TP2 - Configuration "moderne" du réseau et `ssh`

L'objectif est, dans la continuité du précédent TP, de voir comment configurer la connexion avec le pendant graphique de ***`network-manager`****, puis directement sous* ***`systemd`****. En effet, les outils graphiques ont été prévus principalement pour une utilisation sur un ordinateur portable et ne devraient donc pas être utilisés sur un serveur. Bien entendu, modifier la configuration réseau d'une machine suppose d'être connecté en super-utilisateur (*`root`*). Seules les grandes lignes des commandes seront décrites, pour avoir la syntaxe complète d'une commande on vous invite à utiliser le manuel :* `man [commande]` *([ ] indique que c'est optionnel), exemple :* `man ls`*.*

IMPORTANT : changer le mot de passe de root !!

1 Connexion réseau avec `network-manager`

Les démons comme `network-manager` et `wicd` (une alternative) sont fournis avec une interface utilisateur graphique et en ligne de commandes (éventuellement via l'installation d'un autre package comme `network-manager-gnome`. Ils permettent de connecter aisément un système à un réseau, qu'il s'agisse de la configuration d'un réseau filaire ou sans fil. La configuration se fait indépendamment de l'ancien package `ifupdown`.

Lorsque l'un de ces démons est installé, sa documentation est fournie dans le répertoire `/usr/share/doc/`, ce qui dans le cas du `network-manager` correspond au fichier suivant :

```
/usr/share/doc/network-manager/README.Debian
```

Les manipulations à effectuer sont :

1. lire le fichier de documentation et en déduire comment faire pour que le `network-manager` gère l'interface réseau de la machine ;
2. modifier le fichier `/etc/network/interfaces` en conséquence ;
3. arrêter le processus `dhclient` (lancé via la directive `dhcp` du fichier `interfaces`, il peut écraser une configuration mise en place ultérieurement) via la commande `kill -9` suivi du PID du processus à "tuer". Pour obtenir le `pid` de `dhclient`, procéder comme suit :

   ```
   ps -aef | grep dhclient
   ```

4. redémarrer le `network-manager` via la commande suivante :
   `service NetworkManager restart`
5. un message devrait indiquer que la connexion est maintenant gérée par `network-manager` ou l'apparence de l'icône changer ;
6. pour obtenir des informations sur la configuration actuelle, cliquer avec le bouton droit sur l'icône dans la barre du haut, puis cliquer sur `Informations sur la connexion` (apparaît probablement avec le nom `Connexion filaire 1` ;
7. cliquer à nouveau avec le bouton droit, puis choisir `Modifier les connexions` suivi de `Connexion filaire 1` et enfin cliquer sur la roue dentée à côté de + -. Consulter ensuite l'onglet `Paramètres IPv4` ;

8. la configuration actuelle étant automatique et dynamique grâce au protocole DHCP, vous allez maintenant la changer pour passer à une configuration statique / manuelle. Pour cela, les informations qui sont à utiliser sont, d'une part celles données par le contenu du fichier `/etc/network/interfaces` que vous pouvez trouver au niveau de la page 5 de l'énoncé de TP1, d'autre part celles du contenu du fichier `/etc/resolv.conf` donné à la fin de la page 8 du même énoncé. Naturellement, à l'issue de la configuration il faut que le réseau continue à fonctionner, sinon cela signifie que vous aurez mal saisi une information nécessaire...

2 Connexion réseau avec `systemd`

RAPPEL : `systemd` n'est pas très adapté pour gérer des connexions temporaires.

Sous `systemd`, il y a deux démons qui servent à configurer le réseau :

1. `systemd-networkd`, qui est le démon servant à définir la configuration ;
2. `systemd-resolved`, qui peut être utilisé pour la résolution de nom en lieu et place du fichier `/etc/resolv.conf` vu dans le TP1.

Il faut également noter que ces services sont souvent activés par défaut et qu'il ne devraient pas être désactivés.

En terme de fichiers de configuration, les fichiers associés aux deux démons peuvent être placés dans deux répertoires : `/usr/lib/systemd/network` ou `/etc/systemd/network`. De manière générale, on trouve dans le répertoire `/usr/lib/systemd` les fichiers de configuration définis par les packages qui sont installés, alors que `/etc/systemd` est prévu pour accueillir les fichiers de configuration définis par l'administrateur de la machine. Les fichiers dans `/etc/systemd/network` ont une priorité supérieure à ceux placés dans l'autre répertoire.

Pour ce qui est des fichiers de configuration pour `systemd-networkd`, ils sont de trois types :

— `.link` → un tel fichier est chargé quand une interface réseau "apparaît" dans le système (via `udev`), si un fichier correspondant à l'interface est trouvé ;
— `.netdev` → un fichier de ce type est utilisé pour créer des interfaces virtuelles ;
— `.network` → un fichier avec cette extension est chargé pour associer une configuration réseau à une interface.

Chaque fichier commence par une section `[Match]` qui définit toutes les conditions qui doivent être vérifiées pour que le profil défini par le fichier soit activé. Si la section est vide alors le profil sera systématiquement mis en place. Les fichiers de configuration sont traités dans l'ordre lexicographique. Pour une description plus détaillé du contenu de ces fichiers il faut consulter les pages du manuel via `man systemd.link`, `systemd.netdev` ou `systemd.network`. Dans la suite, nous ne mettrons en place que des fichiers `.network`.

2.1 Gestion des services avec `systemd`

Avant de nous focaliser spécifiquement sur la configuration du réseau avec `systemd`, voyons quelques manipulations qui permettent de gérer des services quelconques. Celles-ci utilisent toutes la commande `systemctl` (cf. `man systemctl`), en prenant généralement comme paramètre un fichier associé à un service. Comme exemple on considérera le service (serveur) `ssh`.

En comparaison avec SysV, on a :

— les fichiers `.service` qui correspondent aux scripts `init`, par exemple pour le service correspondant au serveur web Apache on aurait
`/etc/init.d/apache2` ⇔ `/lib/systemd/system/apache2.service` ;

— et `systemctl` qui joue le même rôle que la commande `service`, d'où l'équivalence `service apache2 start` ⇔ `systemctl start apache2.service`.

Les manipulations à effectuer sont :

1. afficher la liste des services actifs

   ```
   systemctl list-units --type=service
   ```

   Le service `ssh.service` est-il actif ?

   Remarques : comme expliqué précédemment, le fichier `ssh.service` qui est présent dans `/lib/systemd/system` est l'équivalent du fichier `ssh` présent dans `/etc/init.d`. Il faut également noter que l'auto-complétion fonctionne pour le nom du service.

2. connaître le statut d'un service, en l'occurrence `ssh`

   ```
   systemctl is-active ssh.service
   ```

   pour avoir plus de détail

   ```
   systemctl status ssh.service
   ```

   Qu'obtenez-vous comme affichage en remplaçant `ssh.service` par `sshd.service` ?

3. arrêter un service (puis vérification de son état / statut)

   ```
   systemctl stop ssh.service
   systemctl status ssh.service
   ```

   Il n'est alors plus possible de se connecter avec `ssh` sur votre machine. Vous demanderez à l'un de vos voisins de faire le test, afin de constater que la connexion est bien refusée.

4. démarrer un service (puis vérification de son état / statut)

   ```
   systemctl start ssh.service
   systemctl status ssh.service
   ```

   Comme plus haut, demander à quelqu'un d'essayer de se connecter sur votre machine.

5. redémarrer un service

   ```
   systemctl restart ssh.service
   ```

6. recharger la configuration d'un service pour prendre en compte un changement sans interrompre le service

   ```
   systemctl reload ssh.service
   ```

   En effet, un redémarrage stoppe un service, ce qui peut, par exemple, pour certains services aboutir à la déconnexion d'utilisateurs.

7. configurer un service pour qu'il ne soit plus lancé automatiquement au démarrage (les commandes d'arrêt / de démarrage ne sont valables que jusqu'au prochain reboot)

   ```
   systemctl disable ssh.service
   ```

   on vérifie qu'il est désactivé en regardant son statut

   ```
   systemctl status ssh.service
   ```

   puis on réactive le lancement automatique au démarrage avec

   ```
   systemctl enable ssh.service
   ```

2.2 Désactivation de `network-manager`

Avant de procéder à la configuration via `systemd`, il faut naturellement veiller à désactiver `network-manager`. Pour cela, vous allez devoir identifier le service correspondant, l'arrêter, puis pour éviter qu'il soit à nouveau présent suite à un éventuel redémarrage, le désactiver.

Les manipulations à effectuer sont :

1. trouver le service associé au `network-manager` ;
2. le stopper et constater le changement au niveau de l'icône associée au `network-manager` ;
3. regarder le statut du service. Avez-vous encore accès au réseau, à Internet ?
4. désactiver le service.

2.3 Configuration réseau basique

Nous allons maintenant utiliser `systemd-networkd` et `systemd-resolved`, via la définition d'un fichier `.network`, pour définir différentes configurations : une configuration utilisant le protocole DHCP et une configuration avec une adresse IP statique.

Les fichiers qui suivent devront être placés dans le répertoire `/etc/systemd/network`.

2.3.1 Configuration dynamique via DHCP

Contenu du fichier `10-dhcp.network`

```
[Match]
Name=eno1

[Network]
Description="Carte ethernet config. dynamique"
DHCP=ipv4
LinkLocalAddressing=ipv6
IPv6AcceptRA=true

[DHCP]
UseDNS=true
UseDomains=true
UseNTP=true
UseRoutes=true
UseTimezone=true
```

2.3.2 Configuration statique

Contenu du fichier `20-static.network` (`X` est le numéro de votre machine multiplié par 10, le numéro est écrit en bas à droite de l'écran)

```
[Match]
Name=eno1

[Network]
Description="Carte ethernet config. statique"
Address=172.20.20.X/24
```

```
Gateway=172.20.20.254
UseDomains=yes
Domains=iut-bm.univ-fcomte.fr
DNS=193.52.61.11
DNS=194.57.86.193
```

2.3.3 Mise en pratique

Les manipulations à effectuer sont :

1. saisir les deux fichiers (`10-dhcp.network` et `20-static.network`) ;
2. démarrer le service `systemd-networkd` via

   `systemctl start systemd-networkd.service`

   ou en version courte

   `systemctl start systemd-networkd`
3. démarrer le service `systemd-resolved` ;
4. consulter le contenu du fichier `/run/systemd/resolve/resolv.conf` ;
5. remplacer le fichier `/etc/resolv.conf` par un lien symbolique sur le fichier maintenu par `systemd`

   `ln -sfv /run/systemd/resolve/resolv.conf /etc/resolv.conf`
6. activer les deux services pour qu'ils soient démarrés lors d'un reboot ;
7. vérifier que le statut des deux services est correct (`enabled`) ;
8. rebooter la machine.

Normalement, après le reboot, la machine est correctement configurée suivant les indications données dans l'un des deux fichiers. Pour vérifier que c'est effectivement le cas, nous allons utiliser la commande `networkctl` qui permet d'afficher des informations sur la configuration. Cette commande est propre à `systemd`.

Les manipulations à effectuer sont :

1. affichage de l'état des interfaces

   `networkctl`
2. affichage de la configuration des interfaces

   `networkctl status`
3. affichage de la configuration détaillée de l'interface `eno1`

   `networkctl status eno1`
4. affichage de la configuration détaille de toutes les interfaces

   `networkctl status --all`

Au vue des informations données par `networkctl`, quelle est la configuration active ? Quelle est la raison qui explique le choix de la configuration effectivement active ?

Les manipulations à effectuer sont :

1. apporter les modifications pour que la configuration statique soit celle qui est activée au prochain reboot ;
2. rebooter la machine ;
3. vérifier que la configuration statique est bien active et opérationnelle ;
4. comment revenir à la configuration dynamique sans rebooter ?

3 Connexion sécurisée à distance : le retour de `ssh`

3.1 Pourquoi `ssh` est incontournable pour le travail à distance

Pour l'utilisateur et comme on l'a déjà évoqué lors des TP de la ressource R1.03, une connexion `ssh` peut sembler identique à `telnet`. Cependant, `ssh` fait des choses bien plus complexes qui permettent de garantir :

- la confidentialité → chiffrement des paquets qui circulent ;
- l'intégrité → on a la garantie que les paquets circulant ne sont pas altérés / modifiés ;
- l'authentification → lors de l'établissement de la connexion. L'identité du serveur est vérifiée par sa clé dans `~/.ssh/known_hosts` (fichier stocké sur la machine cliente), puis celle du client par son mot de passe ou par une clé publique stockée sur la machine serveur dans `~/.ssh/authorized_keys`. La clé du serveur est récupérée par le client lorsque l'on répond `yes` à la première connexion sur le serveur.

`ssh` permet également de mettre des limitations à la connexion via un fichier `authorization` et de créer des tunnels qui transportent des informations habituellement en clair.

3.2 Authentification par clé

Par défaut, `ssh` authentifie l'utilisateur via son mot de passe, ce qui laisse une vulnérabilité potentielle si celui-ci est facile à découvrir. Face à cette faiblesse, l'authentification par clé est l'alternative la plus efficace car elle permet de garantir au serveur que le client (l'utilisateur) est bien celui qu'il prétend être.

L'authentification par clé fait intervenir trois éléments :

- une clé publique qui sera exportée sur chaque machine (ou hôte) serveur sur laquelle on veut pouvoir se connecter ;
- une clé privée (liée mathématiquement à la clé publique) qui permet de prouver son identité, elle est stockée sur la machine cliente ;
- une passphrase qui sécurise l'utilisation de la clé privée.

On voit bien que la sécurité est améliorée car sans la passphrase on ne peut utiliser la clé privée et inversement la passphrase seule ne sert à rien.

3.3 Création de paires de clés et connexion avec celles-ci

La commande permettant de créer une paire de clés est `ssh-keygen`. Elle permet d'autre part de gérer les clés stockées dans `~/.ssh/known_hosts`. Par exemple, lorsqu'une machine distante est réinstallée de nouvelles paires de clés utilisées lorsqu'elle fait office de serveur (stockées dans `/etc/ssh`) sont mises en place, or si une machine cliente a une clé serveur ancienne la connexion ne sera pas possible. Lorsque ce problème se pose il faut donc commencer par supprimer les anciennes clés serveur sur la machine cliente avec `ssh-keygen`.

Différents types de clés sont disponibles avec pour certains la possibilité de préciser sa taille :

- DSA et RSA dont on peut préciser la taille en bits ;
- ECDSA et Ed25519.

En terme de choix, DSA semblerait à éviter, tandis que pour RSA une taille minimale de 2048 bits est indiquée, mais une taille de 3072 ou 4096 bits serait préférable. ECDSA et Ed25519 sont a priori les choix les plus robustes (cryptographie sur les courbes elliptiques).

Dans la suite, vous allez tout d'abord regarder la configuration serveur `ssh` de votre machine, puis vous générerez une paire de clés que vous exporterez sur une machine voisine.

Les manipulations à effectuer sont :

1. consulter le contenu du répertoire `/etc/ssh`.
   Combien y a-t-il de paires de clés serveur et de quels types sont-elles ?
2. le cas échéant, déterminer la taille en bits des clés avec `ssh-keygen -lf` ;
3. quels sont les fichiers définissant, respectivement, la configuration de `ssh` en mode client et en mode serveur ?
4. en consultant le contenu de ces fichiers que constatez-vous ?
5. retourner dans le répertoire racine du compte `tpreseau`

   ```
   cd ~tpreseau
   ```

6. générer une paire de clé de type Ed25519

   ```
   ssh-keygen -o -a 100 -t ed25519
   ```

7. consulter le manuel pour déterminer le rôle des options `-o` et `-a` ;
8. générer une paire de clé de type RSA de taille 4096 bits ;
9. dans quel répertoire se trouvent vos deux paires de clés ?
10. passer `root` avec un `su -` et créer un compte utilisateur `voisin` avec la commande `adduser` (ce compte `voisin` permettra à l'un de vos camarades de se connecter sur votre machine et d'y exporter ses clés) ;
11. se connecter avec `ssh` sur le compte `voisin` d'un de vos camarades.
    Quel est le type de la clé publique que le serveur `ssh` (la machine de votre camarade) exporte dans votre compte ? Comment êtes-vous authentifié ?
12. exporter / distribuer les deux clés publiques créées précédemment avec la commande `ssh-copy-id` ;
13. se reconnecter avec `ssh` sur le compte `voisin` d'un de vos camarades.
    Comment êtes-vous maintenant authentifié ?
14. consulter le contenu du fichier `authorized_keys` du répertoire `.ssh` à la racine du compte `voisin` ;
15. se déconnecter de la machine distante ;
16. pour éviter de saisir à chaque connexion la passphrase associée à une clé privée (identité), celle-ci peut être mémorisée via le `ssh-agent`. Commençons par effacer toutes les "identités" mémorisées avec

    ```
    ssh-add -D
    ```

    Ensuite on ajoute nos "identités"

    ```
    ssh-add
    ```

17. supprimer la clé serveur de la machine de votre camarade du fichier `known_hosts` grâce à la commande `ssh-keygen` ;
18. exécuter un `ssh-add -D` et se reconnecter sur la machine de votre camarade via `ssh` en utilisant l'option `-i ~tpreseau/.ssh/id_rsa` (nom du fichier de la clé publique RSA).

ATTENTION : `ssh` permet de déporter ("*forwarder*") un agent `ssh`. Le mécanisme de forward d'agent `ssh` permet de se connecter très simplement à un serveur, en rebondissant sur un serveur intermédiaire sans avoir à stocker sa clef privée sur ce dernier. Ce mécanisme peut toutefois donner lieu à de nouvelles cyberattaques via le vol d'agent `ssh`.

3.4 Exécution de commande / copie de fichier à distance et connexion en mode graphique

En plus d'avoir une connexion distante permanente, `ssh` permet également d'exécuter des commandes à distance et de récupérer leur résultat sans rester connecter. D'autre part, dans la section précédente on ne s'est connecté qu'en mode texte, or il est également possible de se connecter de façon à pouvoir lancer, sur la machine serveur, des applications ouvrant des fenêtres graphiques dont l'affichage sera déporté sur la machine cliente.

Les manipulations à effectuer sont :

1. on fait un `ls` via `ssh` dans le compte `voisin` sur la machine `localhost`

   ```
   ssh voisin@localhost 'ls -al'
   ```

   Quelle machine désigne `localhost` ?
2. créer un petit fichier texte avec le commande `echo`

   ```
   echo 'Hello world!!' > hello.txt
   ```
3. le copier dans le compte `voisin` sur la machine d'un de vos camarades via `scp` (utiliser la manuel pour avoir la syntaxe de la commande) ;

Le travail en mode graphique sur la machine distante / serveur repose sur le concept d'`export de display`, c'est-à-dire que la fenêtre graphique d'une application lancée sur la machine distante sera exportée sur la machine locale / cliente. La machine locale / cliente enverra quant à elle les données du clavier et de la souris vers la machine distante / serveur.

`ssh` permet de faire de l'export de display à condition de configurer le serveur pour l'autoriser. La connexion se fait alors en utilisant l'option `-X`.

Les manipulations à effectuer sont :

1. consulter le fichier de configuration du serveur `ssh` de votre machine et identifier le paramètre qui active l'export de display ;
2. utiliser `ssh` pour se connecter à la machine d'un de vos camarades en activant l'export de display et lancer le navigateur `firefox` sur la machine distante ;
3. se déconnecter, modifier le paramétrage de votre serveur `ssh` de façon à ce que l'on ne puisse plus se connecter en mode graphique sur votre machine, puis recharger la configuration du serveur via `systemctl` ;
4. demander à un de vos camarades d'essayer de se connecter en mode graphique avec `ssh` à votre machine et de lancer à `firefox`.

3.5 Paramétrage du serveur

Dans le fichier définissant la configuration du serveur (ou démon) `ssh`, toutes les valeurs commentées (précédées d'un #) sont placées à leur valeur par défaut. Par exemple, la ligne `#PubKeyAuthentication yes` signifie que par défaut le paramètre `PubKeyAuthentication` est à `Yes` et que donc l'authentification par clé est autorisée. Pour modifier un paramètre, il suffit donc de le décommenter et de définir sa valeur.

Les manipulations à effectuer sont :

1. modifier le numéro du port sur lequel le démon `ssh` est à l'écoute pour qu'il passe de 22 à 2022 ;
2. faire en sorte que le serveur utilise sa paire de clés Ed25519

3. désactiver l'authentification par mot de passe (naturellement cela n'a de sens que si les utilisateurs qui peuvent se connecter chez vous on effectivement mis en place une authentification par clé) ;
4. désactiver l'accès distant à votre compte `root` ;
5. après avoir rechargé la configuration du service `ssh` vous vérifierez que celle-ci correspond bien à ce qui est demandé

Pour ceux qui arriveraient là et auraient encore du temps, regarder comment il est possible de configurer le client `ssh` (le manuel est votre ami...).