



ECE

PARIS ÉCOLE D'INGÉNIEURS

Rapport Final Linux



Sommaire

I) Ce que l'on a retenu en complément du cours écrit	3
Une vision globale de Linux	3
Séance numero 1	4
Pendant le TP :	4
Séance numéro 2	5
Pendant le cours :	5
Pendant le TP :	6
Séance numéro 3	7
Pendant le cours :	7
Pendant le TP :	7
II) Notre recherche personnelle : La connexion sécurisée à distance avec SSH.....	8
Plusieurs moyens de connexion à un Linux :	8
SSH	8
Création d'un tunnel SSH.....	8
Cryptage RSA	9
Attaque lors de la connexion SSH.....	9
La connexion.....	10
Création d'une clé SSH	12
Les droits sur les fichiers SSH.....	12
L'envoi de fichier et de répertoires	13
Notre application au ssh.....	14
<i>Premiere partie du script</i>	14
<i>Deuxieme partie du script</i>	15
Sources :	18

I) Ce que l'on a retenu en complément du cours écrit

Une vision globale de Linux

Linux n'est pas un OS c'est un noyau.

La philosophie de Linux est : «Tout est fichier ». Les disques durs sont des fichiers, l'écran est un fichier, les applications sont des fichiers...

Les applications fonctionnent toute à partir de paquets. Un paquet est « l'atome » d'un programme : toutes les applications sont constituées de plusieurs paquets. Chaque paquet est un code d'environ 120 lignes qui a une fonction spécifique et qui est suivi régulièrement par des développeurs. Il y a une grande sécurité car les paquets sont contrôlés et mis à jour fréquemment donc il n'y a pas de virus. Le système de Linux est ainsi très stable. L'ensemble des paquets sont stockés dans un serveur appelé dépôt (« repository » en anglais) : c'est à cet endroit que l'on peut télécharger les paquets que l'on souhaite.

De quoi est constitué un paquet ?

Il est constitué d'un fichier binaire(.deb), d'une architecture, d'un numéro de version, d'une description et le plus important d'un ensemble de dépendances : il y a le nom de ces dépendances (avec quels autres paquets il est dépendant), la description fonctionnelle, les programmes et la version.

Le shell n'est pas une console mais un interpréteur. Il fait le lien entre le noyau (kernel) et l'utilisateur qui communique par la console. Les scripts Shell permettent l'automatisation de tâches.

Les commandes sont situées dans une variable appelée PATH. Il faut mettre ./ avant un dossier pour le distinguer d'une commande : si on ne le met pas alors le shell va croire que c'est une commande (au lieu d'un fichier) et va aller chercher dans la variable PATH cette fausse commande. A noter que toutes les commandes sont des programmes.

Apt est le gestionnaire de paquet. **Nano**, **cat** et **gedit** sont trois éditeurs de texte pour la console.

Séance numero 1

Pendant le TP :

On a eu un schéma explicatif sur les rôles du proxy .Le proxy nous donne à l'ECE le droit d'accéder à des pages internet. Si l'on quitte internet quelque temps il faudra redemander au proxy le droit de se connecter à internet. On peut aussi voir les informations de connexion et les matériels réseaux avec **ifconfig**

On a vu la console graphique mais aussi les 6 différentes consoles classiques (CTRL+ALT+F6, CTRL+ALT+F5...).On repasse à la console graphique avec CTRL+ALT+F7.Ces consoles servent surtout quand la console graphique a planté. On aura toujours accès à une autre console non graphique pour relancer le système.

Nous avons appris les commandes basiques **Clear** efface la console et la touche **tab** permet l'auto-complétion avec **ctrl+r** on accède à l'historique des commandes.

On a besoin d'être en « super utilisateur » pour pouvoir supprimer ou installer des paquets. Le super utilisateur a tous les pouvoirs sur la machine c'est pour cela qu'on nous demande notre mot de passe lors de la demande **sudo** car linux est très sécurisé : on ne voit même pas les caractères que l'on tape car c'est encore pour une raison de sécurité : pour que l'on ne puisse pas compter le nombre de caractères de notre mot de passe.

On a surtout utilisé : **sudo apt-get install [nom_du_logiciel]**

On a aussi trouvé comment afficher le code source C d'un logiciel en l'occurrence ici celui de vlc avec la commande **apt-get source [nom_du_logiciel]**

Lors d'un doute sur une commande il ne faut pas hésiter à utiliser le manuel avec **[cmd] man**

Séance numéro 2

Pendant le cours :

Si il y a un bug de la mémoire vive alors son contenu est transféré automatiquement dans une mémoire cache : ce qui nous évite de perdre des données.

Les fichiers sont regroupés sous plusieurs partitions en voici quelques une et leur signification :

les extensions	leur rôle et ce qu'elles contiennent
bin et sbin	ensemble des fichiers et commandes binaires
boot	GRUB programme de boot
dev	point de montage matériel
etc	Fichier de configuration de tous les fichiers installés
home	fichier personnelle
lib	configuration personnelle (librairie système)
root	espace personnel du super-utilisateur
usr	user space logiciel utilisateur
tmp	fichier temporaire des programmes

-> Mettre à jour un paquet implique de mettre à jour aussi les paquets dont il dépend. C'est pour cela que l'on met tout à jour.

-> Pour lire un fichier pendant qu'il est écrit il y a la commande très pratique : **tail -f -fichier** Cela permet d'afficher le texte du fichier au fur et à mesure qu'on le lit.

Les scripts Shell

De quoi sont composés ces scripts ?

1-Du sha-bang qui permet au système de savoir quel Shell doit être utilisé.

2-D'un ensemble de commandes qui seront copiées par le système dans la console.

On a étudié un script. On a découvert que le printf du langage c n'était en fait ni plus ni moins qu'un #DEFINE printf fprintf de même pour #DEFINE scanf fscanf(). Le scanf est un fscanf qui lit dans le fichier stdout et le printf est un fprintf qui écrit ce qui est envoyé en arguments dans un fichier texte : stdin.

On a vu que dans un script le \$variable permet d'insister sur le contenu de la variable.

Il existe plusieurs Shell tel que **sh bash csh** .

Pendant le TP :

Comment court-circuiter le proxy de l'ECE

export proxy_http= «http://proxy.ece.fr : 3128»

Il y a plusieurs éditeurs de texte en console qui sont dans le tableau page1 : nous prendrons **gedit**.

-Les fichiers .zip sont dangereux car ils peuvent cacher des virus contrairement au .b2z

On a parlé des alias qui sont des commandes personnalisées qui raccourcissent des actions (ce sont des macros). Exemple : alias linux= « cd/home/toto/doc/linux».

Nous avons étudié un script qui permet de trier automatiquement un ensemble de musique mp3 avec des critères que nous choisissons. exemple : trier les musiques selon l'auteur,l'album...

Nous avons découvert la commande **whereis [nom fichier]** qui nous indique l'emplacement du fichier recherché.

On a voulu exécuter le script de l'exercice mais il y a eu une erreur :

->erreur de location bash

->on a trouvé l'emplacement en exécutant **whereis bash** qui nous a renvoyé à l'écran de la console 3 informations :

1-la page du manuel ou l'on parle du bash

2-bashrc qui est le script bash pour configurer le bash

3- le chemin où se situe le programme ce qui nous intéresse le plus.

On a donc modifié le script shell mvmp3 et avons indiqué le bon chemin du programme bash.

Ensuite nous avons exécuté le script ./mvmp3 mais il nous manquait id3info. Nous avons fait quelque recherche et avons trouvé que pour avoir id3info, il fallait télécharger la bibliothèque libid3 (en utilisant **apt-cache search id3inf**).

On a relancé le script et ça a bien fonctionné ! Nous pouvons ainsi trier nos musiques selon les critères que l'on écrit dans la console.

-Lorsque l'on fait la liste (-ls) des éléments d'un fichier il y a toujours en première position : '.' et '..'.

'.' Représente le dossier actuel et '..' représente le dossier parent.

Séance numéro 3

Pendant le cours :

Un Framework est un ensemble de logiciel qui permet de concevoir d'autres logiciels.

On a différencié application et logiciel. Une application est un ensemble d'élément de librairie et est composée de plusieurs programmes. Une application a une seule fonction alors que le logiciel a plusieurs fonctionnalités. Exemple : Sur un Smartphone il y a des applications mais le logiciel qui les gère est Androïde.

Pendant le TP :

Dans le cas de la compilation manuelle, il vaut mieux se connecter en root seulement pour la configuration du système. En effet, si l'on compile un .o en tant que root on devient alors le propriétaire du fichier, cela pose donc un problème si on passe notre projet à un ami : il sera obligé d'être à son tour root sur le PC qu'il utilise pour pouvoir avoir les droits d'accès.

Pour quitter le mode Super Utilisateur on utilise la commande exit.

Sudo -s ou **sudo su** permettent d'ouvrir une console en root.

II) Notre recherche personnelle : La connexion sécurisée à distance avec SSH

Plusieurs moyens de connexion à un Linux :

- Telnet mais pas sécurisé
- SSH qui utilise un cryptage asymétrique au moment de la connexion

SSH

Le SSH est un moyen de communiquer de façon sécurisée avec d'autre PC, au moment de la connexion les données sont cryptées et on peut ainsi accéder à un SHELL d'un autre pc partout dans le monde. Le port du SSH est 22, un port est une porte qu'un logiciel écoute afin de recevoir des informations. (Le numéro correspond souvent à une tâche précise exemple port web : 80 ftp : 21 ...)

Création d'un tunnel SSH

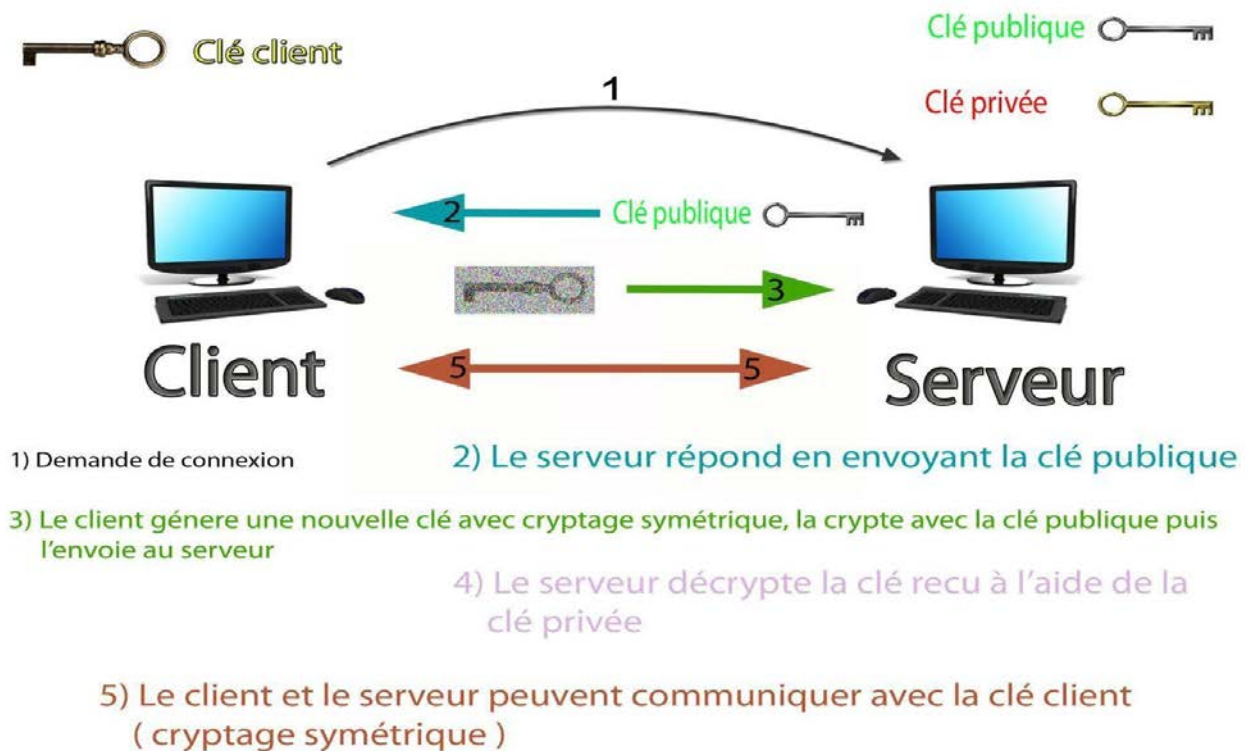
Lors de la création d'un tunnel sécurisé SSH, le serveur génère une clé double une publique de cryptage des données et une privée de décryptage. Les deux clés fonctionnent ensemble.

Il envoie la clé de cryptage au client qui va générer une clé aléatoire de cryptage et décryptage, c'est une clé symétrique, puis la crypte à l'aide de la clé reçue par le serveur. A ce moment, si le client perd sa clé aléatoire il n'aura aucun moyen de la retrouver car elle a été cryptée et il n'y a aucun moyen de revenir en arrière sauf avec la clé secrète du serveur. Mais évidemment, le client garde sa clé générée et envoie la clé cryptée au serveur qui va la décrypter à l'aide de la clé privée. Le client et le serveur possèdent ainsi tous les deux une clé de cryptage et décryptage donc ils peuvent communiquer à l'aide d'un cryptage symétrique.

Chaque serveur a une fingerprint unique qui est un code de la forme 45 :1a:4b :2a :61 :a5 :f9

Donc au moment de la connexion au serveur le client obtient le fingerprint de celui-ci et peut ainsi vérifier si l'empreinte est toujours la même. Dans le cas contraire, il s'agit forcément d'une autre personne qui essaye de se faire passer pour le serveur.

Schéma Bilan



Cryptage RSA

Il existe différents types de cryptage dont le RSA qui a pour principe d'utiliser des fonctions mathématique pour le cryptage et des fonctions inverse pour le décryptage.

Attaque lors de la connexion SSH

L'attaque du milieu est l'une des plus connues. C'est lorsque vous voulez vous connecter chez une autre personne et qu'un pirate informatique se fait passer pour celle-ci. Ainsi, lorsque vous demandez la clé publique de l'autre personne c'est en fait le pirate qui vous envoie sa clé publique. Vous allez ainsi, sans le vouloir, crypter un message qui pourra être décodé par le pirate. Et, pour passer inaperçu, le pirate va prendre la clé publique de la personne à qui était destinée dès le départ le message et va recrypter celui-ci et l'envoyer enfin à son destinataire qui n'aura pas connaissance de ce qu'il s'est passé.

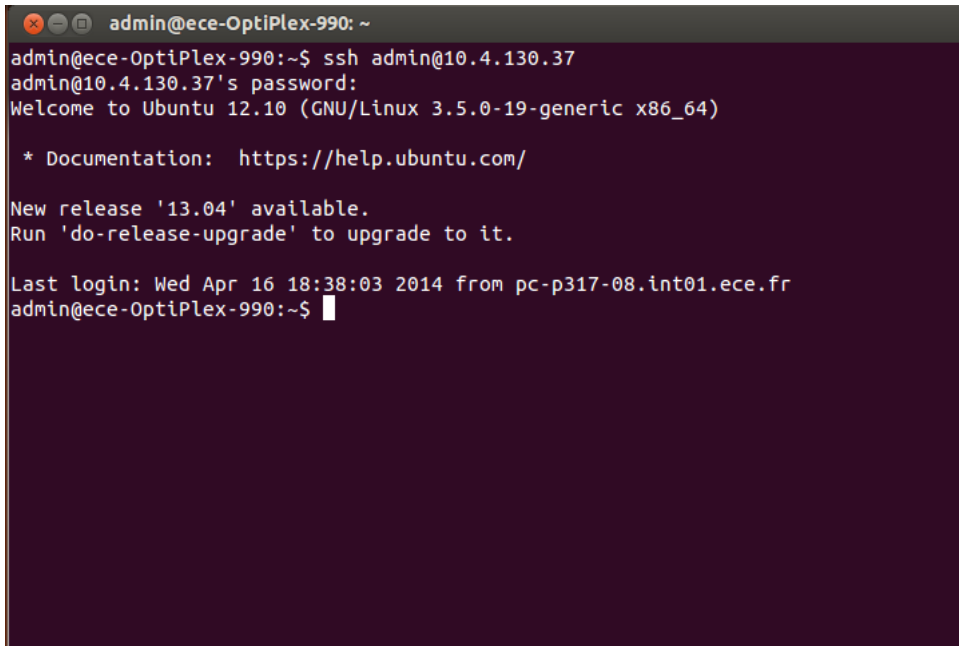
Il y a également d'autres types d'attaque comme celle de la force brute qui permet de trouver le mot de passe pour être root de la machine en testant des combinaisons pour essayer de le trouver. Il vaut mieux lors d'une connexion ssh éviter d'être en root car l'on est probablement plus vulnérable. Certains logiciels comme GPU Brutforcer permettent ce type d'attaque.

La connexion

Pour pouvoir se connecter en SSH, il suffit premièrement d'installer sur le client et le serveur OpenSSH avec les commandes respectives **apt-get install openssh-client ; apt-get install openssh-server**. Puis, pour se connecter sur la machine serveur on utilise la commande **ssh login@ip**

Par exemple d'un à un pc de l'école depuis un autre pc de l'école on écrit

ssh admin@10.4.130.37 le login est admin comme sur tout les pc puis on trouve l' IP à l'aide de la commande **ifconfig**. Une fois la commande effectuée on nous demande le mot de passe de la session

A screenshot of a terminal window with a dark background. The window title is 'admin@ece-OptiPlex-990: ~'. The terminal shows the following text: 'admin@ece-OptiPlex-990:~\$ ssh admin@10.4.130.37', 'admin@10.4.130.37's password:', 'Welcome to Ubuntu 12.10 (GNU/Linux 3.5.0-19-generic x86_64)', '* Documentation: https://help.ubuntu.com/', 'New release '13.04' available.', 'Run 'do-release-upgrade' to upgrade to it.', 'Last login: Wed Apr 16 18:38:03 2014 from pc-p317-08.int01.ece.fr', and 'admin@ece-OptiPlex-990:~\$' followed by a cursor.

```
admin@ece-OptiPlex-990: ~
admin@ece-OptiPlex-990:~$ ssh admin@10.4.130.37
admin@10.4.130.37's password:
Welcome to Ubuntu 12.10 (GNU/Linux 3.5.0-19-generic x86_64)

* Documentation:  https://help.ubuntu.com/

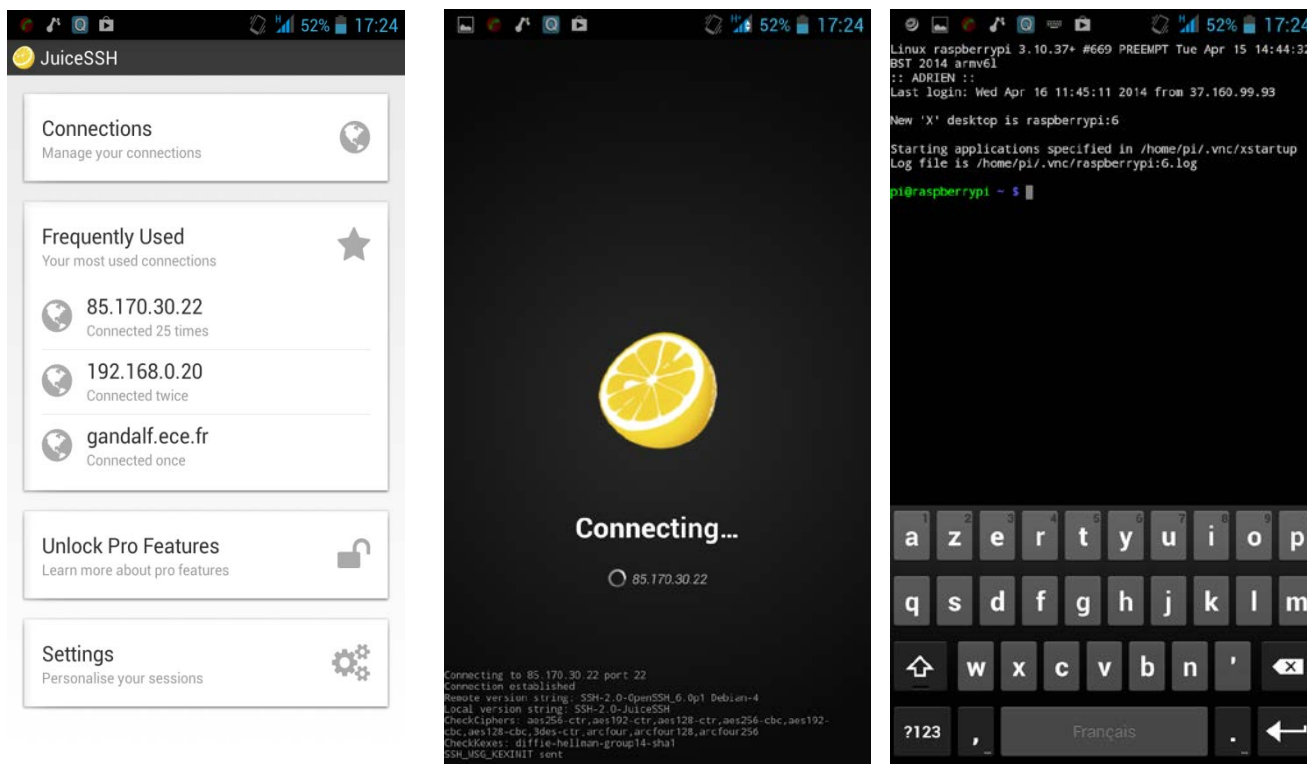
New release '13.04' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Wed Apr 16 18:38:03 2014 from pc-p317-08.int01.ece.fr
admin@ece-OptiPlex-990:~$
```

puis on a accès à l'autre pc à l'aide de la console.

Nous avons aussi essayé de nous connecter sur un raspberry Pi branché à distance, cette fois ci on prend l'ip de la box internet et on n'oublie pas d'ouvrir le port correspondant au ssh c'est à dire le 22 sur la box qu'on redirige sur le raspberry Pi.

Donc depuis un portable à l'aide d'un logiciel de ssh on a réussi à se connecter à un linux :



Création d'une clé SSH

Pour créer une clé SSH avec le cryptage rsa on utilise la commande **ssh-keygen -t rsa**

```
prieux@paris:/etc/ssh$ ssh-keygen -l
Enter file in which the key is (/home/ingenieurs/ing2017/prieux/.ssh/id_rsa): ssh_host_dsa_key
1024 7b:2a:29:6e:84:49:b2:0c:17:57:cc:fb:4f:ff:cb:52 ssh_host_dsa_key.pub (DSA)
prieux@paris:/etc/ssh$ ssh-keygen -l
Enter file in which the key is (/home/ingenieurs/ing2017/prieux/.ssh/id_rsa): ssh_host_rsa_key
2048 b8:48:aa:00:ad:19:d0:2e:56:a0:b3:4b:d5:a4:7d:d8 ssh_host_rsa_key.pub (RSA)
prieux@paris:/etc/ssh$

prieux@paris:/etc/ssh$ ssh-keygen -t rsa
Generating public/private rsa key pair.

Enter file in which to save the key (/home/ingenieurs/ing2017/prieux/.ssh/id_rsa): Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ingenieurs/ing2017/prieux/.ssh/id_rsa.
Your public key has been saved in /home/ingenieurs/ing2017/prieux/.ssh/id_rsa.pub.
The key fingerprint is:
31:ef:5c:b6:83:6f:1d:9e:dc:f3:ae:37:45:8f:d7:79 prieux@paris
The key's randomart image is:
+--[ RSA 2048 ]-----+
|
|      o
|    +   .
|  S . o 0+|
| o + .o.E|
| + 0+ =0|
| ...=00|
| .. .+*|
+-----+

```

Pour revoir la clé ssh il suffit de taper la commande **ssh-keygen -l**

Les droits sur les fichiers SSH

```

-rw-r--r-- 1 root root 125749 2010-03-08 17:11 moduli
-rw-r--r-- 1 root root 1616 2010-03-08 17:11 ssh_config
-rw-r--r-- 1 root root 2460 2010-07-27 10:58 sshd_config
-rw-r--r-- 1 root root 668 2010-07-22 11:38 ssh_host_dsa_key
-rw-r--r-- 1 root root 603 2010-07-22 11:38 ssh_host_dsa_key.pub
1 2 3

```

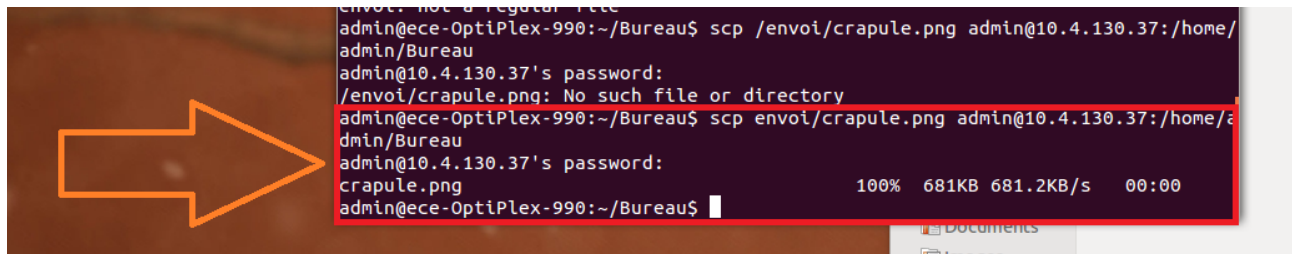
Ici on peut voir les droits d'accès des fichiers liés au ssh

- 1 Tous les fichiers sont autorisés en droit lecture et écriture par le propriétaire, c'est normal car la machine en a besoin pour modifier les clés de connexion.
- 2 Les fichiers de configuration du ssh sont autorisés à la lecture seulement pour le groupe mais pour le fichier `ssh_host_dsa_key` dans lequel la clé de sécurité ssh y est enregistrée le groupe n'y a aucun droit, ce qui empêche le groupe de lire la clé. Mais la `key.pub` qui correspond à la clé publique dispose des droits de lecture par le groupe et les autres utilisateurs on pourra donc la partager lors de la connexion.
- 3 On est dans le même cas que le groupe pour les droits de tout le reste des utilisateurs. La clé reste protégée.

L'envoi de fichier et de répertoires

On peut copier avec la commande scp un fichier de l'ordinateur client vers le serveur sur lequel on se connecte.

exemple : Scp fichier.txt pi@85.170.30.22:/home/pi/Desktop/test/



```
admin@ece-OptiPlex-990:~/Bureau$ scp /envoi/crapule.png admin@10.4.130.37:/home/
admin/Bureau
admin@10.4.130.37's password:
/envoi/crapule.png: No such file or directory
admin@ece-OptiPlex-990:~/Bureau$ scp envoi/crapule.png admin@10.4.130.37:/home/a
dmin/Bureau
admin@10.4.130.37's password:
crapule.png                                100% 681KB 681.2KB/s   00:00
admin@ece-OptiPlex-990:~/Bureau$
```

Mais on peut aussi copier un fichier du serveur sur notre ordinateur

exemple : Scp pi@85.170.30.22:/home/fichier.txt copie_fichier.png

On utilise de la même façon la commande rcp qui permet quant à elle de déplacer des répertoires (pouvant contenir plusieurs fichiers).

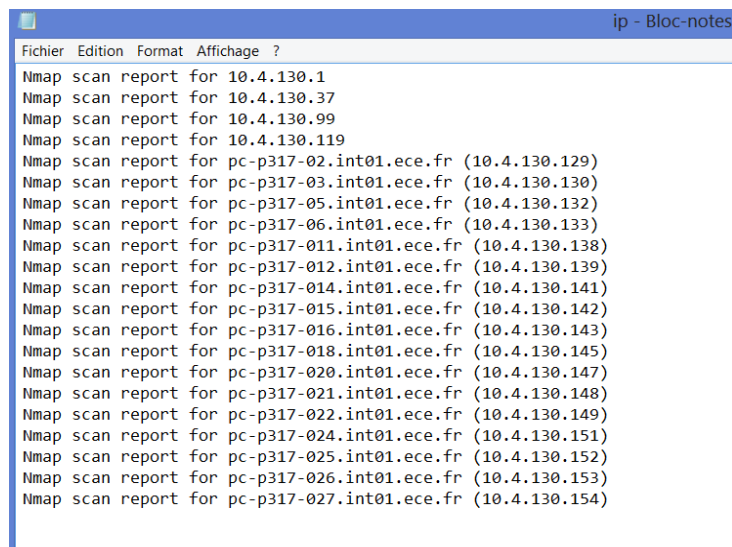
Notre application au ssh

Nous avons décidé de rédiger un script Shell pour pouvoir se connecter automatiquement en ssh à tous les ordinateurs allumés sous linux de la salle P317, ordinateurs appartenant au même réseau de l'ECE et ayant le même mot de passe root. Nous allons vous présenter ce script puis le résultat du rendu. Notre objectif a été d'afficher une image d'un chien sur tous les ordinateurs allumés sous linux de la salle.

Première partie du script

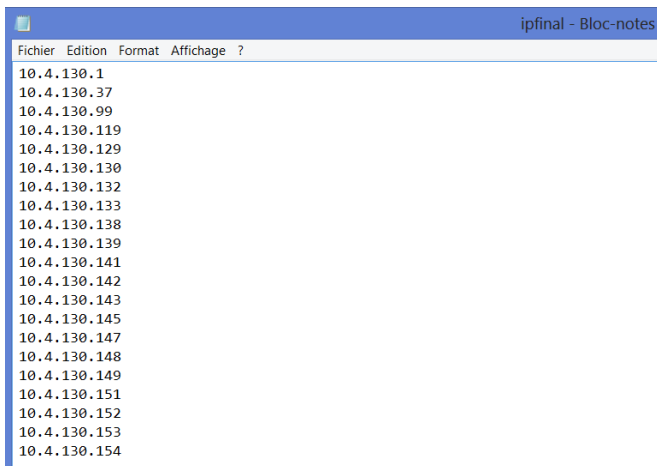
```
1  #!/bin/bash  #Déclaration de notre script en tant que script bash
2
3  # 1- Déclaration des ressources
4  declare -a tab #c'est le premier tableau qui contiendra tout le contenu de la console qui nous intéresse
5  declare -a ip  # c'est le deuxieme tableau
6  rm ip.txt      #fichier texte où l'on fait une premiere selection pour isoler les adresses ip qui nous interessent
7  rm ipfinal.txt #fichier texte où l'on a uniquement les adresses ip des ordinateurs allumés du réseau.
8
9
10 # 2- On souhaite récupérer la liste de toute les adresses ip des ordinateurs allumés de la salle P316 de l'ECE dans un fichier txt
11
12 # 2-1) On récupère les lignes concernant les adresses ip du réseau qui nous intéressent
13 # Avec la commande qui suit on cherche les adresses ip dont le numéro de fin est compris entre 0 et 24
14 # On sélectionne que les lignes des adresses ip où l'ordinateur n'est pas éteint (-v hostdown), qu'il n'est pas un hôte
15 # et qu'il y est écrit Nmap scan report for. Si la ligne de la console vérifie bien toute ces conditions, alors
16 # on l'écrit dans le fichier texte ip.txt
17 nmap -v -sP 10.4.130.0/24 | grep -v "host down" | grep -v "Host" | grep "Nmap scan report for">> ip.txt
18
19 tab=( $( cat ip.txt ) )## On charge en mémoire le fichier ip.txt.
20 ## Automatiquement chaque ligne du fichier sera enregistré dans une case du tableau
21
22 # 2-2) On extrait juste les adresse ip en elle memes des lignes
23 i=0
24 while [ "$i" -lt "${#tab[*]}" ] ##On parcourt chaque du tableau tab
25 do
26   if [[ ${tab[$i]} =~ "10" ]]##Si cette case contient le numero 10 alors...
27   then
28     #echo "${tab[$i]}"
29     echo "${tab[$i]}" | sed -e 's\\(\\|\\)' | sed -e 's\\)\\|\\)' >> ipfinal.txt ## ...On affiche le contenu de la case, puis on enleve de
30     ## ce contenu les paratheses '(' et ')' pour ne récupérer
31     ## seulement l'adresse ip et l'écrire dans ipfinal.txt
32   fi
33   ((i++))
34 done
```

Ce qu'on obtient juste après l'étape 2-1) dans ip.txt



```
Fichier Edition Format Affichage ?
Nmap scan report for 10.4.130.1
Nmap scan report for 10.4.130.37
Nmap scan report for 10.4.130.99
Nmap scan report for 10.4.130.119
Nmap scan report for pc-p317-02.int01.ece.fr (10.4.130.129)
Nmap scan report for pc-p317-03.int01.ece.fr (10.4.130.130)
Nmap scan report for pc-p317-05.int01.ece.fr (10.4.130.132)
Nmap scan report for pc-p317-06.int01.ece.fr (10.4.130.133)
Nmap scan report for pc-p317-011.int01.ece.fr (10.4.130.138)
Nmap scan report for pc-p317-012.int01.ece.fr (10.4.130.139)
Nmap scan report for pc-p317-014.int01.ece.fr (10.4.130.141)
Nmap scan report for pc-p317-015.int01.ece.fr (10.4.130.142)
Nmap scan report for pc-p317-016.int01.ece.fr (10.4.130.143)
Nmap scan report for pc-p317-018.int01.ece.fr (10.4.130.145)
Nmap scan report for pc-p317-020.int01.ece.fr (10.4.130.147)
Nmap scan report for pc-p317-021.int01.ece.fr (10.4.130.148)
Nmap scan report for pc-p317-022.int01.ece.fr (10.4.130.149)
Nmap scan report for pc-p317-024.int01.ece.fr (10.4.130.151)
Nmap scan report for pc-p317-025.int01.ece.fr (10.4.130.152)
Nmap scan report for pc-p317-026.int01.ece.fr (10.4.130.153)
Nmap scan report for pc-p317-027.int01.ece.fr (10.4.130.154)
```

Ce qu'on obtient à l'étape 2-2) dans ipfinal.txt



```
ipfinal - Bloc-notes
Fichier Edition Format Affichage ?
10.4.130.1
10.4.130.37
10.4.130.99
10.4.130.119
10.4.130.129
10.4.130.130
10.4.130.132
10.4.130.133
10.4.130.138
10.4.130.139
10.4.130.141
10.4.130.142
10.4.130.143
10.4.130.145
10.4.130.147
10.4.130.148
10.4.130.149
10.4.130.151
10.4.130.152
10.4.130.153
10.4.130.154
```

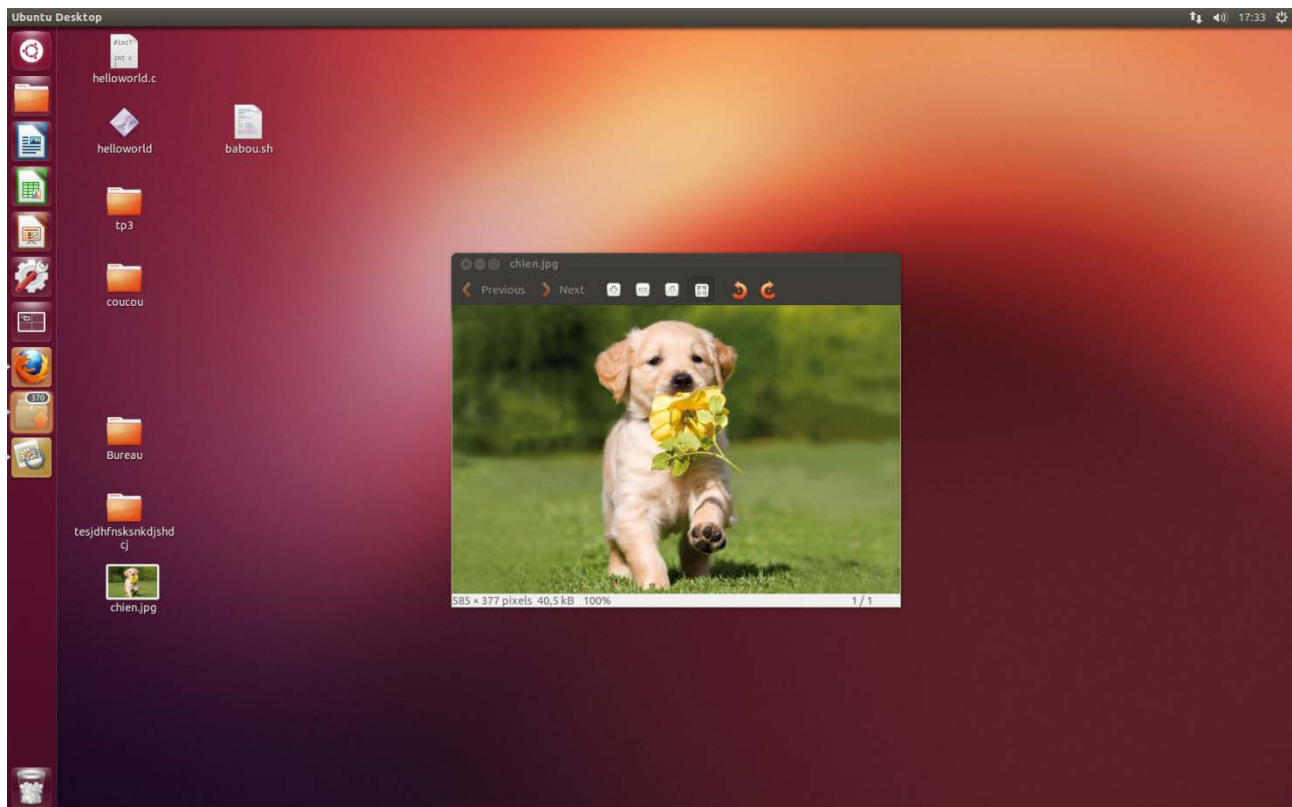
Deuxième partie du script

```
49 # 3-On souhaite afficher une image d'un chien avec une balle sur tous les ordinateurs allumés
50 # et connectés à linux dans la salle par connexion ssh
51
52 # 3-1)On charge en mémoire dans le tableau ip notre fichier texte contenant les ip
53 ip=( $( cat ipfinal.txt ) )
54
55
56 i=0
57 mkdir coucou
58
59 # 3-2)On parcourt prend chaque adresse ip une à une (chaque case du tableau) :
60 while [ "$i" -lt "${#ip[*]}" ]
61 do
62 echo "Connexion par ssh sur l'ip : ${ip[$i]}" #on affiche en console l'adresse sur laquelle on va se connecter
63
64 scp -r /home/admin/Bureau/Linux/chien.jpg admin@${ip[$i]}:/home/admin/Bureau/ #on envoie notre image du chien sur le bureau de l'ordinateur sur lequel on se connecte
65 ssh admin@${ip[$i]} 'export DISPLAY=:0 && cd Bureau && eog chien.jpg' # on se reconnecte en ssh, on exporte l'affichage sur l'écran de l'ordinateur sur lequel
66 # on se connecte puis l'on affiche le chien sur cet ordinateur.
67
68 ((i++)) #on continue à parcourir le tableau
69 done
70
```

Le rendu

```
admin@ece-OptiPlex-990: ~/Bureau/Linux
Connexion par ssh sur l'ip : 10.4.130.37
admin@10.4.130.37's password:
chien.jpg 100% 40KB 39.6KB/s 00:00
admin@10.4.130.37's password:
Connexion par ssh sur l'ip : 10.4.130.99
admin@10.4.130.99's password:
chien.jpg 100% 40KB 39.6KB/s 00:00
admin@10.4.130.99's password:
ssh: connect to host 10.4.130.99 port 22: Connection refused
Connexion par ssh sur l'ip : 10.4.130.119
ssh: connect to host 10.4.130.119 port 22: Connection refused
lost connection
ssh: connect to host 10.4.130.119 port 22: Connection refused
Connexion par ssh sur l'ip : 10.4.130.129
ssh: connect to host 10.4.130.129 port 22: Connection refused
lost connection
Connexion par ssh sur l'ip : 10.4.130.130
admin@10.4.130.130's password:
chien.jpg 100% 40KB 39.6KB/s 00:00
admin@10.4.130.130's password:
Connexion par ssh sur l'ip : 10.4.130.132
admin@10.4.130.132's password:
chien.jpg 100% 40KB 39.6KB/s 00:00
admin@10.4.130.132's password:
```

Voici le script en action, on voit que pour chaque IP une connexion est effectuée. Ensuite le mot de passe est demandé pour transférer l'image du chien et on voit sur la droite si l'image a bien été transmise (taille et fichier d'envoi du fichier) puis on l'affiche à l'écran sur le pc à distance image suivante. Certaines connexions ont été refusées car le client ssh n'était pas installé sur celle-ci.



Voilà le rendu sur chaque pc.

Ce script est donc capable de trouver toutes les adresses IP d'un réseau puis d'y effectuer une connexion ssh, une fois cette dernière réussie on peut faire ce que l'on souhaite sur les pc distants. On a choisi seulement de transférer une image puis de l'afficher mais on aurait pu par exemple éteindre tout les pc. Ce script peut être utile par exemple pour une entreprise qui désire effectuer une mise à jour de tous les serveurs ou alors d'éteindre tout les pc du réseau. On peut encore l'améliorer le script en automatisant l'entrée des mots de passe et aussi en ajoutant une possibilité d'entrée en paramètre la plage d'IP à analyser.

Sources :

Le cours de Mr DUHART

Les explications de Mr DUHART au cours des TP

Le site du zéro (openclassrooms) pour apprendre à se servir de Linux :

<http://fr.openclassrooms.com/informatique/cours/reprenez-le-controle-a-l-aide-de-linux>

SSH : <http://fr.openclassrooms.com/informatique/cours/reprenez-le-controle-a-l-aide-de-linux/la-connexion-securisee-a-distance-avec-ssh>

<http://formation-debian.via.ecp.fr/ssh.html>

Les attaques :

<http://www.nikopik.com/2012/04/un-outil-pour-attaquer-les-acces-root-dun-machine-sous-linux.html>

<http://fr.openclassrooms.com/informatique/cours/les-reseaux-de-zero/la-faiblesse-de-ce-protocole>

Pour avoir un bilan des commandes existantes :

<http://www.fr.hscripts.com/tutoriels/linux-commands>