Configuration d'une station X-Window avec xdm*

Nous examinerons dans cet article un mécanisme puissant permettant d'utiliser un PC sous Linux comme station X-Window capable de gérer l'affichage de différentes machines se trouvant sur le même réseau. Ce système, s'appuie sur le démon xdm, bien connu dans les environnements où l'on emploie fréquemment des terminaux X (bureaux d'études, laboratoires, etc.) Malheureusement, les possibilités de xdm sont nettement moins exploitées sur les stations Linux courantes ; espérons que l'on trouvera ici quelques idées pour en tirer profit.

Je travaille souvent chez des clients ayant des systèmes très divers, et même lors d'une intervention sur un site unique (par exemple un aéroport), il me faut fréquemment agir sur des machines réparties en différents lieux. Je suis donc amené à transporter régulièrement des fichiers-source, des exécutables, des bibliothèques ainsi que des fichiers de données ou de documentation. Pour simplifier cette tâche, je me suis équipé d'un ordinateur portable que j'utilise comme poste de travail principal. Toutefois, pour des sessions de développement un peu longues, l'utilisation de l'écran LCD et du clavier relativement restreint est assez pénible. De retour à mon bureau je préfère donc travailler sur un terminal '' classique '', avec un moniteur cathodique et un clavier habituel.

Je pourrais simplement connecter un écran et un clavier sur les sorties d'extension du portable, mais sur ce lieu de travail, je dispose également de plusieurs PC sous Linux réservés à des tâches d'enregistrement de données, supervision, filtrage, etc. Ces machines sont regroupées côte à côte sur une étagère et, ne nécessitant pas d'intervention en fonctionnement normal, elles n'ont ni écran ni clavier. J'ai donc décidé d'utiliser un poste de travail central fonctionnant sous X-Window, et me permettant de me connecter sur n'importe quelle station ou sur mon portable. Pour cela, il est possible de configurer un simple PC sous Linux, qui se comportera comme un terminal X, et permettra de déporter l'affichage de n'importe quelle machine connectée au réseau local.

Serveur X-Window et applications clientes

Les distributions Linux actuelles proposent dès l'installation de configurer la machine afin qu'elle fonctionne directement en mode graphique sous X11. Ceci est très agréable, mais induit, dans l'esprit des nouveaux utilisateurs, une certaine confusion entre trois éléments pourtant bien distincts :

- Le noyau et les applications-système : il s'agit de programmes fonctionnant essentiellement en mode texte. On emploiera par exemple ici les possibilités réseau offertes par le noyau. Pour observer le comportement de ces utilitaires seuls, on peut initialiser le système dans un mode non-graphique en invoquant la commande `init` suivie du niveau désiré – par exemple 3 sur les distributions Red-Hat ou Slackware afin de disposer de toutes les fonctionnalités réseau mais pas de mode graphique.

- Le serveur X11 : ce programme est capable de gérer la carte vidéo de la machine, ainsi que les périphériques d'entrée comme le clavier ou la souris, mais également des tables à digitaliser, boule roulante, etc. Lorsqu'un système Linux a été initialisé en mode non-graphique, on peut faire démarrer le serveur X11 en invoquant la commande `X` seule. On voit alors que le serveur est prêt, qu'il initialise le mode vidéo et le fond d'écran, et qu'il gère la souris. Mais c'est tout ! Aucune application cliente n'est présente. On peut tuer le serveur X11 en pressant Ctrl-Alt-Backspace (la flèche d'effacement se trouvant au-dessus de la touche Entrée).

- Les applications clientes X11 : cette fois-ci il s'agit de tous les programmes susceptibles d'envoyer des requêtes graphiques à un serveur X, et de recevoir en retour des notifications d'évènements (mouvement de souris, pression de touche). Les applications clientes comprennent également le gestionnaire de fenêtres, et les environnements graphiques complets comme KDE ou Gnome.

Pour observer ces trois éléments distinctement on peut effectuer la manipulation suivante :

On initialise le système en mode texte. Sur les distributions Red-Hat ou Slackware, il faut invoquer '' `init 3` '' (en étant root). Pour les autres distributions on examinera les commentaires contenus dans `/etc/inittab`. Une fois l'initialisation réalisée, nous n'avons plus accès qu'aux applications-système hors X-Window.

En étant connecté sur un terminal virtuel on démarre le serveur X11 en invoquant la commande '' `X &` ''. Le serveur X-Window est un processus comme un autre, mais il peut accéder directement à la mémoire vidéo de la machine. Les applications clientes peuvent le contacter par le biais de sockets réseau, et lui envoyer des requêtes graphiques suivant le protocole X.

---

* Cet article a été publié dans le numéro 21 (Octobre 2000) de Linux Magazine France.

Revenant sur le terminal virtuel en mode texte avec Ctrl-Alt-F1, nous pouvons alors lancer une application X11. On se contentera de ‘‘ `xterm –display :0.0` ’’. Lorsque l’on retourne sur l’écran du serveur X avec Ctrl-Alt-F7, on observe alors la présence d’une fenêtre X-Term, prête à recevoir nos ordres. Comme nous n’avons pas lancé de gestionnaire de fenêtres (Xfce, Kde, Fvwm, Gnome, etc.), il n’y a aucune décoration autour du terminal X-term. L’argument ‘‘ `-display :0.0` ’’ sert à indiquer où se trouve le serveur X11 à contacter pour l’affichage. Lorsque nous appelons `xterm`, rien ne nous oblige en effet à l’envoyer sur le serveur X que nous venons de démarrer. C’est la force du protocole X-Window d’autoriser l’exécution d’une application cliente sur une machine totalement distincte de celle sur laquelle s’exécute le serveur X11.

Connexion en mode graphique avec xdm

Lorsque la machine démarre directement en mode graphique, (avec un niveau d’exécution 5 sur une Red-Hat par exemple ou 4 sur une Slackware), un composant supplémentaire vient gérer la connexion de l’utilisateur. Il s’agit d’un programme nommé `xdm` (*X Display Manager*), lancé automatiquement par le processus `init` lors de la lecture de `/etc/inittab`. On y rencontre en effet une ligne du genre (Red-Hat 6.1) :

```
x:5:respawn:/etc/X11/prefdm –nodaemon
```

Cela signifie que lors d’une initialisation au niveau 5, `init` doit lancer en boucle (`respawn`) l’application `prefdm`, en la redémarrant sitôt qu’elle se termine. Cette ligne est dotée d’un identificateur `x` servant à distinguer les messages qui la concernent dans les traces d’exécution de `init`. En réalité, `prefdm` est un simple script shell qui se base sur les préférences de l’administrateur, exprimées dans le fichier `/etc/sysconfig/desktop`, pour lancer soit `xdm`, soit `kdm` ou `gdm` qui en sont des équivalents fonctionnant dans le style Kde ou Gnome.

Sur une distribution Slackware, on trouve dans `/etc/inittab` une ligne :

```
x1:4:wait:/etc/rc.d/rc.4
```

Le choix du gestionnaire d’affichage se fait donc au sein de ce script `/etc/rc.d/rc.4`.

Rappelons que le choix du niveau de démarrage automatique de la machine s’effectue aussi dans le fichier `/etc/inittab`, grâce à la ligne :

```
id:5:initdefault:
```

qui contient le niveau désiré.

Lorsqu’il démarre, `xdm` consulte des fichiers de configuration se trouvant dans le répertoire `/usr/X11R6/lib/X11/xdm`. Il s’agit souvent d’un lien symbolique sur `/etc/X11/xdm`, car la partition `/usr` doit pouvoir être partagée entre différentes machines, alors que ces fichiers de configuration sont – comme tout ce qui réside dans `/etc` – propres à une station donnée. Les fichiers qui nous concernent sont nommés `Xaccess`, `Xservers`, et `xdm-config`. On peut signaler également que les fichiers `Xresources` ou `Xsetup_0` peuvent permettre de configurer l’aspect graphique de l’écran de connexion.

Dans `xdm-config`, on rencontre essentiellement les chemins d’accès pour trouver les autres fichiers de configuration. Quand `init` lance `xdm`, grâce à la ligne de `/etc/inittab` décrite plus haut, aucun serveur X ne fonctionne sur la machine. Le contenu du fichier `Xservers` permet donc à `xdm` d’en démarrer un :

```
:0  local  /usr/X11R6/bin/X –bpp 16
```

Cette ligne indique la ligne de commande pour lancer X, avec ses éventuelles options (ici `–bpp 16` pour avoir une palette de 65536 couleurs). Cela définit un affichage local identifié par le numéro d’écran `:0`. Ensuite, `xdm` propose une boîte de connexion login/password et prend en charge la validation du mot de passe pour ouvrir une session de travail pour l’utilisateur.

Ce que nous désirons ici, c’est employer `xdm` pour qu’il nous permette de nous connecter de manière identique sur les différentes stations présentes sur le réseau. Ceci est rendu possible par l’emploi de requêtes XDMCP (*XDM Control Protocol*) transitant entre les applications `xdm` se trouvant sur les différentes machines.

Connexion sur une machine distante

Nous allons nous intéresser ici à trois machines différentes :

- Un ordinateur portable (*venux*) fonctionnant avec un niveau d’exécution offrant une boîte de connexion graphique `kdm`. Il s’agit d’une version de `xdm` pour Kde, avec des possibilités directes de réinitialisation ou d’arrêt de la machine. Ce poste doit fonctionner de manière totalement indépendante des autres stations du réseau.

- Une machine de production (*gimli*) sans écran ni clavier. Auparavant nous n’utilisions que des connexions en mode texte depuis le réseau (par `telnet` ou `rlogin`), mais nous voudrions à présent disposer d’une connexion graphique. La carte vidéo de cet ordinateur ne permet pas de faire fonctionner X-Window de manière satisfaisante, aussi ce poste devra donc employer `xdm` pour autoriser les connexions externes, mais n’offrira pas de serveur X11 local.

- Un poste de travail central (*tracy*) permettant de se connecter graphiquement sur n’importe laquelle des deux stations décrites ci-dessus, ainsi naturellement qu’une connexion sur son propre système.

On l’imagine aisément, ces trois stations seront configurées différemment, la plus complète étant la dernière. Tout d’abord intéressons-nous à l’ordinateur portable. La seule intervention à assurer sur cette machine consiste à indiquer à `xdm` que nous acceptons qu’un serveur X11 distant puisse se connecter sur la station. Pour cela nous devons éditer le fichier de configuration `Xaccess`, et nous assurer que la ligne suivante n’est pas en commentaire :

```
*           # any host can get a login window
```

Suivant les distributions, cette ligne peut en effet être ou non précédée d’un dièse # qui la désactive. L’astérisque *

autorise la connexion depuis n'importe quelle station. Si on désire restreindre cette possibilité, on consultera les commentaires présents dans le fichier `Xaccess` pour limiter la liste des hôtes tolérés. Naturellement, si on doit modifier ce fichier, il faut relancer `xdm`. Cela peut être réalisé avec ‘‘ `init 3 ; init 5` ’’ ou ‘‘ `init 3 ; init 4` ’’.

Pour notre seconde machine, *gimli*, nous devons naturellement réaliser le même travail dans `Xaccess`, mais de plus nous devons empêcher `xdm` de lancer systématiquement un serveur X11 sur la station. Pour cela, on agit dans le fichier `Xservers`, en commentant à l'aide d'un dièse la ligne suivante :

```
# :0  local  /usr/X11R6/bin/X
```

Nos deux ordinateurs sont donc à présent prêts à accepter une requête XDMCP pour offrir une connexion sur un serveur X distant. Il ne nous reste plus qu'à configurer la station graphique principale *tracy* qui viendra les interroger. Ici nous devons réaliser plusieurs travaux :

Tout d'abord il nous faut scinder le lancement du serveur X11 de celui de `xdm`. Pour cela nous commentons – comme sur notre machine aveugle *gimli* – dans `Xservers`, la ligne :

```
# :0  local  /usr/X11R6/bin/X
```

Ainsi `xdm` ne lancera pas le serveur, nous le ferons nous-même un peu plus loin. Nous voulons accepter une connexion depuis le processus `xdm` fonctionnant sur la machine *tracy* elle-même, aussi nous assurons-nous que la ligne :

```
*          # any host can get a login window
```

de `Xaccess` n'est pas en commentaire. De surcroît, nous voulons que `xdm` envoie régulièrement une requête XDMCP *broadcast*, diffusée à toutes les stations du réseau local, et qui leur demande s'ils acceptent les connexions. Ce comportement est activé en vérifiant que la ligne :

```
*    CHOOSER BROADCAST    # any indirect [...]
```

n'est pas en commentaire dans `Xaccess`. Comme toujours on peut restreindre la liste des stations qui seront interrogées si on le désire.

À présent il faut quand même lancer le serveur X11, indépendamment de `xdm`. Nous ajoutons donc une ligne supplémentaire dans `/etc/inittab`, à la suite du lancement de `xdm` :

```
x2:5:respawn:/usr/bin/X11/X –indirect tracy
```

On peut ajouter les options nécessaires (par exemple `–bpp 16`). L'option `–indirect` demande au serveur X d'interroger par une requête indirecte le processus `xdm` fonctionnant sur la machine `tracy` (elle-même). Celle-ci répercutera alors la requête en mode *broadcast* sur le réseau, et proposera la liste des machines prêtes à accepter une connexion. Sur les distributions de type Slackware, le second champ doit contenir le niveau d'exécution 4.

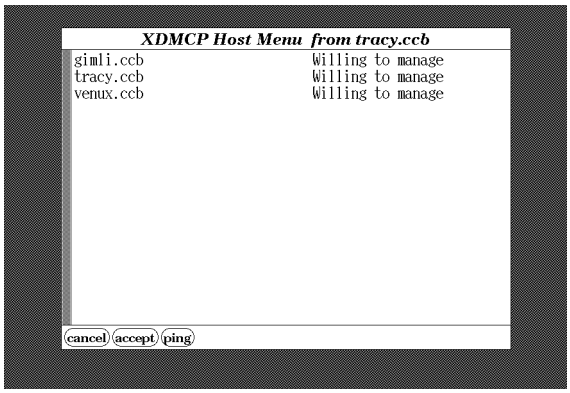
Résultats

Finalement, qu'avons-nous obtenu ? Trois configurations différentes :

- Sur le portable, les modifications sont totalement transparentes. Nous avons dès le démarrage un écran de connexion Kde, et la machine fonctionne de manière autonome.

- La station Gimli ne présente pas non plus de différence, elle se trouve toujours sur son étagère, sans clavier ni écran !

- Le poste central, par contre a un comportement nettement plus intéressant. Dès son démarrage, il présente un écran permettant de choisir la station où l'on désire se connecter. La liste est obtenue en envoyant une requête XDMCP diffusée en mode *broadcast* sur le réseau. Les stations présentes répondent, et leurs noms s'affichent dans la boîte de choix. Un bouton ‘‘ *ping* ’’ permet de renvoyer une nouvelle requête – par exemple après avoir rebranché le portable sur le réseau ou après avoir allumé une autre station.



Une fois que l'on a choisi une machine, la boîte de connexion login/password habituelle est affichée. Remarquons que la présentation par défaut de l'écran de choix des stations n'est pas des plus esthétiques, mais qu'on peut l'améliorer en modifiant les caractéristiques décrites dans le fichier `Xresources` de `xdm`.

Xdm, Kdm, Gdm, Login.app etc.

Nous avons presque toujours parlé de l'utilitaire `xdm` original, livré avec les outils X11 du MIT. Néanmoins, les distributions Linux actuelles proposent des outils de remplacement plus attractif au niveau graphique, notamment `kdm` et `gdm` associés aux environnements Kde et Gnome, ou encore `Login.app` qui fait partie du projet WindowMaker. Attention, seules sont concernées les boîtes de dialogue permettant la connexion de l'utilisateur. L'environnement de travail lui-même n'est aucune perturbé par le déport de l'affichage sur un serveur X11 distant.

Qu'en est-il donc de la configuration de ces utilitaires ?

Tout d'abord commençons par les bonnes nouvelles : `kdm` est utilisable de manière totalement transparente en remplacement de `xdm`. Les utilisateurs préférant l'aspect de l'interface Kde pourront ainsi l'utiliser sur n'importe quelle machine. Cela peut être réalisé en remplaçant l'invocation de `xdm` par celle de `kdm` dans `/etc/inittab` ou, si votre

système emploie le script `/etc/X11/prefdm`, en appelant :

```
echo KDE > /etc/sysconfig/desktop
```

L'utilitaire `gdm` par contre ne permet pas de diffuser les requêtes XDMCP indirectes en mode *broadcast*. Les auteurs espèrent pouvoir intégrer cette fonctionnalité dans la prochaine version. Pour l'instant on ne peut donc pas utiliser `gdm` sur la station graphique centrale *tracy*. En revanche, on peut parfaitement l'employer sur le portable *venux* ou sur la machine aveugle *gimli*. Dans le cas de cette dernière, on obtiendra la boîte de login Gnome uniquement lors d'une connexion distante puisqu'il n'y a pas de serveur X11 sur cette station.

Comme pour `kdm`, on peut remplacer l'appel de `xdm` par celui de `gdm` dans `/etc/inittab` ou utiliser, sur une machine utilisant `prefdm` :

```
echo GNOME > /etc/sysconfig/desktop
```

Il faut toutefois indiquer à *gdm* qu'il doit accepter les requêtes XDMCP. Cela s'effectue en configurant les deux options suivantes dans `/etc/X11/gdm/gdm-conf` :

```
[xdmcp]
Enable=1
HonorIndirect=1
```

Enfin, en ce qui concerne `Login.app`, il faut comprendre que cette application n'est qu'un encadrement graphique esthétique d'une boîte de dialogue de connexion, mais ne représente pas du tout un gestionnaire d'affichage complet. Cet utilitaire ne connaît pas les requêtes XDMCP, et ne permet donc pas de connexion distante. Il est prévu pour servir uniquement sur une station isolée, où applications et serveur X11 tournent sur la même machine. On ne pourra donc pas l'employer ici.

Conclusion

Nous avons survolé ici quelques aspects très performants du système X-Window, mais `xdm` offre encore de nombreuses autres possibilités, notamment en ce qui concerne la gestion des terminaux X ‘‘ purs ’’. Pour plus de renseignements, on pourra consulter à profit l'ouvrage ‘‘ X Window System Administrator's Guide ’’, par Linda Mui et Eric Pearce aux éditions O'Reilly & Associates.

Christophe Blaess < ccb@club-internet.fr>

http://perso.club-internet.fr/ccb/