05 août 2008

Créez un terminal X

Catégorie : News Tags : lmhs

~~Retrouvez cet article dans : Linux Magazine Hors série 21~~

Une machine de récupération sans disque dur est souvent mise de côté «pour pièces». Elle pourrait pourtant devenir un poste tout à fait utilisable en profitant des avantages dont dispose n'importe quel système UNIX et le système X Window.

Les capacités réseau et client/serveur sont profondément enracinées dans les systèmes UNIX et ce, depuis leur apparition et leur montée en puissance depuis plus de 25 ans (une éternité à l'échelle informatique). Il en va de même pour le système graphique dont l'idée a germé dans les laboratoires Xerox en 1973. C'est en 1984 que le X Window System fait son apparition et, dès lors, il est client/serveur.
La plupart des distributions GNU/Linux ne configurent pas XFree86, l'une des implémentations libres du X Window System, de manière à autoriser le fonctionnement client/serveur pour des raisons de sécurité. Cependant, une simple modification de la configuration et le tour est joué. Dans le cas de Debian Sarge, par exemple, il faut chercher l'option ~~nolisten~~ passée au serveur X et la supprimer (dans /~~etc/X11/xinit/xserverrc~~ ou /~~etc/X11/?dm/Xservers~~). XFree86 permet ainsi, localement et via le réseau, de lancer des applications X11 sur un serveur et d'exporter l'affichage sur un client. En d'autres termes, vous pouvez lancer, par exemple, The Gimp sur une machine A, voir et utiliser son interface sur une machine B.

Il en va de même si vous souhaitez ajouter d'avantage de sécurité en faisant transiter tous les pixels et les évènements X dans un tunnel SSH via l'option ~~-X~~ de la commande ~~ssh~~ et l'option ~~X11Forwarding~~ de votre ~~sshd_config~~.
Cela est rendu possible par le fait que XFree86 est conçu de base pour le permettre ; il n'existe quasiment pas de limite dans les capacités client/serveur du système. C'est ce que va démontrer cet article de manière ludique et concrète.

Xnest permet de lancer un pseudo-serveur X dans une fenêtre. Très pratique sur bien des points

Jouons un peu avec X11

Imaginons deux utilisateurs, Alice et Bernard. Alice a lancé X11 et est en train d'utiliser le système. Pour une raison personnelle, elle doit utiliser le compte de Bernard. De ce fait, dans un ~~xterm~~ elle fait :

```
$ su bernard
Password:

$ echo $DISPLAY
:0.0

$ xclock
Xlib: connection to „:0.0“ refused by server
Xlib: No protocol specified
Error: Can't open display: :0.0
```

Sous l'identité de Bernard, Alice ne semble pas pouvoir lancer l'application X. La variable d'environnement ~~DISPLAY~~ définie par ~~su~~ correspond au display «possédé» par Alice. Bernard n'a tout simplement pas le droit de l'utiliser.
Une méthode « barbare » pour régler le problème est d'utiliser ~~xhost~~ sous l'identité d'Alice afin d'autoriser l'utilisation du display depuis un hôte donné. Ici, il s'agira du localhost : ~~xhost +localhost~~. Dès lors, n'importe quel utilisateur de la machine pourra utiliser le display0.0. Cette solution n'est cependant pas acceptable et manque de finesse puisqu'il n'est pas possible de traiter les utilisateurs individuellement. Vous avez ici une première démonstration des capacités client/serveur du système X Window. Un utilisateur peut faire usage d'un display X appartenant à un autre utilisateur. Et ce, même au travers d'un réseau.
Il existe un autre moyen d'authentifier les connexions. Le principe est celui du partage de secret. Le secret en question est une valeur de 128 bits représentée en hexadécimal et appelée MAGIC-COOKIE. Si le cookie est partagé entre l'utilisateur Alice et Bernard, l'utilisation du display est possible, sinon elle est refusée.
La gestion des cookies se fait via la commande ~~xauth~~. Celle-ci peut être utilisée de manière interactive ou via passage de paramètres. Les cookies eux-mêmes sont stockés dans le ~~~/.Xauthority~~ de chaque utilisateur, mais ce fichier n'est utilisable que via ~~xauth~~.

En tant qu'Alice, nous pouvons très simplement lister les cookies en notre possession :

```
$ xauth list
morgane/unix:0  MIT-MAGIC-COOKIE-1
  116c0c35226f6b112b293071002c3d4b
```

Cette seule entrée suffit à autoriser Alice à lancer des applications graphiques. Elle possède un seul cookie aussi bien pour l'aspect client que serveur. Bernard, quant à lui, ne possède pas de cookies. C'est un compte utilisé exclusivement en mode console et aucune session X n'a jamais été ouverte. La syntaxe d'une entrée ~~xauth~~ est relativement simple : display protocole clef. Le protocole sera ici ~~MIT-MAGIC-COOKIE-1~~ mais il est possible d'en utiliser d'autres. La désignation du display peut se décliner en trois versions :
:~~D.S~~ où ~~D~~ et ~~S~~ sont respectivement le numéro de display et le numéro d'écran. Ici on ne spécifie pas l'hôte qui sera implicitement localhost.

- ~~nom_hôte:D.S~~ qui est très proche de ma syntaxe précédente mais où l'on précise explicitement l'hôte concerné.
- ~~hôte/unix:D.S~~ correspond à l'utili-sation d'un socket placé dans ~~/tmp/.X11-unix/~~. C'est une source fréquente de confusion et de problèmes car, en toute logique, le socket n'est accessible que depuis l'hôte lui même.

Avec xauth, sous l'identité de Bernard, il suffit d'ajouter le cookie d'Alice pour autoriser la connexion :

```
$ su bernard
Password:

$ xauth add morgane/unix:0  \
MIT-MAGIC-COOKIE-1 \
116c0c35226f6b112b293071002c3d4b
xauth: creating new authority
   file /home/bernard/.Xauthority

$ xclock
...
```

La clef de 128 bits a simplement été copiée/collée. Notez cependant que vous pouvez générer vos propres clefs à l'aide de la commande ~~mcookie~~ et ajouter les entrées dans le ~~~/.Xauthority~~ d'Alice et Bernard.
Si l'on souhaite lancer des applications X depuis une autre machine où Bernard possède un compte tout en utilisant le display d'Alice sur une autre machine, Bernard n'aura qu'à ajouter une entrée :

```
$ xauth add morgane:0 \
MIT-MAGIC-COOKIE-1 \
116c0c35226f6b112b293071002c3d4b
```

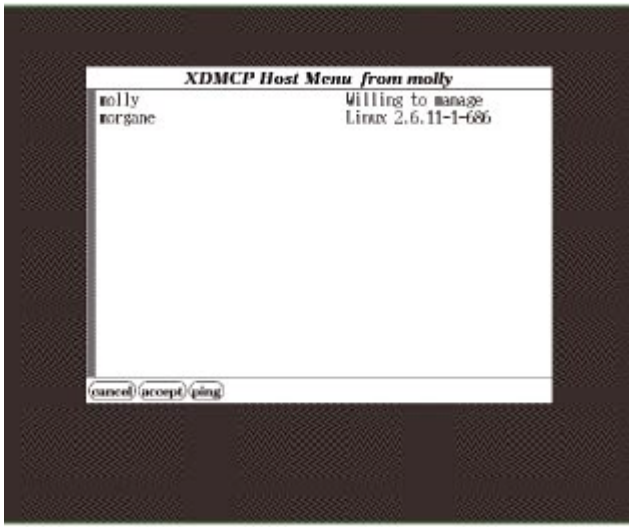
Puis à préciser le contenu de la variable d'environnement ~~DISPLAY~~ :

```
 $ export DISPLAY=morgane:0
```

Ou en version Shell C :

```
$ setenv DISPLAY morgane:
```

Vous l'aurez compris, ces infor-mations nous sont déjà utiles. En effet, une machine un peu «vieille» ne possèdera peut-être pas un adaptateur graphique ou un écran de qualité (contrairement aux machines elles-mêmes, les moniteurs restent des périphériques invariablement coûteux et fragiles).
Cependant, en profitant des capacités client/serveur du système X Window, nous pourrions parfaitement y copier nos photos de vacances et installer des applications graphiques. Nous pourrons ensuite utiliser les ressources et les données de la machine recyclée tout en profitant de l'interface graphique de la machine de travail courante. Il ne s'agit là que d'un exemple ; on peut également imaginer l'utilisation d'applications graphiques plus «sérieuses» pour faire de la surveillance ou de la supervision.
Notez enfin, mais je reviendrai sur ce point en détail plus loin, qu'il existe plusieurs systèmes X Window pour MS Windows (95/98/NT/XP). Ceux-ci fonctionnent en surcouche de l'interface graphique Microsoft et permettent l'utilisation d'applications X locales ou distantes.



Le chooser par défaut de XDM. Graphiquement pas très moderne mais parfaitement fonctionnel

Xnest, un X dans X

Xnest est un élément important lorsque l'on met en place une configuration client/serveur autour de X Window.
Il s'agit d'un serveur X fonctionnant comme une application cliente. Il s'agit d'un serveur car il crée un nouveau display et il s'agit d'une application cliente car l'affichage utilise un display X existant.
Rien de tel qu'un exemple pour mettre les idées en place. Assurez-vous d'avoir ~~Xnest~~ d'installé (paquet Debian ~~xnest~~) et depuis un ~~xterm~~, par exemple, lancez :

```
$ Xnest -kb :1
```

En fonction des paramètres par défaut, vous obtenez une fenêtre d'une taille plus ou moins importante contenant le tramage caractéristique d'un serveur X. Vous pouvez ensuite lancer des applications utilisant le nouveau display, comme, par exemple un ~~xclock~~, depuis un autre ~~xterm~~ en spécifiant le display cible :

```
$ xclock -display :1
```

L'utilisation de l'option -display nous évite de définir ou changer le contenu de la variable d'environnement DISPLAY. Dès la commande validée, la petite horloge apparaît dans la fenêtre Xnest. Xnest utilise ses valeurs par défaut s'il est exécuté sans aucun paramètre autre que l'option -kb pour désactiver l'extension clavier et le display à créer. Il est cependant possible de spécifier une résolution pour ce nouveau display (-geometry) et d'utiliser quasiment les mêmes paramètres que pour le serveur X standard.

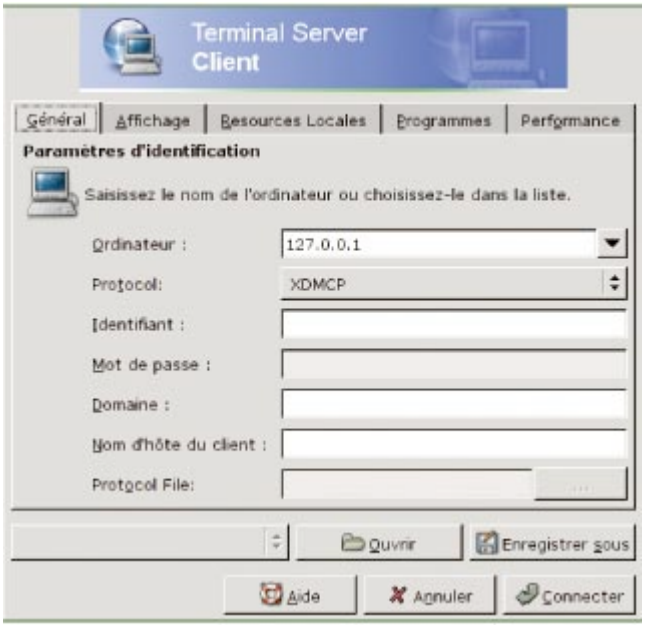
Xnest est donc l'outil idéal pour procéder à des tests. Concernant l'authentification des utilisateurs vous pouvez choisir :

- De laisser faire Xnest en autorisant, via xhost, les connexions locales uniquement ;
- De désactiver le contrôle en autorisant tous les hôtes à utiliser le nouveau display via une option -ac ;
- De spécifier un fichier de type ~/.Xauthority via l'option -auth et ainsi gérer pleinement et finement les authentifications. Notez que xauth permet également la manipulation d'un fichier autre que le classique ~/.Xauthority en utilisant une option -f.

Un simple Xnest -help devrait vous apprendre tout ce qu'il faut savoir sur les options utiles. Grâce à Xnest nous allons pouvoir procéder à quelques essais de configuration de teminal X sans avoir à utiliser une seconde machine.

XMDCP

Vous connaissez sans doute XDM, le X Display Manager ou l'une de ses évolutions comme GDM (Gnome), KDM (KDE), WDM ou encore Login.app. XDM permet le plus couramment d'obtenir un login graphique local. En d'autres termes, lorsque le système démarre, il lancera automatiquement le Display Manager qui lancera le serveur X et l'utilisera pour présenter une fenêtre spécifique. Ce dernier procèdera à l'authentification de l'utilisateur et ouvrira une session en son nom en utilisant le gestionnaire de fenêtres de son choix.



Le Terminal Server Client de Gnome permet un accès simplifié aux serveurs RDP, VNC, mais

aussi XDMCP

Dans ce cas, le Display Manager lance et gère le serveur X local mais il peut faire bien mieux. XDM, ou équivalent, peut gérer un serveur X distant. Comprenez bien que le Display Manager est en attente de connexion sur le port 177, c'est donc un serveur. Cependant, pour pouvoir afficher des pixels, il doit accéder à un serveur X et donc se comporter en client. Dans ce dernier cas de figure, le protocole utilisé est XDMCP ou X Display Manager Control Protocol.

Lorsque vous installez ou configurez un Display Manager, un certain nombre d'éléments de configuration sont définis par défaut. Parmi tous ces éléments nous trouvons le fait que XDM lance automatiquement un serveur X local au démarrage. Ceci est certes pratique puisque l'objectif le plus souvent poursuivi est d'obtenir un login graphique, mais ce n'est pas obligatoire.

XDM et WDM partagent un schéma de configuration identique. GDM, quant à lui, sort un peu du lot avec un fichier de configuration totalement différent.

XDM et WDM

XDM est le plus ancien Display Managers dont WDM découle. Le modèle de configuration est le même pour les deux programmes. On préfèrera cependant WDM, qui est un peu plus moderne en termes d'aspect et offre plus de souplesse pour l'utilisateur en proposant le choix du gestionnaire de fenêtres par exemple.
Peu de fichiers sont concernés pour la configuration d'une machine en serveur XDM/WDM. J'utiliserai ici comme configuration de référence celle de WDM. Pour l'adapter à XDM, vous n'aurez qu'à remplacer une lettre dans toutes les occurrences de wdm dans les chemins et noms de fichiers. Les fichiers de configuration de WDM sont placés dans ~~/etc/X11/wdm~~. Vous trouverez ici :

- ~~Xserver~~ qui contient les informations pour le lancement du ou des serveurs X locaux. Avec le système utilisé pour cet article, le fichier contenait d'origine la ligne :

```
:0 local /usr/bin/X11/X -nolisten TCP
```

Celle-ci est destinée à lancer le binaire X avec une option interdisant l'acceptation de connexions clientes. On remarquera que les différences d'options passées au binaire X sont souvent sources de confusion. Je pense naturellement à l'option ~~-dpi 100~~ présente lors d'un lancement par ~~startx/xinit~~ et absente de la configuration de certains Dis-play Managers. Ceci provoque habituellement des problèmes dans l'affichage des polices de caractères selon qu'on utilise ou non le Display Manager.
Cette ligne n'est utile que si vous souhaitez exécuter un serveur X local. Si le but de la manœuvre est de disposer d'un système X partiel (bibliothèques et applications mais pas de serveur X), vous pouvez parfaitement placer cette ligne en commentaire en la faisant débuter par un dièse. En temps normal, le protocole XDMCP permet de découvrir et de gérer les serveurs X disponibles. Il est possible cependant que certains terminaux n'utilisent pas ou mal ce protocole. Il faut alors spécifier manuellement dans le fichier ~~Xserver~~ les serveurs X disponibles par «ailleurs» :

```
molly:0 foreign
pao:0 foreign
```

La syntaxe est simple. On reconnaît le couple hôte/display suivi de la directive ~~foreign~~. Ceci ne

sera normalement pas nécessaire dans le cadre de cet article qui traite, justement, de XDMCP.

- ~~Xaccess~~ permet, comme son nom l'indique, de contrôler les connexions entrantes. On peut utiliser un motif représentant les hôtes dont on accepte les connexions. Inutile de vous dire que le plus souvent, ce motif est * autorisant les connexions de n'importe quelle provenance. Selon l'architecture réseau et la politique de sécurité utilisée cela peut être acceptable ou complètement stupide.

Une autre solution est de spécifier une liste d'hôtes. Pour cela, il suffit de lister ces derniers les uns à la suite des autres. Il est également possible d'explicitement interdire la connexion depuis un hôte en faisant précéder la ligne d'un point d'exclamation (signifiant la négation). Les motifs sont évalués dans leur ordre d'apparition, c'est la première correspondance vérifiée qui déterminera la réponse. Attention donc à l'ordre des lignes dans ce fichier.
Bien sûr, ce type de sécurité n'est valable que dans un environnement non hostile. Le spoofing est une technique très facile à mettre en œuvre dans ce type de configuration. On devra sérieusement réfléchir à une solution de tunneling ou de VPN en cas d'utilisation d'un réseau public ou d'Internet.
Arrêtons là la description de ce fichier. Nous y reviendrons plus tard en parlant des connexions indirectes et du chooser.

- ~~wdm-config~~ est tout simplement le fichier de configuration du Display Manager. Il regroupe un nombre important d'options. Fort heureusement, la plupart d'entre elles sont déjà renseignées de manière à garantir un fonctionnement minimal par défaut. On notera toutefois, dans le cas de la distribution Debian (et peut-être d'autres), la présence de la ligne :

```
DisplayManager.requestPort:   0
```

Cette directive permet de spécifier le port d'écoute. Si la valeur 0 est définie, l'utilisation du protocole XDMCP est tout simplement désactivée. Assurez-vous donc de commenter cette ligne afin d'autoriser les connexions au port UDP 177 utilisé par défaut.

Vous pouvez également jeter un œil à la page de manuel de WDM (ou XDM) pour obtenir une liste complète des directives pour le fichier de configuration et une description précise de leur utilité.

Le chooser et les connexions indirectes

Lorsqu'on dispose de plusieurs machines faisant fonctionner des Display Managers en attente de connexions, il devient vite lassant de devoir spécifier l'hôte cible à chaque connexion. Ceci est d'autant plus vrai lorsqu'on parle de terminaux X puisque dans ce cas la simplicité doit prévaloir. Pour ce type de situation (un ou plusieurs terminaux, et plusieurs serveurs *DM) une solution a été prévue.

Ainsi, plutôt que d'accéder directement (et je pèse mes mots) à un hôte spécifique, nous pouvons faire appel à un chooser. Il ne s'agit ni plus ni moins qu'une boîte de dialogue présentant les différents serveurs *DM en présence sur le réseau et permettant d'en choisir un.

La sélection faite, on accède alors à l'hôte comme nous l'aurions fait manuellement.

Le chooser est exécuté sur l'un des hôtes *DM. Ceci implique que le client doit, de toutes façons, accéder à un serveur cible. Ce dernier le réorientera éventuellement vers un autre serveur en lui proposant une liste de serveur *DM. Le chooser peut être configuré de plusieurs manières. Soit nous définissons manuellement une liste de serveurs *DM connus avec (dans le fichier ~~Xaccess~~) :

```
%lalist molly morgane
*   CHOOSER %lalist
```

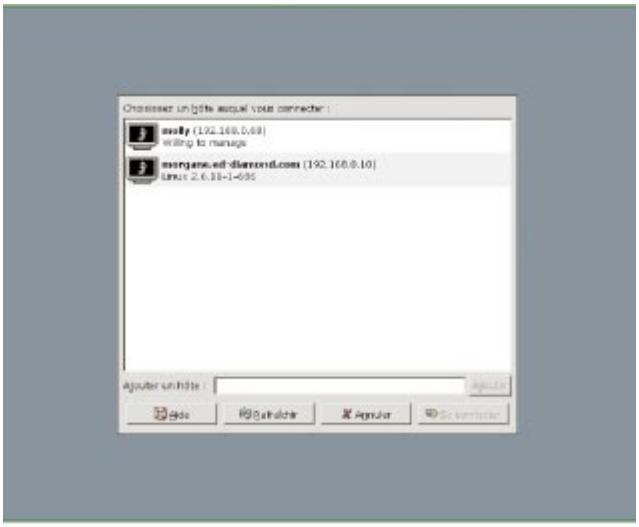
Le symbole ~~%~~ permet de définir une macro. Ici ~~%lalist~~ sera donc équivalent à molly ~~morgane~~. La seconde ligne précise que n'importe quel hôte distant accédant de manière indirecte provoquera le lancement du chooser lui présentant les hôtes *DM ~~molly~~ et ~~morgane~~. Voyez comme cela peut être modulaire.

En définissant plusieurs macros, il est possible de créer des profils regroupant des serveurs. Il est ensuite possible de ne proposer aux clients qu'un nombre limité de serveurs et ainsi gérer des sous-réseaux et des postes spécifiques de manière très fine.

Une autre solution pour obtenir une telle liste de manière plus «dynamique» est de demander au chooser d'envoyer un message en broadcast sur le réseau. Il retournera ensuite les différents serveurs disponibles dans la liste proposée à l'utilisateur :

```
*   CHOOSER BROADCAST
```

Petite remarque concernant WDM. Le chooser associé avec ce Display Manager n'existe pas ou est encore considéré comme étant «loin d'être terminé» (dixit un post de la liste officielle).



Le chooser livré avec GDM. Personnalisable à souhait et simple à configurer

Il faut donc utiliser le chooser de XDM qui est loin d'être un modèle d'esthétisme (mais on peut toujours améliorer la chose en jouant avec les ressources).
Par défaut, la configuration ne précise pas l'emplacement et le nom du binaire correspondant. Vous devrez donc, sans doute, le préciser vous-même dans votre ~~wdm-config~~ :

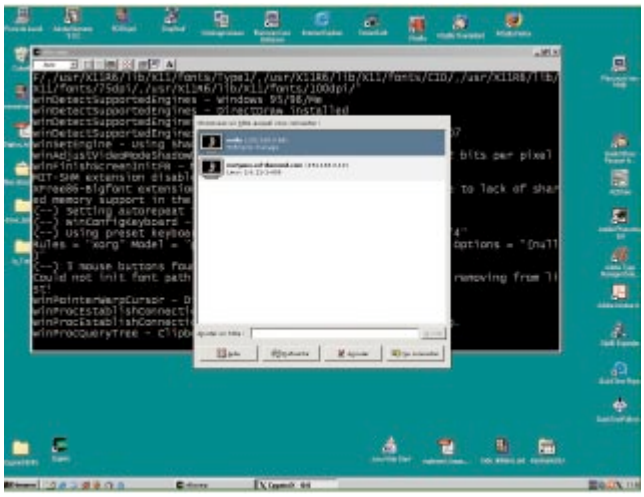
```
DisplayManager*chooser: /usr/X11R6/bin/chooser
```

En l'absence de cette ligne et si WDM ne trouve pas le chooser, vous risquez d'avoir un certain nombre de problèmes (tentative de connexion en boucle en l'absence de l'option -once sur le

client).

GDM

GDM est le GNOME Display Manager. Il est moderne, configurable à souhait, d'aspect moins rébarbatif que XDM, possède des possibilités de configuration plus poussées et il offre la possibilité d'utiliser des thèmes.
En dehors de tous ces aspects positifs, il faut également prendre en considération le fait que GDM utilise une syntaxe de configuration très différente et qu'il possède de fortes dépendances vis-à-vis de GNOME (ce qui est normal me direz-vous).
La configuration de GDM peut se faire via son fichier de configuration ~~/etc/gdm/gdm.conf~~ (il existe un lien symbolique ~~/etc/X11/gdm~~), mais égale-ment via un applicatif graphique ~~gdmsetup~~.
Il faut avouer que la syntaxe du ~~gdm.conf~~ étant très différente de ce que l'on trouve habituellement, mieux vaudra utiliser, dans un premier temps, l'interface graphique.
On pourra par la suite se pencher sur le fichier de configuration.
Il faut faire la distinction claire avec XDM/WDM. GDM s'occupe de tout via des binaires qui lui sont propres mais qui peuvent être changés.
Ainsi, le chooser est ~~gdmchooser~~ et la fenêtre d'authentification se décline en deux versions. La première est classique et sous forme de fenêtre (~~gdmlogin~~) et la seconde est la forme graphique (~~gdmgreeter~~), idéale pour les démonstrations mais peu recommandée pour les connexions distantes.
En termes de configuration, l'interface graphique et les commentaires du fichier de configuration parlent d'eux-mêmes.
Sachez simplement que tout est centralisé, des thèmes aux icônes, en passant par les temps d'attente et les systèmes d'authentification.
Il est possible de personnaliser et moduler l'ensemble.
Vous risquez de passer un bout de temps avec le nez dans le fichier de configuration mais cela vaut vraiment le coût pour obtenir quelque chose de personnel et tape à l'œil tout en restant utilisable et optimisé.



La fenêtre du chooser GDM dans un Windows avec un serveur X en mode rootless

Cas pratique et connexion cliente

Nous savons à présent comment configurer un Display Manager de manière à ce qu'il accepte les connexions externes, qu'il pro-pose une liste de serveurs qui lui sont semblables, et qu'il puisse authentifier un utilisateur et lui ouvrir une session.
Nous sommes donc prêts pour la partie cliente.
Le client ou terminal X ne doit savoir faire qu'une seule chose : démarrer un système (quel qu'il soit) et un serveur X fonctionnel et configuré.
Dans le cadre de ce numéro hors série, la machine en question sera une configuration relativement légère. Il n'en reste pas moins cependant qu'elle devra répondre à un certain nombre d'impératifs :

- Un système GNU/Linux doit pouvoir y démarrer et fonctionner normalement.
  Ceci implique la présence de mémoire vive (64 Mo) et d'un disque dur d'une taille raisonnable (300 Mo) ;
- Elle doit disposer d'un adaptateur graphique capable d'afficher une résolution et un nombre de couleurs rendant l'ensemble utilisable (800x600 ou 1024x768 en 65k couleurs). Inutile de préciser que le moniteur doit « suivre » ;
- Divers périphériques propres à un terminal X (clavier, souris, adaptateur réseau).

Il ne s'agit donc pas d'une machine totalement en fin de vie de type 486/16 Mo avec adaptateur EGA/VGA ISA ou VLB. La puissance CPU et la mémoire vive ne sont pas un élément clef.
En effet, le processeur n'aura à sa charge qu'un serveur X fonctionnant au-dessus d'un système réduit au minimum. Même constatation côté mémoire : les applications qui seront utilisées occuperont des ressources sur le serveur *DM où sera ouvert la session, et non sur le terminal lui-même qui se bornera presque à un travail d'affichage.
Installez la machine avec une distribution capable d'occuper le moins d'espace disque et le moins de ressources possibles. J'aurais tendance à vous conseiller naturellement une distribution Debian mais d'autres feront tout aussi bien l'affaire.
N'hésitez pas, éventuellement, à ressortir des tiroirs de vieilles distributions (type RedHat 6.2).
En effet, le terminal étant composé, normalement, de vieux éléments, vous ne devriez pas rencontrer de problèmes de compatibilité.

Des fenêtres WindowMaker avec des applications X dans un Windows 98. De quoi surprendre le pauvre l'utilisateur Windows qui passe :)

Configurez la distribution de manière à obtenir un démarrage en mode console. Nous lancerons, dans un premier temps, le serveur X manuellement.
Vous pourrez toujours, par la suite automatiser ce lancement dans le démarrage du système.

Assurez-vous que le serveur X est correctement configuré puis tentez votre première connexion :

```
$ X -query morgane
```

Nous lançons directement le serveur X sans passer par le classique ~~startx~~. Nous n'en avons pas besoin. L'option ~~-query~~ suivie par le nom d'hôte cible permet d'ouvrir une connexion directe vers un serveur *DM.

Si tout est correctement configuré, vous devriez voir apparaître exactement la même chose que lorsque vous vous authentifiez localement sur le serveur.

En cas de problème avec un serveur XDM/WDM, voici quelques points de départ pour une recherche :

- Le serveur XDM/WDM est-il lancé (c'est idiot mais en phase de test cela arrive) ?
- Le serveur accepte-t-il les connexions (ligne ~~DisplayManager.requestPort~~ du ~~wdm-config~~) ?
- Autorisez-vous les connexions depuis ce client (fichier ~~Xaccess~~) ?

Afin de vous éviter d'utiliser la combinaison de touches [CTRL]+[ALT]+[BACKSPACE] pour tuer votre serveur X, vous pouvez ajouter l'option -once. Celle-ci permet de limiter la connexion à une seule et unique session.

Si vous avez décidé d'autoriser les connexions indirectes, vous pouvez utiliser l'option ~~-indirect~~ en lieu et place de ~~-query~~. Cette option prend en argument un nom de serveur qui répondra en fournissant la liste des serveurs *DM.

Ici, l'option ~~-once~~ est plus que conseillée pour le premier lancement puisqu'en cas de défaillance du chooser le serveur X du terminal risque de «boucler».
Je pense par exemple à WDM qui, en l'absence de chooser lui étant propre, semble poser problème dans sa configuration par défaut.
Il faudra dans ce cas s'assurer du chemin spécifié via l'option ~~DisplayManager*chooser:~~ du fichier ~~wdm-config~~.
Normalement, le dialogue entre le serveur et le client permet de mettre en place toutes les autorisations nécessaires (voir début d'article).
Cependant, si vous n'arrivez pas à résoudre un problème de connexion, vous pouvez toujours essayer de lancer le serveur X avec l'option -ac permettant de désactiver le contrôle d'accès.
Si tout est en ordre, il ne vous reste plus qu'à tenter d'alléger encore davantage la distribution sur le terminal X.
Au risque de ne plus pouvoir maintenir à jour la distribution, vous pouvez tenter d'optimiser au maximum. Plus vous réduirez le temps de démarrage, plus vous vous rapprocherez d'un véritable terminal.

Inversion des rôles

Nous avons pour l'instant toujours envisagé l'utilisation d'une machine aux ressources graphiques viables mais disposant d'une configuration CPU/Mémoire/Disque trop légère. La situation inverse peut également se présenter.
En effet, assembler une machine autour d'un Celeron/PIII/Athlon avec 512 Mo ne reviendra pas très cher pour peu que l'on choisisse un disque de moins de 40 Go et une carte mère d'entrée de

gamme.
Des unités centrales de ce type se trouvent, neuves, pour un peu moins de 180 euros sur n'importe quelle boutique en ligne, le tout avec un boîtier qui ne fera pas «tache» dans le bureau :)
Ici, le point sensible n'est plus la configuration mais les périphériques, et en particulier l'écran.
Pourquoi donc ne pas faire de cette machine un poste de travail, de test, de développement... en utilisant le serveur X à votre disposition sur votre machine principale ?
Certes, vous pouvez déjà le faire de manière ponctuelle en exécutant les applications graphiques à distance, mais l'utilisation d'un Display Manager vous permettra d'ouvrir une session complète et ce, dans un environnement particulier.
Le principe de configuration reste, bien sûr, le même, à la simple différence que le serveur *DM est installé sur la machine secondaire (ex-terminal X) et que le serveur X utilisé sera celui de la machine principale. Comme précédemment, vous pouvez démarrer en mode console et lancer le serveur manuellement :

```
$ X -query molly :1
```

On utilisera toutefois un paramètre supplémentaire :1 en fin de ligne pour préciser le display supplémentaire à créer. Par défaut, celui-ci est 0.
Or, si un serveur X est déjà en route sur la machine, ce display est déjà occupé.
Lancer un second serveur n'est pas toujours une bonne idée. Cela consomme des ressources et le fait de basculer de la machine locale à celle distante devient très vite lassant. Pourquoi ne pas lancer un serveur X dans le serveur X déjà en cours d'utilisation ? Je parle, bien sûr, de Xnest. Ce serveur «niché» prendra en argument quasiment les mêmes options que le vrai serveur X :

```
$ Xnest -geometry 1024x768 -query molly -kb :1
```

L'option ~~-kb~~ permet de désactiver l'extension clavier et de conserver la configuration actuelle.
Nous connaissons déjà les autres options et remarquons, là aussi, la présence du :1 afin de ne pas entrer en conflit avec le serveur X déjà présent.
Et voilà, nous obtenons, dans une fenêtre, une ouverture de session *DM directement sur la machine secondaire. De quoi jouer avec un environnement alors qu'on en utilise présentement un autre.
Ou encore, un bon moyen de lancer des calculs de Ray-tracing sur une machine distante en gardant un œil sur le résultat.
Bien entendu, ce système possède des limitations : lancer la lecture d'une vidéo sur la machine distante n'est pas une idée de génie, tout comme le fait de tenter d'utiliser des applications OpenGL.
On notera au passage que n'importe quel système capable de disposer d'un serveur X peut entrer dans la danse. Il peut s'agir de systèmes UNIX comme Solaris ou *BSD, mais également de MS Windows.
En effet, l'installation de Cygwin/X permet à ce système de disposer de nombreuses fonctionnalités UNIX parmi lesquelles le fait de lancer un serveur X.
Il est ainsi possible d'ouvrir une session *DM sur une machine GNU/Linux, dans une fenêtre, en plein écran ou... en mode rootless. Ce dernier cas attisera à coup sûr la curiosité des utilisateurs Windows passant dans le coin.
Des applications X comme The Gimp, Xbill, GQview dans des fenêtres Windows... quelle étrangeté :)
Une fois l'environnement Cygwin/X installé, le lancement est tout aussi simple que pour GNU/Linux :

```
XWin.exe -query 192.168.0.10 -rootless
```

Sur plateforme Mac OS X, celle-ci intégrant un serveur X, la meilleure solution consiste à utiliser Xnest tout comme on le ferait sous GNU/Linux.
On obtient ainsi une fenêtre parfaitement intégrée au reste du système.
Il doit également être possible de lancer le serveur X natif en utilisant l'option -query ou -indirect mais ceci n'a pas été expérimenté dans le cadre de cet article.

Conclusion

Nous venons de voir qu'un grand nombre de combinaisons client/serveur sont utilisables à peu de frais.
Faire ainsi revivre une machine à l'abandon offre des perspectives intéressantes que je vous laisse deviner.
Bien au-delà des possibilités qu'offrent des solutions comme VNC, les fonctionnalités incluses de base dans tout système UNIX/X sont celles qui permettent la plus grande souplesse de personnalisation et de configuration.
A vous maintenant de tirer parti au mieux de tout cela.

~~Retrouvez cet article dans : Linux Magazine Hors série 21~~

Posté par Denis Bodor (Lefinnois) | Signature : Denis Bodor | Article paru dans 

Il y a actuellement un commentaire dans "Créez un terminal X"

1. 1 Le 6 août 2008, *ib@w[10]* ecrivait:

   Bonjour,

   J'ai un souci concernant l'utilisation de la commande Xnest.

   Je voudrais utiliser dans la fenêtre déportée un clavier azerty et le pavé numérique
   - L'option +kb me permet d'avoir le pavé numérique directement mais le clavier est qwerty.
   - L'option -kb donne un clavier azerty mais pour utiliser le pavé numérique (à droite), il faut appuyer sur la touche "shift" (ce que je voudrais éviter) et en plus certaines touches ("4″ et "7″) ne fonctionne pas correctement dans les formulaires ("4″ remplace le caractère précédent par "4″ et "7″ remplace toute la ligne par "7″…)

   Sauriez-vous me dire comment avoir le résultat attendu ?
   Merci,

Laissez une réponse

Vous devez avoir ouvert une session pour écrire un commentaire.

« Précédent Aller au contenu »

Identifiez-vous
Inscription
S'abonner à UNIX Garden

Articles de 1ère page

- Pear et les librairies PHP
- FreeDOS
- Mise en œuvre d'une passerelle Internet sous Linux
- Les pseudo-classes en CSS
- GNU/Linux Magazine N°108 - Septembre 2008 - Chez votre marchand de journaux
- Linux Pratique N°49 - Septembre/Octobre 2008 - Chez votre marchand de journaux
- Donner du style à son texte : utiliser les lettrines
- Installer un serveur Syslog
- Quatre serveurs FTP hyper sécurisés avec vsftpd
- Le point sur... Les formats d'images sur le web

Il y a actuellement

- 718 articles/billets en ligne.

Recherche

Catégories

- Administration réseau
- Administration système
- Agenda-Interview
- Audio-vidéo
- Bureautique
- Comprendre
- Distribution
- Embarqué
- Environnement de bureau
- Graphisme
- Jeux
- Matériel
- News
- Programmation
- Réfléchir
- Sécurité
- Utilitaires
- Web

Archives

- août 2008
- juillet 2008
- juin 2008
- mai 2008
- avril 2008
- mars 2008
- février 2008
- janvier 2008
- décembre 2007
- novembre 2007
- février 2007

GNU/Linux Magazine

- Ce billet est hors ligne montrer/masquer GNU/Linux Magazine 108 - Septembre 2008 - Chez votre marchand de journaux
- Edito : GNU/Linux Magazine 108
- GNU/Linux Magazine HS 38 - Septembre/Octobre 2008 - Chez votre marchand de journaux
- Edito : GNU/Linux Magazine HS 38
- GNU/Linux Magazine 107 - Juillet/Août 2008 - Chez votre marchand de journaux

- ## GNU/Linux Pratique

- - Linux Pratique N°49 -Septembre/Octobre 2008 - Chez votre marchand de journaux
  - Edito : Linux Pratique N°49
  - À télécharger : Les fichiers du Cahier Web de Linux Pratique n°49
  - Linux Pratique Essentiel N°3 - Août/Septembre 2008 - Chez votre marchand de journaux
  - Edito : Linux Pratique Essentiel N°3

- ## MISC Magazine

- - Misc 38 : Codes Malicieux, quoi de neuf ? - Juillet/Août 2008 - Chez votre marchand de journaux
  - Edito : Misc 38
  - Références de l'article « Détection de malware par analyse système » d'Arnaud Pilon paru dans MISC 38
  - Références de l'article « La sécurité des communications vocales (3) : techniques numériques » d'Éric Filiol paru dans MISC 38
  - Misc 37 : Déni de service - Mai/Juin 2008 - Chez votre marchand de journaux