Sujet de TP - Connexions et transferts

Remarques préliminaires

Les remarques constructives relatives à ce suport sont à transmettre à l'adresse `clot@univ-lyon1.fr`.

Ce support aborde deux sujets : d'une part les détails des opérations de connexions locales et distantes, et d'autre part les outils de transfert de fichier. Concernant le premier sujet, les paramètres de connexion, directs ou indirects, qui déterminent l'environnement du shell avec lequel l'utilisateur interagit sont présentés, ainsi que les différents niveaux d'interactivité auquels l'usager peut recourir.

La seconde partie présente quelques commandes permettant la copie de fichiers d'un système unix à un autre, puis les outils classiques de transfert de fichier, et également quelques commandes simples vous permettant d'utiliser le biais du mail pour de «petits» transferts.

Il est vivement conseillé de travailler à la fois avec votre compte étudiant et votre compte personnel sur `minisfa`, i.e. dans un environnement graphique et un environnement console : certains mécanismes d'initialisation varient d'un environnement à l'autre.

Connexion locale via la console

Lors d'une connexion sur un système via la console, le shell de connexion utilisé est celui déclaré par l'administrateur comme shell de démarrage. Selon le système d'information employé pour la gestion des comptes, cette information est plus ou moins facilement consultable et modifiable. Sur `minisfa`, ce shell est le dernier champ de la ligne renvoyée par `id -P`. Sur un système Linux, cette ligne se trouve dans le fichier `/etc/passwd` (ou l'équivalent utilisé pour la gestion des utilisateurs).

Ce shell de connexion initialise votre environnement (redéfinition des séquences générant des signaux, modification de variables telles que `PATH`, définition d'alias, exécution de commande...) notamment en fonction d'informations consultées dans certains fichiers. En fonction du shell de login, les fichiers consultés sont différents :

- `sh` tente de lire les instructions placées dans `/etc/profile` puis `~/.profile` s'ils existent [1] ;
- `bash` exécute tout d'abord les commandes se trouvant dans le fichier `/etc/profile` s'il existe. Après lecture de ce fichier, il recherche `~/.bash_profile`, `~/.bash_login`, et `~/.profile`, dans cet ordre, et exécute les commandes se trouvant dans le premier fichier existant et accessible en lecture.
- `ksh` procède différemment, de même que `csh`...

Notez que lorsque ces shells sont invoqués localement alors que l'utilisateur est déjà connecté, les fichiers lus lors de leur démarrage sont différents.

Déterminez votre shell de connexion. Pour cela, plusieurs solutions s'offrent à vous : vous pouvez déterminer la liste des processus dont vous êtes propriétaire, chercher dans les fichiers de gestion des comptes (ou utiliser la commande `id -P` si vous êtes sur `minisfa`) ou plus simplement afficher

1. pour plus d'informations, consulter le manuel.

la variable du shell permettant de consulter le programme en cours (comme vous le feriez à partir d'un script, i.e. en affichant le contenant de la variable `$0`).

Déterminez les fichiers d'initialisation lu par votre shell lors de votre login et identifiez ceux existant dans votre home sur `minisfa` et sur les machines du campus. Si de tels fichiers existent, analysez leur contenu. Sinon, créez avec `touch` un fichier qui sera completé plus loin.

Il est possible de se connecter sur un mac directement dans une console. Il faut pour cela activer l'ouverture de session par nom et mot de passe. Une fois cela effectif, il est possible de saisir dans le champ correpondant au nom (i.e. login) la valeur `>console` qui permet d'accéder à une console.

A ce stade, vous avez dû identifier un fichier dans lequel vous pourrez placer des commandes qui seront exécutées lors de votre login. Si ce n'est pas le cas, reprenez la question précédente. Dans la suite de ce support, nous supposerons que ce fichier est nommé `INIT` (ce qui ne veut pas dire de travailler sur un fichier nommé `INIT`). Créez un fichier nommé `.alias` et contenant la définition de vos alias préférés (e.g. cf. sujet portant sur le shell). Vous testerez la validité de vos définitions. Dans le fichier `INIT`, placez une commande permettant d'afficher le message `Lecture du fichier INIT` et un test d'existence du fichier `.alias` conditionnant l'exécution de la commande `source ~/.alias`. Cette commande devrait être exécutée au prochain login. Vous testerez l'effectivité de votre nouveau paramétrage en vous re-connectant.

Créez un répertoire nommé `script` dans votre home et placez à l'intérieur un fichier contenant les lignes

```
#!/bin/bash
echo "Je suis un script invoque avec $# arguments"
```

Rendez ce script exécutable, puis ajoutez au contenu de la variable `PATH` la chaîne `:~/script`. Assurez-vous que le script précédemment créé peut être exécuté en l'invoquant par son simple nom. Si cela fonctionne, ajoutez la re-définition de la variable `PATH` dans `INIT` afin que le script soit invocable dans tout nouveau shell.

Ajoutez au fichier `INIT` une ligne permettant de tester la présence de nouveau mail. La commande `mail` pourra être utilisée pour tester cela (ne fonctionne que sur `minisfa`). La chaîne "`Vous avez un nouveau mail`" suivie des en-têtes des cinq premiers mails seront affichées, si la commande `mail` signale de nouveaux mails. La commande `mail` est capable de vous fournir de courtes en-têtes des mails. Sinon, aucun message ne sera imprimé. Avant d'insérer la ligne demandée dans `INIT`, vous mettrez au point dans la console les commandes permettant de réaliser le travail demandé.

Initialisation d'un shell qui n'est pas un shell de démarrage

Lors de l'ouverture d'un terminal dans un environnement fenêtré, le shell démarré n'est pas initialisé selon la même procédure : les fichiers d'initialisation ne sont pas nécessairement les mêmes. Il est important d'identifier les bons fichiers afin d'y placer les initialisations souhaitées...

Identifiez le ou les fichiers permettant de completer l'initialisation d'un shell démarré dans un terminal graphique. Si le fichier n'existe pas, créez-le à l'aide de `touch`. Placez dans ce fichier les commandes permettant d'utiliser les alias définis dans le fichier `~/.alias` s'il existe.

Sur un mac, il y a deux possibilités d'accéder à un shell dans un environnement graphique : par le biais du programme Terminal (Application> Utilitaire> Terminal) ou xterm (Applications> X11). Le shell démarré par le programme xterm s'initialise comme dans un environnement graphique linux standard, alors qu'avec Terminal il est initialité comme un shell de login. Ceci est une exception notable et source de confusion!!. Beware!

Cloture d'un shell de login

Certains shells permettent d'exécuter les commandes présentes dans un fichier lors de la déconnexion.

Déterminez si votre shell dispose de ce mécanisme. Vous savez où chercher cette information...En anglais, déconnexion se dit *logout*.


Si votre shell le permet, définissez les actions suivantes afin qu'elles soient exécutées lors de votre déconnexion : supression de vos fichiers dans le répertoire `tmp`, identification du nom de votre console (cmd `tty`) afin d'extraire de la sortie de la cmd `last` l'heure de votre login et utilisation de cette date pour identifier tous les fichiers modifiés depuis, la liste de ces fichiers étant à transmettre par la commande `mail` vers une de vos adresses électroniques.

Connexion locale via le gestionnaire graphique de connexion

Lors de la connexion à un système via le gestionnaire de connexion graphique, vous pouvez choisir le gestionnaire de fenêtrage (fvwm, kde, gnome, windowmaker...). Le démarrage de cet environnement est assuré par un script, en général `.xinitrc` mais ceci est variable d'une distribution à l'autre (ubuntu, redhat, knoppix...). De plus, chaque gestionnaire de fenêtrage dispose de ses propres fichiers d'initialisation à partir desquels il est possible d'ajouter divers paramètres.

Notez que ces manipulations ne peuvent pas être testées sur `minisfa`. Sur les machines du crip, si vous utilisez l'environnement graphique gnome (celui utilisé implicitement), vous pouvez passer les deux premières questions et les remplacer par les trois suivantes.

- Recherchez sur votre système s'il existe un fichier nommé `.xinitrc`. Si c'est le cas, analysez ce script.
- Démarrez une session graphique et déterminez le gestionnaire de fenêtrage utilisé. Identifiez la commande associée permettant de le démarrer. Recherchez de l'aide relative à cette commande afin d'identifier les fichiers de configuration utilisés pour son démarrage.
- Dans la barre de menu de votre environnement, sélectionnez `systeme>preferences>sessions`, puis dans l'interface ouverte, ajoutez un programme à exécuter au démarrage de gnome. Vous pourrez par exemple demander l'ouverture d'un terminal X (commande `xterm`). Fermez votre session graphique puis reconnectez-vous afin d'observer l'exécution demandée au démarrage de la session.
- Rechercher dans votre home les fichiers modifiés il y a moins de 20 minutes (ou un délai permettant de remonter avant le paramétrage précédent) et contenant le nom de la commande programmée précédente. Quel est le fichier engendré par votre paramétrage ?
- Remplacez dans ce fichier la commande `xterm` par la commande `xeyes` et testez le lancement de cette nouvelle commande au démarrage.
- Placez dans le fichier approprié les lignes nécessaires afin que la présence de nouveaux mails soit signalée par une boite de message comportant le message "`Nouveau mail`". Vous pourrez utiliser la commande `xmessage` pour la création de la boite.
- Ajoutez les lignes permettant de tester si le contenu du fichier pointé par l'url `isfaserveur.univ-lyon1.fr/~denis.clot/IRM1.html` a été modifié depuis votre dernier login. Vous pouvez par exemple utiliser `wc` pour baser votre test sur un comptage des lignes, mots et caractères. Vous pouvez également utiliser un test plus précis basé sur le calcul de somme contrôle telles que `md5sum` ou `sha1sum`.
- Modifiez les lignes précédentes afin de réaliser un contrôle de la publication de nouveaux supports de TD sur `isfaserveur.univ-lyon1.fr/~denis.clot/IRM1.html` et leur téléchargement et permettant d'afficher une boite de message proposant d'ouvrir un navigateur dans le répertoire contenant les derniers documents récupérés si des téléchargements ont été effectués.

Connexion distante

Les commandes de connexion à un système distant ne sont pas légion : `telnet`, `ssh` sont les plus courantes. Les deux commandes permettent de solliciter des services de connexion sur un hôte, le premier n'assurant aucun cryptage des données relatives à la connexion. Il est ainsi possible de récupérer les

paramètres de connexions à partir de certains outils de surveillance du réseau !. . .Pour cette raison, les administrateurs sont fortement incités à remplacer les services classiques par des versions sécurisées et les utilisateurs sensibilisés à la sécurisation de leur échanges (et en particulier à utiliser `ssh` pour réaliser des connexions à distance).

La commande `ssh` permet d'obtenir une connexion sur un hôte en vue d'interagir le plus souvent par un shell. Toutefois, la commande `ssh` permet aussi d'exécuter une simple commande sur un hôte [2]. Il existe une commande, `rsh` qui permet également l'exécution distante de commande, mais comme pour telnet, aucun mécanisme de protection des données lors de la communication n'existe. Par ailleurs, cette commande est faite pour alleger les phases d'authentification lors des communications, ce qui est une faille potentielle de sécurité. A bannir donc.

 Utilisez la commande `ssh` pour vous connecter sur l'hôte de votre voisin.

 Utilisez la commande `ssh mai12345@minisfa w` pour voir qui est connecté sur l'hôte `minisfa`. Adaptez la commande pour voir qui est connecté sur le poste de votre voisin. Consultez la page du manuel de `ssh` afin d'identifier la syntaxe correcte si besoin.

 Adaptez votre commande pour lister tous les processus vous appartenant sur `minisfa`.

 Idem pour lister les fichiers exécutables que vous avez modifié depuis moins de deux semaines. Vous recherchez ces fichiers à partir de votre home.

 Vous avez remarqué avec les questions précédentes que l'exécution de vos commandes implique une étape d'authentification (i.e. la saisie du mot de passe associé au login utilisé). Existe-t-il un moyen de réaliser une authentification non-interactive ? Ceci permettrait d'utiliser la possibilité d'exécuter des commandes distantes au sein de script sans intervention de l'utilisateur. . .

 Quelle commande utiliser pour «tuer» tous les processus associés au votre login sur une machine distante ? Vous pourrez aborder le problème en vous limitant à l'affichage de tous les PIDs des processus à tuer. Le problème ici est de tuer tous les processus avant celui lancé à distance pour tuer tous les autres. . .

Transferts de fichiers

Diverses possibilités peuvent permettre à un utilisateur de transferer des fichiers entre deux hôtes. Tout d'abord, si les deux hôtes sont des systèmes unix sur lesquels l'utilisateur dispose d'accès, et si le système distant permet les connexions ssh, alors il peut recourir à `scp`, une commande de copie «enrobée» dans le protocole `ssh`. Cette commande prend en argument (au moins) deux paramètres : le (ou les) fichier(s) source(s) et la destination. Le principe est le même que la commande `cp`.

Si l'hôte distant accepte les connexions ftp (File Tranfert Protocol), alors l'utilisateur pour ouvrir une connexion ftp et utiliser les commandes de son client ftp pour récupérer des fichiers de l'hôte distant et pour placer sur l'hôte distant des fichiers locaux. La commande `sftp` devrait être préférée à `ftp` pour sa capacité de cryptage de la communication. `sftp` est utilisable dès que les connexions `ssh` sont acceptées [3].

Enfin, en marge de ces solutions classiques, l'usage du mail reste une solution simple pour placer des documents qui seront facilement accessible si l'accès au contenu de la boite mail l'est également. L'ajout de pièce jointe en ligne de commande est possible avec les versions récentes de la commande `mail` (cf `man nail`), mais sur de vieux systèmes, la transmission de fichier comportant des caractères non-imprimables n'est pas naturelle. Il est alors nécessaire d'utiliser la commande `uuencode` (resp. `uudecode`) pour inserer (resp. extraire) des pièces jointes au corps d'un mail.

scp

2. Remarquez que c'est ce qui se passe lorsqu'on interagit avec un shell : la commande invoquée est le shell de connexion !
3. Notez qu'il devient courant de remplacer les serveurs ftp par des serveurs ssh pour le degré de sécurité supplémentaire qu'ils apportent. `ftp` est l'ancêtre qui, comme `telnet`, a ouvert la voie à des services intéressants, mais qui présente trop de risque dans le contexte actuel des communications.

- Téléchargez le fichier `isfaserveur.univ-lyon1.fr/~denis.clot/IRM1.html` et renommez-le en lui préfixant votre login et la date au format HHMMSS, les différentes parties du nom étant séparées par des "-" (ce qui donne `bob-091053-...html`). Utilisez la commande `scp` pour copier ce fichier sur `minisfa` dans votre répertoire personnel.
- Utilisez `ssh` pour renommer le fichier transferé. Le nouveau nom resultera d'un simple ajout d'un préfixe formé par vos nom et prénom (e.g BobyLapointe). Vous pourrez utiliser la commande `id -P` et le mécanisme de substitution de commande.
- Utilisez `scp` pour récupérer le fichier renommé précédent sur votre poste.
- A l'aide de `ssh`, créez sur `minisfa` le répertoire `~/scpRepTest_VotreNomPrenom` puis utilisez `scp` pour copier vers ce répertoire tous deux fichiers html précédents.
- Depuis la machine du crip, testez la commande suivante :
  `scp mai12345@minisfa:scpRepTest_VotreNomPrenom/*.html .`
  Que réalise-t-elle et pourquoi ?

ftp/sftp

Il existe de nombreux clients ftp, graphiques pour la plupart. Nous nous intéressons aux clients en mode texte. Les commandes classiques qui devraient se trouver implémentées sur chaque client sont les suivantes :

- `open`, `bye`, `close`, `disconnect` qui sont des commandes relatives à la connexion ;
- `dir`, `cd`, `lcd` qui permettent d'obtenir des informations sur les répertoires courants (local et distant) et de se re-positionner ;
- `mkdir`, `delete` qui permettent respectivement de créer un répertoire et de supprimer un fichier ;
- `get`, `mget` qui permettent d'obtenir des fichiers distants ;
- `put`, `mput` qui permettent de transferer des fichiers locaux ;
- `binary`, `ascii`, `type` qui permettent de modifier le mode de transfert ;
- avec un peu de chance, `help` qui permet d'obtenir de l'aide sur les différentes commandes disponibles...

Notez qu'un client `sftp`, la version sécurisée de `ftp` et qui permet de faire les mêmes transferts, ne dialogue pas avec un serveur `ftp`, mais avec un serveur `ssh` (c'est la configuration standard). Ainsi, aucun serveur `ftp` n'est nécessaire dans ce cas : une machine pourvue d'un serveur `ssh` permet finalement beaucoup de choses...

- Identifiez le rôle des différentes commandes proposées ci-dessus, en donnant priorité aux commandes `open`, `disconnect`, `binary`, `cd` et `lcd`, `get` et `mget`, `put` et `mput`.
- Ouvrez un connexion `ftp` sur `minisfa` en mode binaire. Identifiez le répertoire dans lequel vous vous situez sur l'hôte distant. Positionnez-vous dans le répertoire `~denis/Public/Images/`. Listez le contenu de ce répertoire.
- Dans ce répertoire, récupérez tous les fichiers d'extention `jpg`. Vous visualiserez les images obtenues afin de contrôler la qualité des fichiers transférés. Pour cela, vous pourrez utiliser la commande `display` si elle est installée sur les machines du crip, ou plus classiquement votre navigateur web en ouvrant l'url `file:///home/x/p05343453/`, désignant votre home.
- Renommez les fichiers récupérés en remplaçant les extensions `jpg` par des extensions `JPG`, puis placez ces fichiers sur `minisfa` dans le répertoire `scpRepTest_VotreNomPrenom`.
- Connectez-vous avec `ssh` afin de supprimer le répertoire créé précédemment.
- Vous pourrez ultérieurement tester le client ftp de microsoft dans une fenêtre DOS...
- Dans le manuel, identifiez les principales commandes de la commande `sftp`.
- Procédez aux manipulations proposées pour `ftp` à l'aide de `sftp`. Quelles changements observez-vous par rapport à la commande `ftp` ?

mail

La commande `mail` permet d'envoyer des emails. Nous avons vu les différentes façons permettant de construire le corps d'un mail. Celui-ci est normalement constitué de texte.


Téléchargez les corrections des évaluations et fabriquez une archive compressée contenant les deux fichiers. Utilisez la commande `mail` pour vous envoyer cette archive. Vous testerez la validité de votre envoi en controlant que le document envoyé est bien lisible.


Envoyez-vous à présent un mail comportant les deux fichiers de correction. Contrôlez la validité de votre envoi.


Considérez à présent la commande `uuencode` pour construire un mail comportant un premier fichier. Testez la validité de votre envoi à l'aide de la commande `uudecode` qui permettra d'obtenir le fichier décodé.


Utilisez `uuencode` pour construire un mail comportant les deux fichiers. Evidemment, vous utiliserez `uuencode` pour chacun des fichiers. Testez la validité de votre envoi.