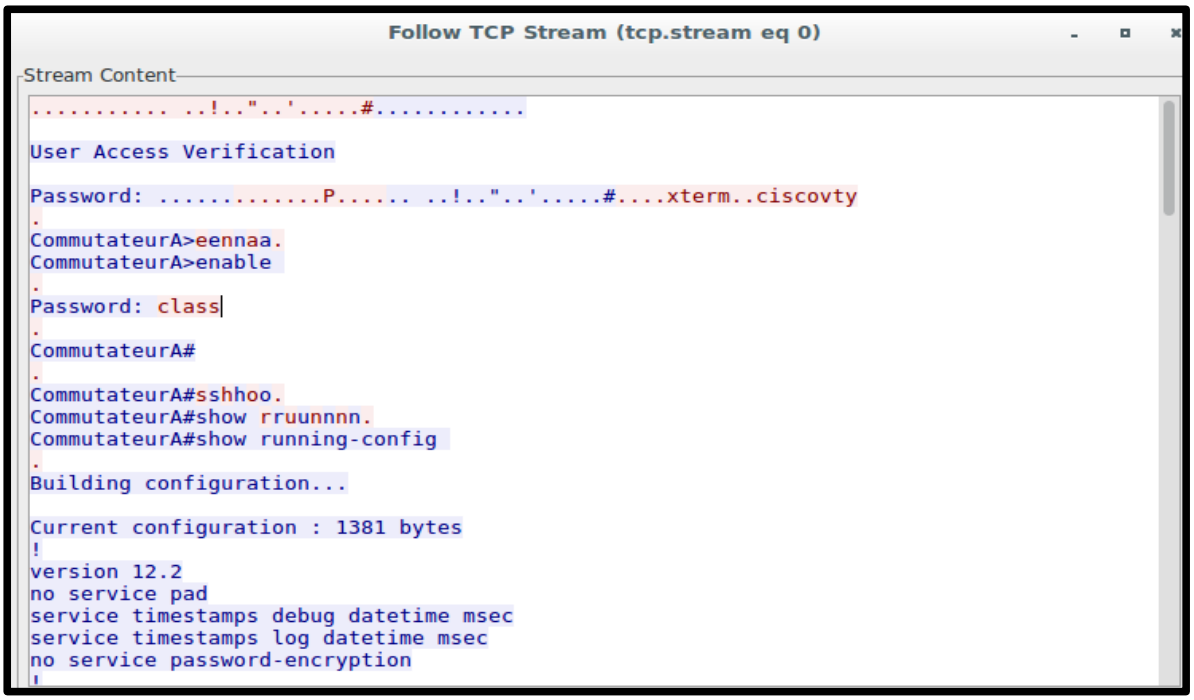
Sur Wireshark on peut apercevoir les trames capturées. Cependant on a pas exactement l'information qui nous intéresse c'est à dire le mot de passe.
En effet, car un hacker va s'intéresser aux données sensible.



Donc on utilise une **fonctionnalité intéressante** que propose Wireshark pour avoir les informations qui nous intéresses. On réalise un clique droit sur l'interface de Wireshark puis on clique sur "**Follow TCP Stream**" :



Wireshark nous ouvre après cela une fenêtre qui contient toute la capture :



On constate que le mot de passe est bien compréhensible par Intel que le mode privilégier est crypté.

C'est pourquoi, on va crypter tous les mots de passe saisies dans le commutateur avec une commande Cisco qui applique un cryptage simple sur le fichier local de configuration du commutateur.

Donc vue que l'idée de protection est évoquée, on va simplement protéger ce fichier local en question.

```
#On applique le cryptage simple
CommutateurA(config)#service password-encryption
CommutateurA(config)#exit

#Et on analyse cette situation
CommutateurA#show running-config
!
line con 0
 password 7 045802150C2E4F4107
 login

line vty 0 4
 password 7 121A0C0411041A1033
 login

line vty 5 15

 password 7 121A0C0411041A1033

 login

!

end
```

On voit belle bien que nos mots de passe ont étaient cryptés localement.

Sécurisation des accès au commutateur

Comme précédemment on avait appliqué des protections d'accès à nos mots de passe.
Maintenant on va se centraliser sur la sécurisation à distance.

Pour cela on va activer le service SSH sur le commutateur.

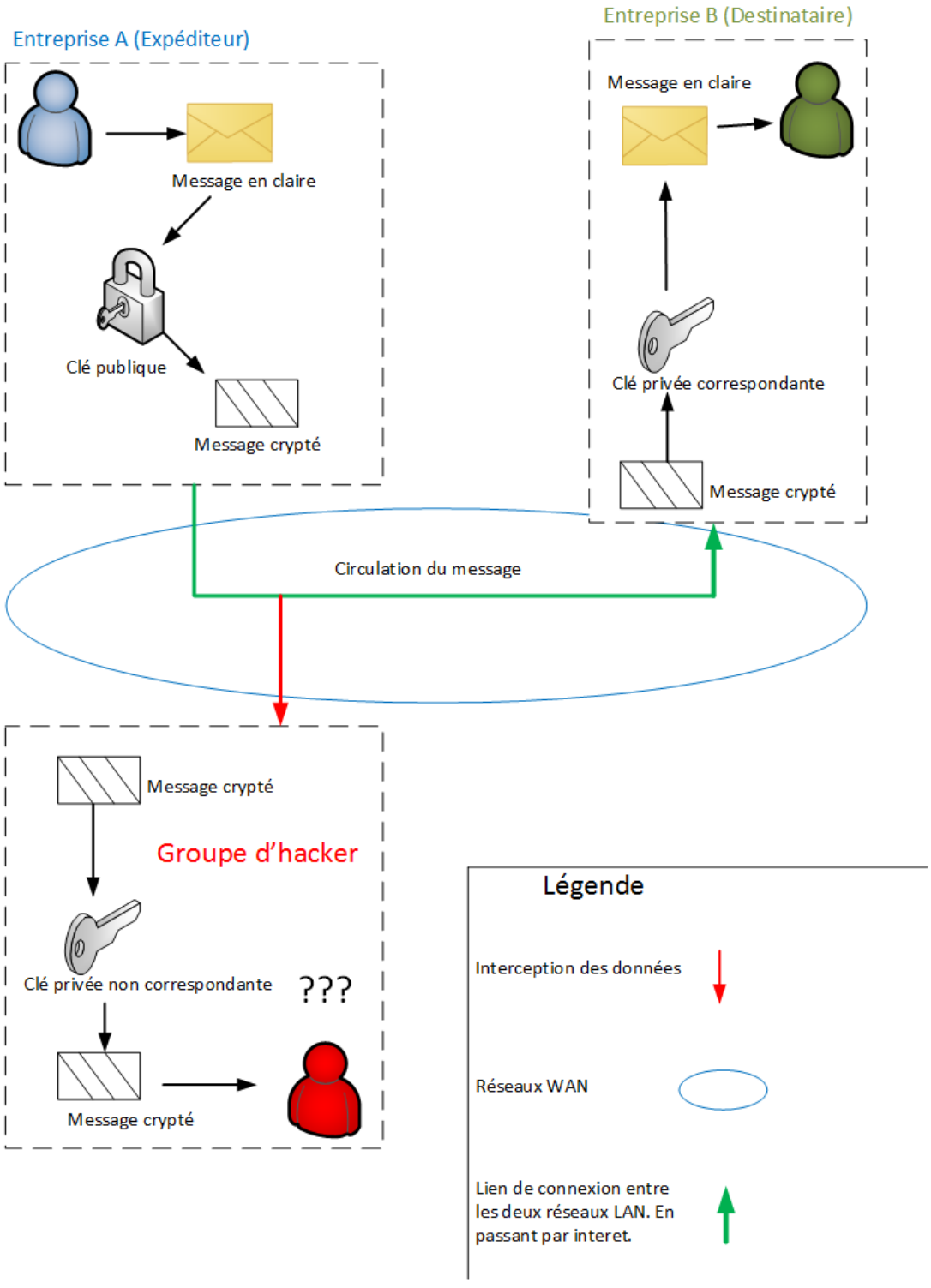
```
#On commence par définir un nom de domaine
CommutateurA(config)#ip domain-name sio2g3.local

#On génère une cryptographie (elle est nécessaire pour la génération des
#clés de cryptages utilisées par SSH)
CommutateurA(config)#crypto key generate rsa

The name for the keys will be: CommutateurA.sio2g3.local
Choose the size of the key modulus in the range of 360 to 2048 for your
 General Purpose Keys. Choosing a key modulus greater than 512 may take
 a few minutes.

How many bits in the modulus [512]: 768
% Generating 768 bit RSA keys, keys will be non-exportable...[OK]

*Mar  1 19:37:55.656: %SSH-5-ENABLED: SSH 1.99 has been enabled

#On fixe le délai d'expiration en cas d'inactivité
CommutateurA(config)#ip ssh time-out 60

#On fixe le nombre d'autorisation de tentative de connexion
CommutateurA(config)#ip ssh authentication-retries 2
```

RSA étant une méthode de cryptage à clé asymétrique ayant 512, 768, 1024, bits ou plus comme longueur des clés utilisées.
La notion d'asymétrie qui est appliquée pour la RSA, repose sur le principe d'utilisation de clés publique / privée. La clé publique ''diffusée'' permet à l'expéditeur d'un message de le chiffrer par conséquent, la clé privée permet de déchiffrer ce message en question.

C'est pourquoi il est important de retenir que la clé de chiffrage publique est à sens unique c'est à dire qu'elle ne peut permet que de crypter alors qu'une clé privée permet le décodage à tous ceux qui possède la clé publique correspondante.

Le schéma suivant va nous permettre de mieux comprendre ce système de chiffrement asymétrique.

Méthode de cryptage RSA



On peut comprendre sur le schéma, que l'entreprise A qui envoie son message à l'entreprise B, possède une clé privée. Et donc l'entreprise B possédant une clé publique a put lire le contenue du message puisque sa clé publique correspond bien à la clé privée de l'entreprise A.

Ce qui est très intéressant de comprendre également concernant le groupe de hacker malgré qu'ils ont en possession les données sensibles récupérer. Ces derniers n'ont pas pu décrypter le message et donc impossible de savoir son contenu !

Ensuite on va limiter l'accès via les lignes VTY en SSH :

```
CommutateurA(config)#line vty 0 15
CommutateurA(config-line)#transport input ssh
CommutateurA(config-line)#exit
```

Puis on retourne sur la station d'administration A, et on essaye de ce connecter en Telnet :

```
root@AdministrateurA:/home/administrateur# telnet 172.32.69.253
Trying 172.32.69.253...
Connected to 172.32.69.253.
```

*Normalement après la tentative de connexion en Telnet, on aura une réponse de la part du commutateur de ce type : "**Connection to 172.32.69.253 closed by foreign host**". On comprend que seule la connexion en SSH est connue et donc tout autre type de connexion sera refusé.*

On va dès à présent créer un utilisateur par défaut pour le commutateur.

```
#On définit par défaut notre utilisateur que l'on va créer
CommutateurA(config)#aaa new-model

#On passe par cette syntaxe cisco pour créer notre utilisateur
CommutateurA(config)#username admin password 0 cisco
CommutateurA(config)#exit
```

On va tester le fonctionnement du SSH sur le commutateur. Donc tout d'abord on lance une capture sur Wireshar à partir du poste administrateur A.

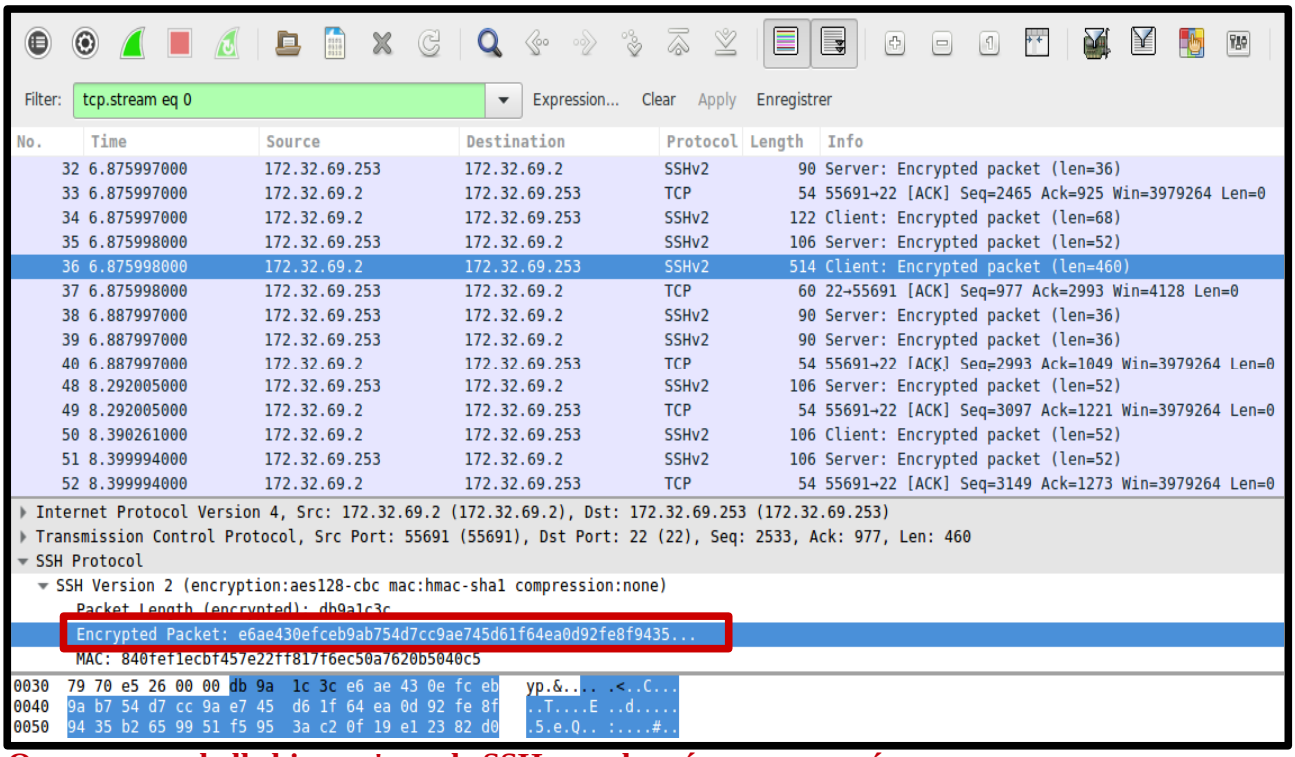
Puis on se connecte sur le port 22 (SSH) au commutateur :

```
root@AdministrateurA:/home/administrateur# ssh admin@172.32.69.253
Password:

CommutateurA>ena
```

On valide notre certificat (permet de s'assurer que le correspondant est bien celui qu'il prétend être) puisque vue que c'est notre commutateur, il n'y a pas de danger liées à l'usurpation.

```
The authenticity of host '172.32.69.253 (172.32.69.253)' can't be established.
RSA key fingerprint is 0e:77:08:75:68:08:33:a5:02:77:fb:16:ae:30:56:a1.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.32.69.253' (RSA) to the list of known hosts.
Password:
```

Après avoir saisie le mot de passe, on applique la syntaxe Cisco : show running-configuration.
Puis on arrête la capture de trame et on ferme la session SSH. On analyse ensuite la trame capturée :



On remarque belle bien qu'avec le SSH, nos données sont cryptées.

Sécurisation des accès au routeur

On va maintenant se concentrer sur le routeur, donc on va faire exactement la même chose que sur le commutateur.

On commence par établir les configurations de base, donc le mot de passe et le nom d'hôte :

```
Router>enable
Router#config
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.

Router(config)#hostname RouteurA

RouteurA(config)#line con 0
RouteurA(config-line)#password ciscocon
RouteurA(config-line)#login
RouteurA(config)#enable secret class


RouteurA# show running-config

enable secret 5 $1$mfYz$NCaEKY2nJyhLOjHgqXLO1/

line con 0
 password ciscocon
 login
line aux 0
line vty 0 4
 login
…

#On crypte les mots de passe du fichier de configuration comme pour le
#commutateur
RouteurA(config)#service password-encryption
```

Ensuite on créer l'utilisateur admin tout comme on a fait pour le commutateur :

```
RouteurA(config)#aaa new-model
RouteurA(config)#username admin password 0 cisco
```

On définit un nom de domaine aussi pour le routeur :

```
RouteurA(config)#ip domain-name sio2g3.local
RouteurA(config)#crypto key generate rsa

The name for the keys will be: RouteurA.sio2g6.local

Choose the size of the key modulus in the range of 360 to 2048 for your

  General Purpose Keys. Choosing a key modulus greater than 512 may take

  a few minutes.

How many bits in the modulus [512]: 1024

% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]
```

On peut constater que la langueur de la clé par la méthode RAS, est de 1024 bits.

On limite également sur le routeur l'accès via les lignes VTY en SSH :

```
RouteurA(config)#line vty 0 4
RouteurA(config-line)#transport input ssh
```

Après avoir testé ,à partir de la station A, la liaison sur le routeur. On se connecte sur ce dernier via le port 23 (Telnet) :

```
root@ServeurA:/home/administrateur# telnet 172.32.69.254
Trying 172.32.69.254...
telnet: Unable to connect to remote host: Connection refused
root@ServeurA:/home/administrateur#
```

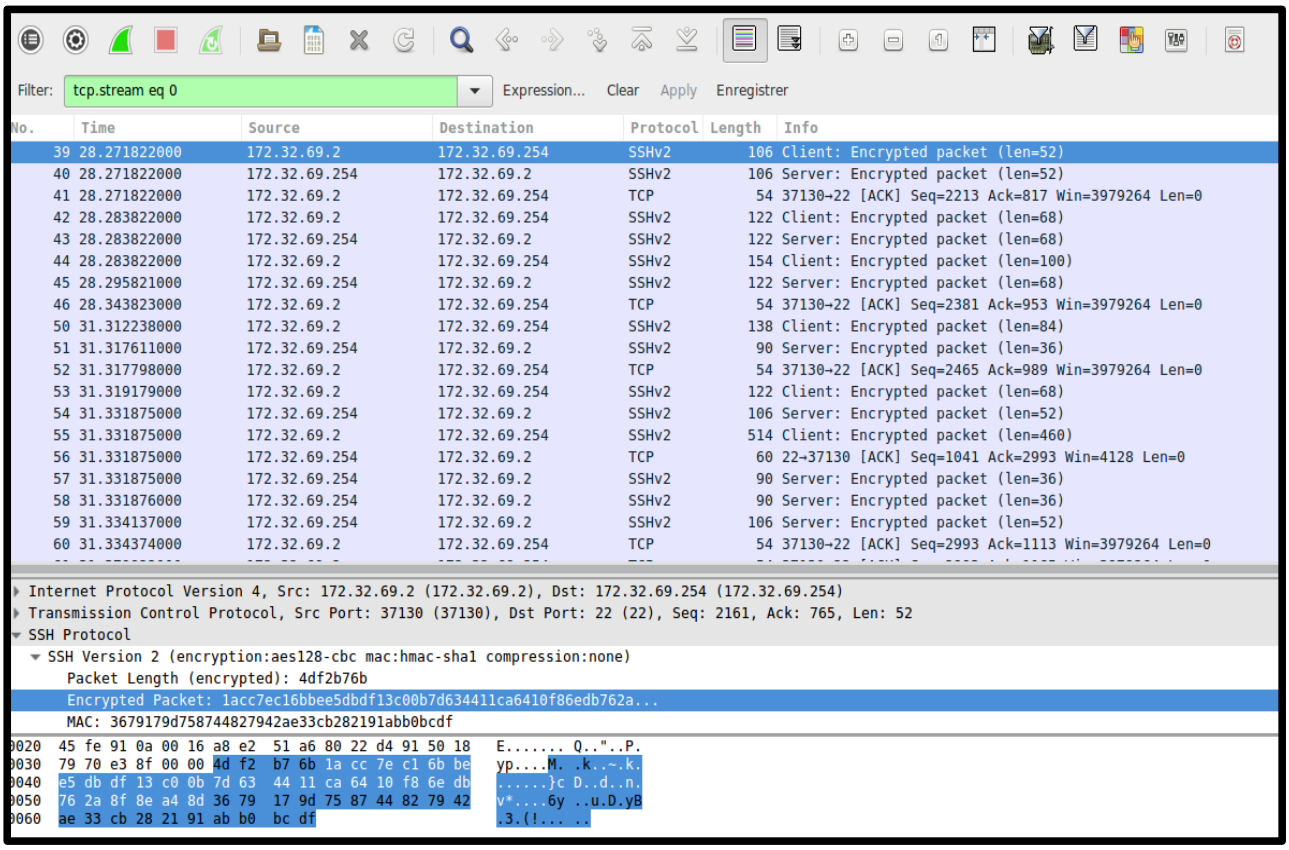
On a une réponse similaire au commutateur et donc ce type de connexion est refusé.

On lance une capture de trame sur le poste administrateur A. Et en parallèle on se connecte avec le compte admin en SSH sur le routeur puis on saisis le mot de passe:

```
root@AdministrateurA:/home/administrateur# ssh admin@172.32.69.254
The authenticity of host '172.32.69.254 (172.32.69.254)' can't be established.
RSA key fingerprint is 0e:e7:08:15:68:08:33:a5:02:48:fb:16:ae:30:56:a1.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.32.69.254' (RSA) to the list of known hosts.
Password:

RouteurA>ena
RouteurA>enable
Password:
RouteurA#sho
RouteurA#show runn
```

On applique la syntaxe show running-config puis on ferme la session et on arrête la capture de trame.

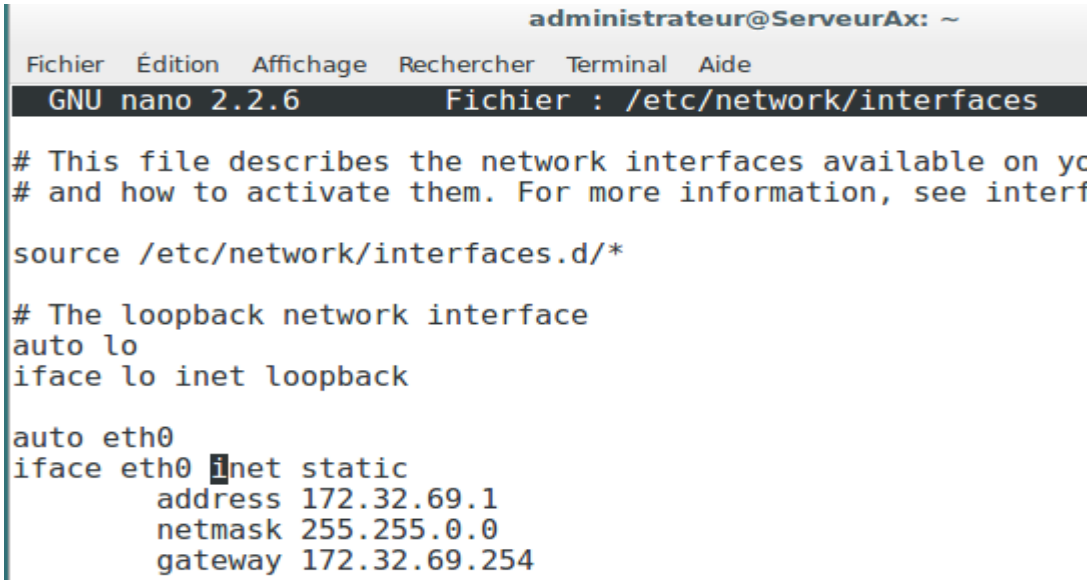
Ensuite, lorsqu'on analyse cette trame, on aperçoit tout comme dans le commutateur on a bien des trames possédant un cryptage.



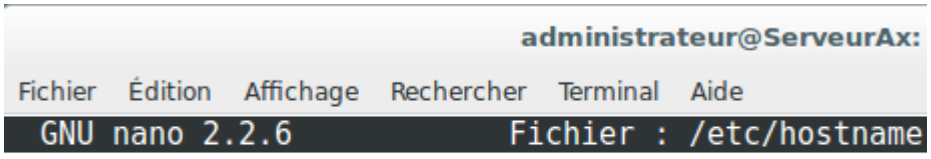
Partie 2 – Mise en place d'un accès sécurisé sur un serveur Linux

Mise en place du service sur le serveur

On mets les paramètres réseau du serveur, on lui donne donc une adresse ip, un masque et une passerelle :



On change son nom d'hôte dans le « /etc/hostname ».



On test la connectivité de notre réseau en faisant un ping de l'administrateurA au serveur. On a bien une réponse c'est que tout est bon.

```
root@AdministrateurA:/home/administrateur# ping 172.32.69.1
PING 172.32.69.1 (172.32.69.1) 56(84) bytes of data.
64 bytes from 172.32.69.1: icmp_seq=1 ttl=64 time=0.000 ms
64 bytes from 172.32.69.1: icmp_seq=2 ttl=64 time=0.000 ms
64 bytes from 172.32.69.1: icmp_seq=3 ttl=64 time=0.000 ms
^C
--- 172.32.69.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2027ms
rtt min/avg/max/mdev = 0.000/0.000/0.000/0.000 ms
root@AdministrateurA:/home/administrateur#
```

On utilise la commande « dkpg -l openssh-server » afin de vérifier la présence du paquet openssh-server. On vois qu'il n'y a aucun paquet de ce nom là, on dois donc l'installer avec la commande « apt-get install openssh-server ».

```
root@debian:/home/administrateur# dpkg -l openssh-server
dpkg-query: aucun paquet ne correspond à openssh-server
root@debian:/home/administrateur# apt-get install openssh-server
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  openssh-sftp-server
Paquets suggérés :
  ssh-askpass rssh molly-guard ufw monkeysphere
Les NOUVEAUX paquets suivants seront installés :
  openssh-server openssh-sftp-server
0 mis à jour, 2 nouvellement installés, 0 à enlever et 130 non mis à jour.
Il est nécessaire de prendre 412 ko dans les archives.
Après cette opération, 1 121 ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n]
```

Une fois le paquet installé on vérifie qu'il est bien en train de fonctionner avec la commande « service ssh status ».Il y a marqué « active (running) » ce qui veut littéralement dire, activé(en fonction).

```
administrateur@ServeurA: ~
Fichier  Édition  Affichage  Rechercher  Terminal  Aide

root@ServeurA:/home/administrateur# service ssh status
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled)
   Active: active (running) since jeu. 2015-11-12 14:30:10 CET; 1min 29s ago
 Main PID: 424 (sshd)
   CGroup: /system.slice/ssh.service
           └─424 /usr/sbin/sshd -D

nov. 12 14:30:11 ServeurA sshd[424]: Server listening on 0.0.0.0 port 22.
nov. 12 14:30:11 ServeurA sshd[424]: Server listening on :: port 22.
```

Sécurisation des accès à partir d'un poste Linux

On a déjà testé la liaison entre L'administrateurA et le serveurA auparavant, maintenant que nous savons déjà que la liaison fonctionne on lance une capture sur wireshark qui nous permettra d'examiner les trames, ensuite on se connecte en ssh sur le serveur à partir du port 22.

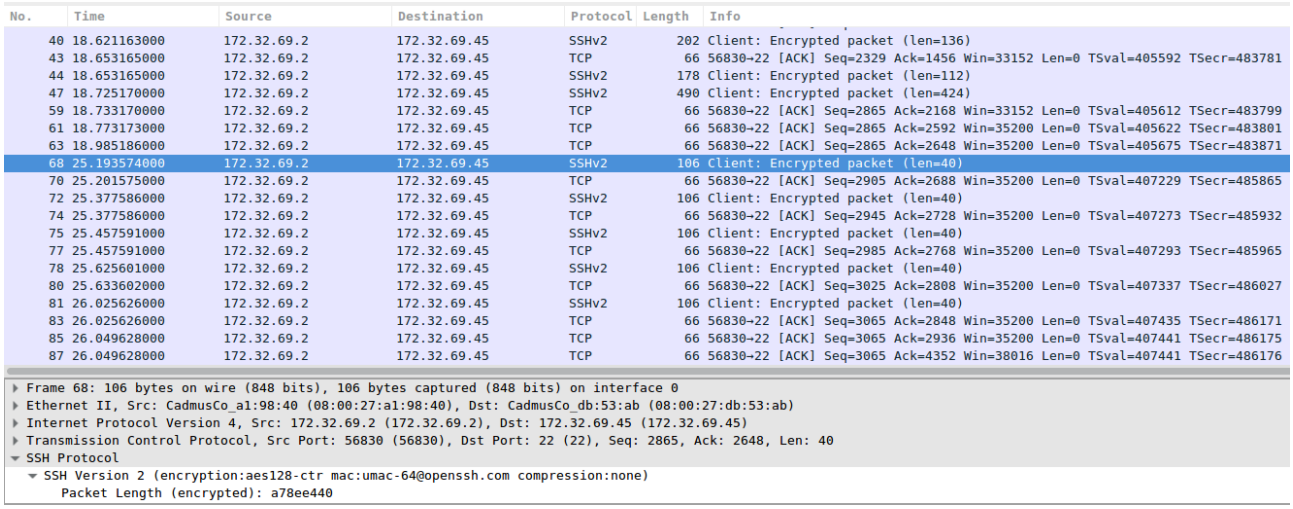
On aperçoit le certificat et donc on valide avec yes :

```
root@AdministrateurA:/home/administrateur# ssh 172.32.69.1
The authenticity of host '172.32.69.1 (172.32.69.1)' can't be established.
ECDSA key fingerprint is 72:e7:0c:ca:4e:bc:43:67:c9:07:97:a8:9f:c6:03:72.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.32.69.1' (ECDSA) to the list of known hosts.
root@172.32.69.1's password:
```

On saisis le mot de passe root du serveur afin de se connecter en ssh, on utilise la commande « ifconfig » afin de vérifier l'adresse ip du serveur auquel on est connecté, il s'agit bien de lui car l'adresse ip est : 172.32.69.45 .

```
administrateur@AdministrateurA: ~
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
root@AdministrateurA:/home/administrateur# ssh 172.32.69.45
root@172.32.69.45's password:
Linux ProxSYLAB 3.2.0-4-486 #1 Debian 3.2.57-3+deb7u1 i686

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov 12 16:26:25 2015 from administrateura.local
root@ProxSYLAB:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:db:53:ab
          inet adr:172.32.69.45  Bcast:172.32.255.255  Masque:255.255.0.0
          adr inet6: fe80::a00:27ff:fedb:53ab/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:353 errors:0 dropped:0 overruns:0 frame:0
          TX packets:393 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:51832 (50.6 KiB)  TX bytes:52725 (51.4 KiB)
```

Une fois cette étape terminé on arrête la capture de trames et on commence à les examiner.
On voit que dans les échanges entre les deux machines les paquets sont « Encrypted » donc crypté.



No.	Time	Source	Destination	Protocol	Length	Info
40	18.621163000	172.32.69.2	172.32.69.45	SSHv2	202	Client: Encrypted packet (len=136)
43	18.653165000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=2329 Ack=1456 Win=33152 Len=0 TSval=405592 TSecr=483781
44	18.653165000	172.32.69.2	172.32.69.45	SSHv2	178	Client: Encrypted packet (len=112)
47	18.725170000	172.32.69.2	172.32.69.45	SSHv2	490	Client: Encrypted packet (len=424)
59	18.733170000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=2865 Ack=2168 Win=33152 Len=0 TSval=405612 TSecr=483799
61	18.773173000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=2865 Ack=2592 Win=35200 Len=0 TSval=405622 TSecr=483801
63	18.985186000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=2865 Ack=2648 Win=35200 Len=0 TSval=405675 TSecr=483871
68	25.193574000	172.32.69.2	172.32.69.45	SSHv2	106	Client: Encrypted packet (len=40)
70	25.201575000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=2905 Ack=2688 Win=35200 Len=0 TSval=407229 TSecr=485865
72	25.377586000	172.32.69.2	172.32.69.45	SSHv2	106	Client: Encrypted packet (len=40)
74	25.377586000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=2945 Ack=2728 Win=35200 Len=0 TSval=407273 TSecr=485932
75	25.457591000	172.32.69.2	172.32.69.45	SSHv2	106	Client: Encrypted packet (len=40)
77	25.457591000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=2985 Ack=2768 Win=35200 Len=0 TSval=407293 TSecr=485965
78	25.625601000	172.32.69.2	172.32.69.45	SSHv2	106	Client: Encrypted packet (len=40)
80	25.633602000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=3025 Ack=2808 Win=35200 Len=0 TSval=407337 TSecr=486027
81	26.025626000	172.32.69.2	172.32.69.45	SSHv2	106	Client: Encrypted packet (len=40)
83	26.025626000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=3065 Ack=2848 Win=35200 Len=0 TSval=407435 TSecr=486171
85	26.049628000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=3065 Ack=2936 Win=35200 Len=0 TSval=407441 TSecr=486175
87	26.049628000	172.32.69.2	172.32.69.45	TCP	66	56830→22 [ACK] Seq=3065 Ack=4352 Win=38016 Len=0 TSval=407441 TSecr=486176

```
▸ Frame 68: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface 0
▸ Ethernet II, Src: CadmusCo_a1:98:40 (08:00:27:a1:98:40), Dst: CadmusCo_db:53:ab (08:00:27:db:53:ab)
▸ Internet Protocol Version 4, Src: 172.32.69.2 (172.32.69.2), Dst: 172.32.69.45 (172.32.69.45)
▸ Transmission Control Protocol, Src Port: 56830 (56830), Dst Port: 22 (22), Seq: 2865, Ack: 2648, Len: 40
▾ SSH Protocol
  ▾ SSH Version 2 (encryption:aes128-ctr mac:umac-64@openssh.com compression:none)
      Packet Length (encrypted): a78ee440
```

On affiche ensuite la clé publique du serveur en saisissant la commande « cat /root/.ssh/know_hosts » afin de voir les hôtes connus. Si cette clé est changé l'administrateurA ne se connectera pas au serveurA.

```
root@ProxSYLAB:~# cat /root/.ssh/known_hosts
|1|ePFOZxjxQ15qh587IBgr12XRsJs=|GdJfKZ/z6mUKU6hBGJvKeiTox94= ecdsa-sha2-nistp256
 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBA46Qmx/SrBluEx5BlyQF9k+At8
O2KFk9VH0kCX7LI7p55DR9VXrP5fFkMJfoA1Csom4AYrPPb8rff/cdTpNIl0=
```

On va maintenant crypter notre clé grâce à une passphrase qui est une phrase de passe qui va servir à renforcer la sécurité du cryptage de la clé privée. On génère ensuite une paire de clé publique / privée avec la commande « ssh-keygen -t rsa ». La passphrase est : coursdesisr4btssio

```
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
root@AdministrateurA:/home/administrateur# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match.  Try again.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
ca:b6:a9:b0:4d:fe:1e:00:58:14:38:bd:a3:d5:c2:70 root@AdministrateurA
The key's randomart image is:
+---[RSA 2048]----+
|  ++.            |
|+oE              |
|.=.o             |
|   *..           |
| o o.    S       |
|.     o .        |
|   . . =         |
|    * . +        |
|   . ++=         |
+-----------------+
```

On liste le contenu du dossier caché .ssh en saisissant la commande « ls -l /root/.ssh/ ». On constate la présence d'une clé privée « id_rsa » et d'une clé publique « id_rsa.pub ». On va envoyé la clé publique au serveur par SSH elle peut aussi être transmise par clé USB ou par voie électronique mais avec des risques.

```
root@AdministrateurA:/home/administrateur# ls -l /root/.ssh/
total 12
-rw------- 1 root root 1766 nov.  12 15:22 id_rsa
-rw-r--r-- 1 root root  402 nov.  12 15:22 id_rsa.pub
-rw-r--r-- 1 root root  948 nov.  12 15:09 known_hosts
root@AdministrateurA:/home/administrateur#
```

On envoie la clé publique par SSH au serveurA avec la commande « ssh-copy-id -i /root/.ssh/id_rsa.pub ».

```
root@AdministrateurA:/home/administrateur# ssh-copy-id -i /root/.ssh/id_rsa.pub
172.32.69.45
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt
ed now it is to install the new keys
root@172.32.69.45's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh '172.32.69.45'"
and check to make sure that only the key(s) you wanted were added.
```

On se connecte ensuite au serveurA par SSH, la passphrase nous ait demandé, on rentre donc celle que l'on a choisit (coursdesisr4btssio). On s'est connecté sans avoir à saisir le mot de passe root du serveurA, la passphrase permet un niveau de sécurité supplémentaire.

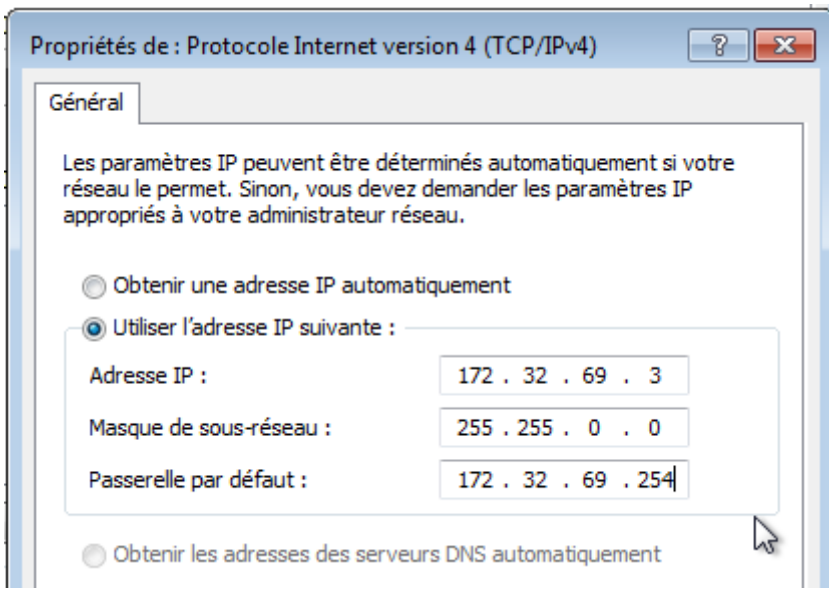
```
root@AdministrateurA:/home/administrateur# ssh 172.32.69.45
Enter passphrase for key '/root/.ssh/id_rsa':
Linux ProxSYLAB 3.2.0-4-486 #1 Debian 3.2.57-3+deb7u1 i686

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov 12 16:48:00 2015 from administrateura.local
```

Une autre méthode connexion avec passphrase est possible en saisissant la commande « ssh-add » et on rentre directement notre passphrase, avec cet méthode aucun mot de passe ne nous est demandé on se connecte directement au serveurA.

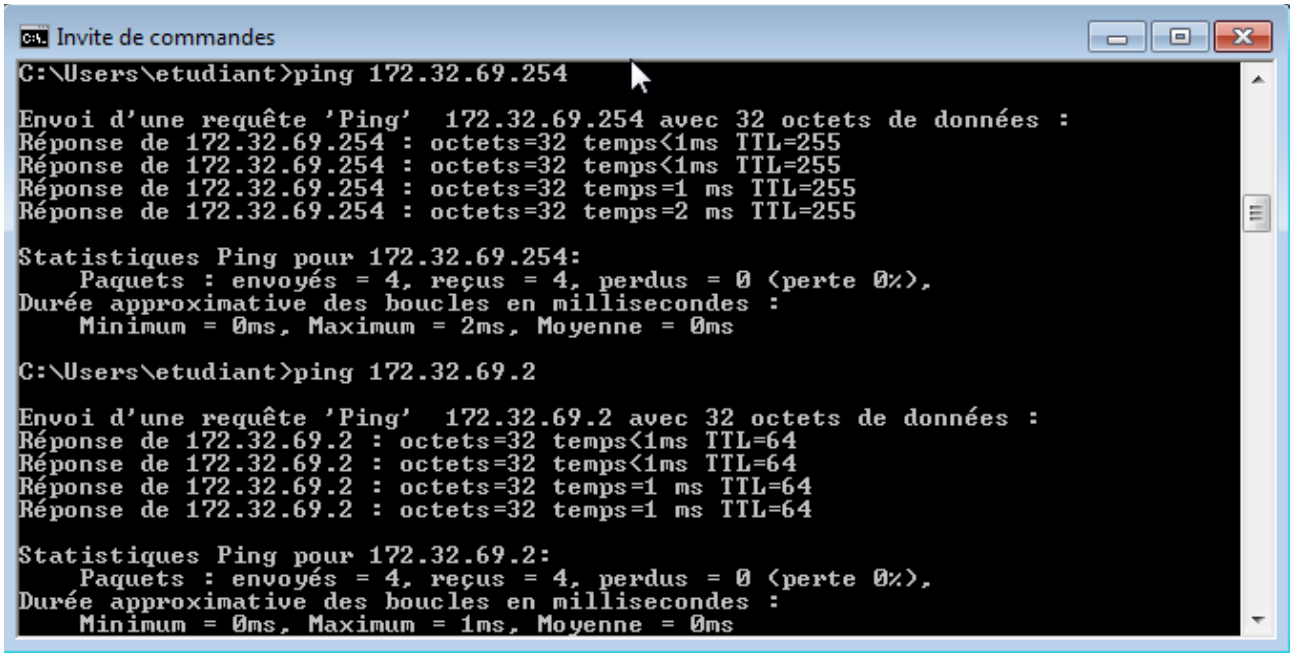
```
administrateur@AdministrateurA: ~
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
root@AdministrateurA:/home/administrateur# ssh-add
Enter passphrase for /root/.ssh/id_rsa:
Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)
root@AdministrateurA:/home/administrateur# ssh 172.32.69.45
Linux ProxSYLAB 3.2.0-4-486 #1 Debian 3.2.57-3+deb7u1 i686

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov 12 16:48:56 2015 from administrateura.local
root@ProxSYLAB:~#
```

Sécurisation des accès à partir d'un poste Windows

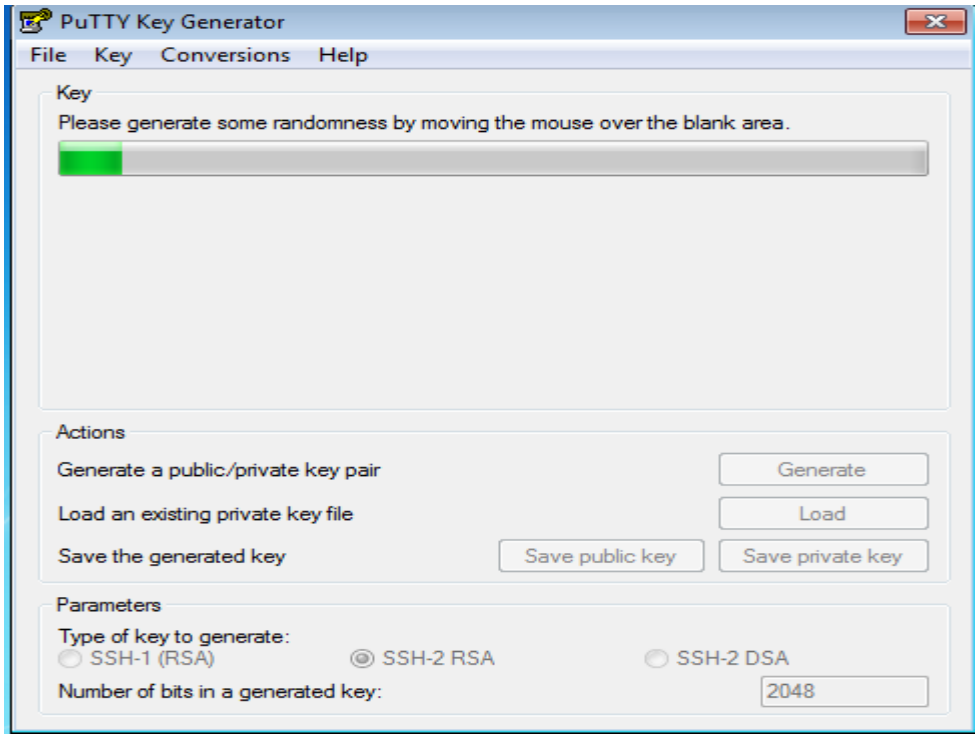
On paramètres la station AdministrateurB en lui attribuant son adresse IP afin qu'il puisse communiquer avec le serveurA.



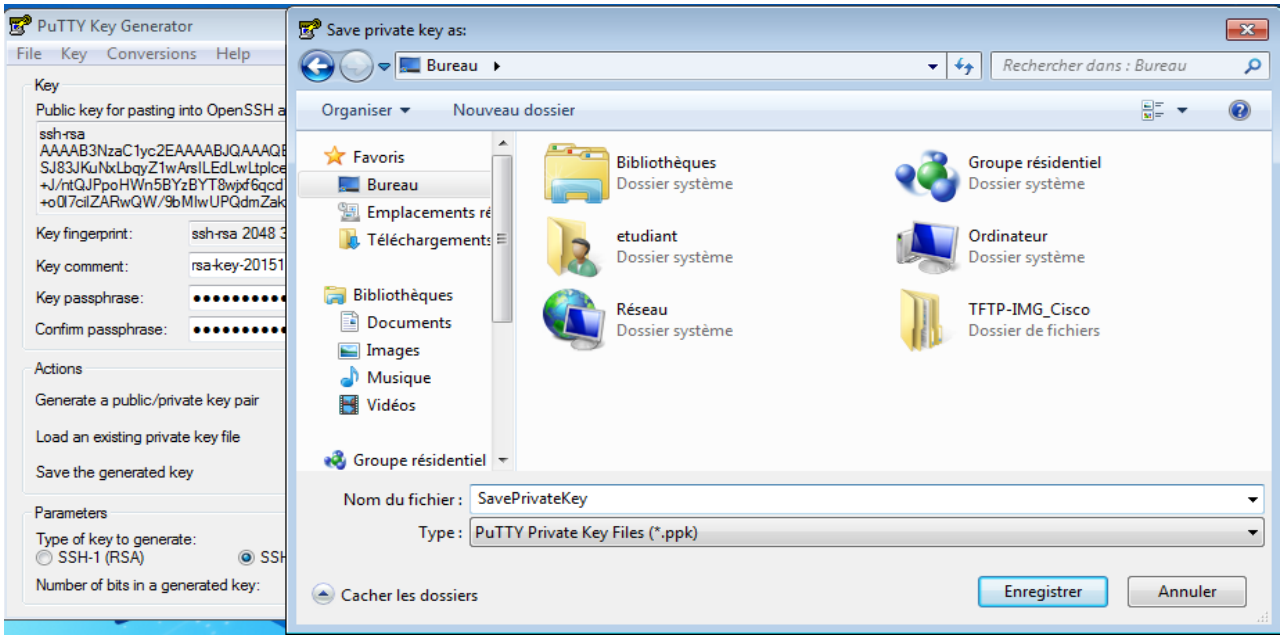
On test la connectivité avec le routeur (172.32.69.254) et le serveur (172.32.69.2) avec la commande ping.



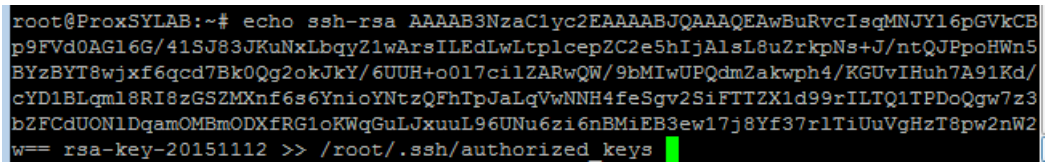
On lance puttygen et on génère une paire de clés publique et privée en cliquant sur le bouton « Generate ».



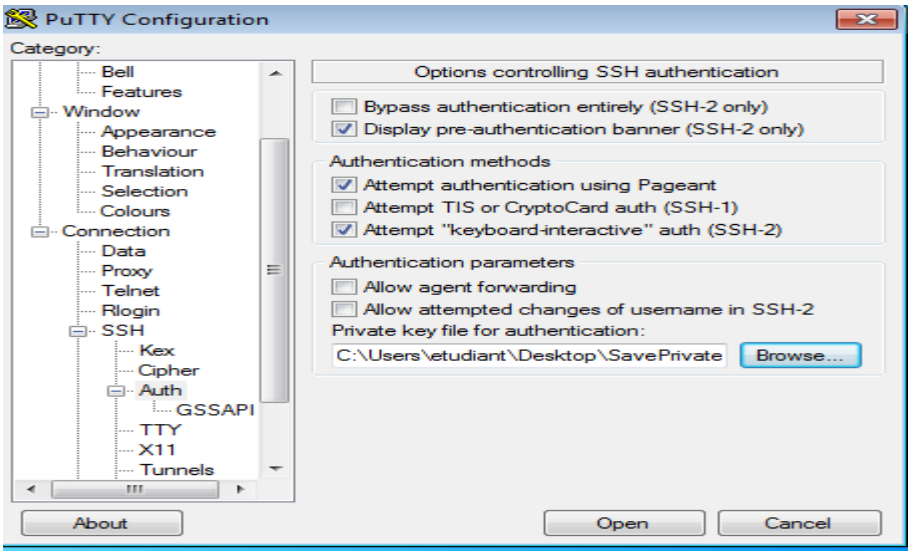
On ajoute la passphrase et on enregistre la clé privée sur le bureau.



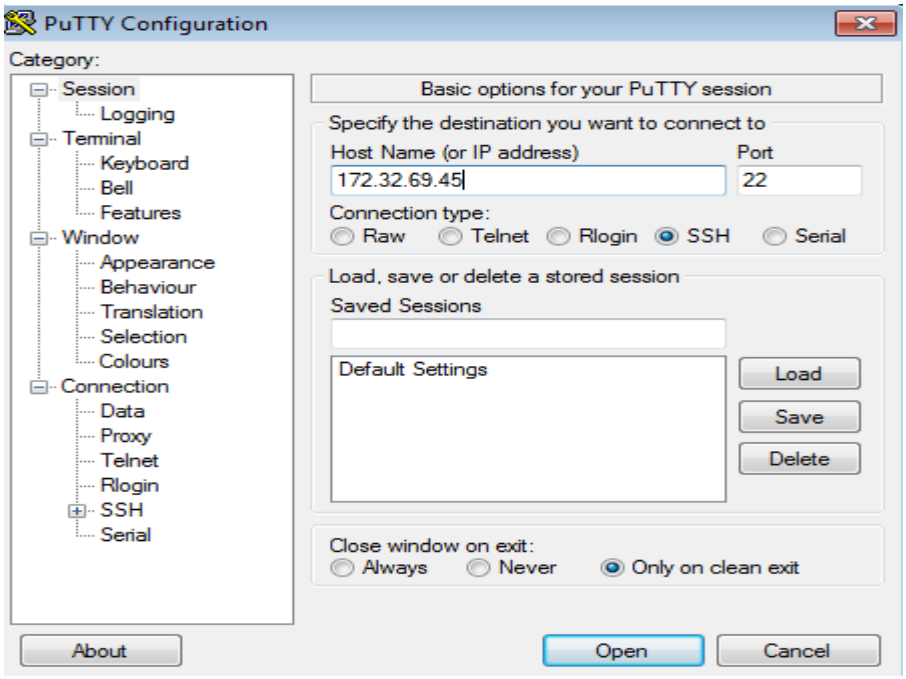
On mets ensuite sur le serveurA la clé publique dans le fichier « authorized_keys » avec la commande : « Echo "<collez ici votre clé publique>" >> /root/.ssh/authorized_keys ».



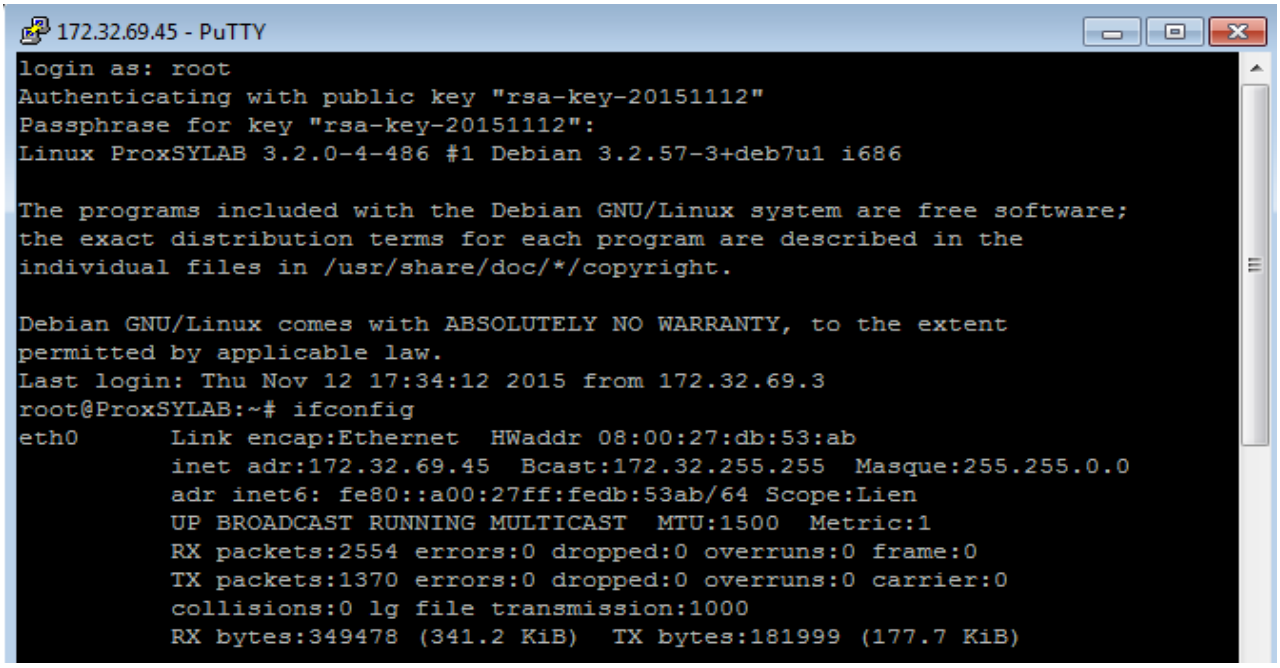
On retourne sur l'administrateurB, on ouvre putty, on vas dans SSH/AUTH et on renseigne la clé privé dans « Private file for authentication ».



Ensuite on se connecte en SSH à l'ip du serveur dans « hostname » on renseigne l'ip du serveur et on clique ensuite sur « open ».



Une fois connecté en tant que root, on rentre la passphrase pour s'authentifier, on fais ensuite un « ifconfig » afin de vérifier l'ip du serveurA, on est bien connecté dessus. On finis par mettre « exit » pour quitter la session.



Astuce : Il est possible de sauvegarder la configuration dans Session sur Putty pour ne pas avoir à paramétrer Putty à chaque fois que l'on veut se connecter à ce serveur.

Conclusion

Nous avons vue que la connexion Telnet malgré une sécurisation des mots de passe n'est pas du tout sécurisé contrairement au protocole SSH d'où le nom Secure Shell.

En effet, ce dernier nous permet d'avoir d'utilisé la méthode RSA est donc être sûr d'avoir une protection de nos données parcourant le réseau WAN.

Nous avons également vue et comprit l’intérêt d'avoir la méthode de cryptage RSA. Vue que cette dernière se base sur l'idée de correspondance.

Avec une méthode RSA, nous pourrons protège nos données en toute sécurité puisque la clé privée doit correspondre à la clé publique. Et dans le cas contraire, le message (ou tout autre données) ne pourra être lue et donc jamais décrypté.