UNIVERSITÉ PARIS-EST Marne-la-Vallée

Unix Système

Licence 2

2012-2013

Sylvain Cherrier

cherrier@univ-mlv.fr

Les users

- Unix multitaches et multiutilisateurs
- Problèmes : disponibilité, identification, confidentialité, imputabilité
- Qui peut faire quoi ?
- Accès à des données
- Accès à des programmes
- Accès à des périphériques

Users

- Authentification : Ce que l'on sait, ce que l'on est, ce que l'on a
- Identification : connaissance de l'utilisateur
- Pour ensuite appliquer des droits : il faut une base de connaissances des objets manipulés et des droits possibles
- Croiser ces informations permet d'obtenir le droit applicable

Cloisonnement du système

Limitations et contrôle

- Chacun son espace mémoire
- Chacun sa priorité d'accès
- Chacun a droit à un certain temps d'utilisation
- Selon le type de système, ce partage est garanti ou « best effort »

Même les actions...

…. sont sécurisées :

- Éteindre la machine
- Changer l'heure
- Changer les réglages réseau
- Installer ou des-installer des programmes

Pourquoi ?

Processus et utilisateurs

- Su
  - Permet de devenir n'importe quel utilisateur dont on connait le mot de passe
- Sudo
  - Certains utilisateurs (dont la liste est controlée par l'admin) peuvent devenir admin (en utilisant leur propre mot de passe)
- SUID
  - Le programme obtient les droits du propriétaire du binaire

Processus et utilisateurs

- Su
  - Permet de devenir n'importe quel utilisateur dont on connait le mot de passe
- Sudo
  - Certains utilisateurs (dont la liste est controlée par l'admin) peuvent devenir admin (en utilisant leur propre mot de passe)
- SUID
  - Le programme obtient les droits du propriétaire du binaire

File System et utilisateurs

- Chaque objet est identifié et des droits sont décrits
- Un objet a un propriétaire et un groupe
- Un utilisateur appartient à plusieurs groupes
- Les droits sont donnés pour le propriétaire, un groupe, et enfin pour les autres
- Il est possible de changer les droits, le groupe et le propriétaire de chaque objet (action réservée au root, et un peu au propriétaire)

Gestion des objets du FileSystem

- Pour un système V : les réglages d'un objet dépendent à la création du propriétaire : il en est le propriétaire, et il appartient à son groupe principal (cohérence utilisateur)
- Pour un BSD, le propriétaire est le créateur là aussi, mais le groupe est celui du répertoire conteneur du fichier (cohérence File System)
- Mais toujours la même démarche : proposer une base sécurisée

De nombreux outils

- umask
  - Permet de définir les droits par défaut (755 ou 644 peuvent être trop permissifs pour vous)
- chmod,chown,chgrp
  - Pour gérer les droits
- ACLs
  - Pour étendre le système, on peut y ajouter les ACLs (pas automatiquement utiles, car tout devient complexes)
  - Getfacls, setfacls permettent ensuite la gestion

Les processus

- Chaque processus est identifié (PID = numéro unique à un moment T)
- Chaque processus correspond à une espace mémoire réservé et borné (d'où l'appel à malloc et à free)
- L'espace est sous contrôle du processeur
- En cas de sortie de l'espace : Seg Fault !
- Lié à l'utilisateur pour les droits d'accès

Commandes processus

- kill
  - Permet d'envoyer un signal à un processus présent en mémoire. Communication rudimentaire
- trap
  - « Attrape » les signaux, afin de les gérer. Mais le SIGKILL et SIGTERM ne peuvent pas être détournés
- ps
  - Liste les processus présents en mémoire... Beaucoup d'options !!! permet de récupérer les pids

Commandes processus

- top
  - Affiche dynamiquement et rafraîchit une liste de processus (avec critéres de tri : consommation CPU, taille mémoire, etc)
- nice et renice
  - Rendre un processus « gentil »
- jobs, &, fg, et bg
  - Liste des processus en arrière plan.

L'ordonnanceur

- Principe de files d'attente, avec tourniquet
- Plusieurs files d'attentes, avec priorité différentes
- De nombreux algorithmes existent afin d'optimiser la fluidité de l'ordonnancement
- L'ordonnanceur interrompt régulièrement les processus (préempte) afin d'en faire passer un autre

Ordonnanceur

- Passage de relais : On parle de « restauration de contexte » d'un processus
- Il suffit de remettre le processeur dans l'état exacte dans lequel il était lorsqu'on a interrompu le processus
- La fréquence de preemption est liée à la durée nécessaire à la « restauration de contexte » (trouver le bon rapport fluidité du multitache/pénalisation du changement de contexte)

Processus et library (bibliothéque)

- Un programme peut être soit autonome, soit dépendant de bibliothèques (DLL ou so)
- L'intéret de l'autonomie est pénalisé par le poids du binaire (la même fonction est utilisée donc recopiée des milliers de fois)
- Les bibiothéques dynamiques allégent la taille des binaires, mais les rendent dépendants de la présence et de la validité de cette Library en mémoire.

La libc et autres libs

- Les programmes C dépendent souvent de la libc fournie avec votre système
- Selon ses besoins, votre application dépend d'autres bibliothèques
  - **#include<...>** prévient le compilateur C que vous allez utiliser des fonctions que vous n'avez pas écrites vous même, et dont le format d'utilisation est décrit dans l'include (il ne s'agit pas vraiment de la libraire, mais de son « mode d'emploi »)
- Ces bibliothèques doivent être installées, mise à jour, sécurisées
- Attention aux versions, aux altérations et aux dépendances.

Outils liés aux bibliothèques.

- ldd
  - Permet d'interroger un binaire afin de connaître ses dépendances aux bibliothèques. L'ensemble des bibliothèques indispensables au fonctionnement du programme est listé
- ldconfig
  - Programme qui scanne et met à jour les liens avec les bibliothèques, interroge la liste des présents
- gcc
  - Pour créer des so : -PIC pour obtenir un objet « relogeable » (position independant code) puis -shared pour avoir le so

Accès au système

- Possible en C (ou d'autres langages) via des bibliothèques
- Par exemple :

```
#include<unistd.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/types.h>
```

Se préparer à programmer système

- Il faut installer les codes sources des bibliothèques
- Les outils de gestion des paquets font cela très bien
- Les pages de man sont installés (man 2, ou man 3)
- Consultez les docs, et faites les includes

Gestion des paquets

- La plupart des Unix vous propose la gestion automatique des paquets, et la mise à jour du système
- Debian (et dérivé) utilise le format deb
- Dpkg permet d'installer un fichier .deb
- Aptitude, apt-get offre une base de données de gestion des dépendances entre paquets : via cet outil, les installations sont fonctionnelles

aptitude

Permet la gestion des paquets

- **aptitude search nom** vous indiquera si le nom existe, la version, et si c'est installé, installable...
- **aptitude install nom** installe l'outil ET ses dépendances
- **aptitude remove nom** supprime l'outil indiqué
- **aptitude update** et **upgrade** synchronise, et mettent à jour votre installation compléte (voire même change de version)