Systèmes d'exploitation

Introduction

Thomas Ropars

Email: thomas.ropars@univ-grenoble-alpes.fr

Website: tropars.github.io

Organisation du cours

- 5 séances de cours (1h30)
- 5 séances de TD (1h30)
- 5 séances de TP (1h30)

Évaluation

- Note de contrôle continu (50%): 1 interro + 1 TP noté
- Examen terminal (50%)

Intervenants

- Thomas Ropars (thomas.ropars@univ-grenoble-alpes.fr)
- Louis Boulanger (louis.boulanger@univ-grenoble-alpes.fr)
- Nicolas Homberg (nicolas.homberg@univ-grenoble-alpes.fr)

Contenu du cours

Étudier les systèmes d'exploitation d'un point de vue utilisateur

- Introduction aux systèmes d'exploitation
- Les processus
- Communication entre processus
- Les fichiers
- La sécurité

Références

Le contenu de ce cours s'inspire:

- Des notes de cours de V. Danjean
- Des cours de V. Marangozova et R. Lachaize

Principales références

- *Computer Systems: A Programmer's Perspective* by R. Bryant and D. O'Hallaron (chap 1, 8 et 10)
- *Operating Systems: Three Easy Pieces* by R. Arpaci-Dusseau and A. Arpaci-Dusseau (available online)
- *Operating System Concepts* by A. Silberschatz, P. B. Galvin, G. Gagne

Dans ce cours

- Rôle d'un système d'exploitation
- UNIX, Posix et Linux
- L'interface d'un système d'exploitation
- Organisation d'un système d'exploitation (Vu en détails dans un cours suivant)

Introduction

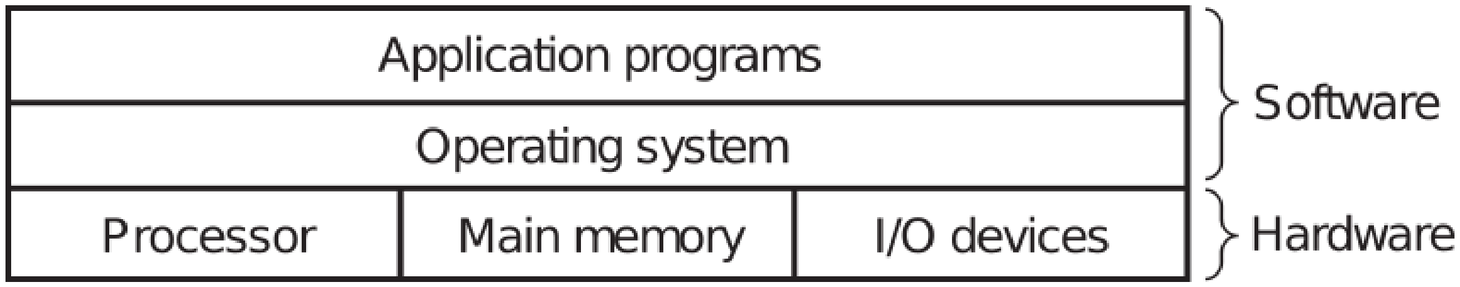
Système d'exploitation -- Les motivations

- Opérations effectuées par un programme (une *application*)
    - Exécuter des instructions
    - Lire et écrire dans la mémoire
    - Lire et écrire des fichiers
    - Accéder aux périphériques

```
Le système d'exploitation doit permettre:
- de *simplifier l'écriture et l'exécution de programmes*
- de *gérer les ressources*
```

Rôle d'un système d'exploitation

Le système d'exploitation est la couche logicielle qui fait le lien entre le matériel et les programmes des utilisateurs



Les 2 rôles principaux d'un système d'exploitation sont:

- de virtualiser les ressources matérielles
- de gérer l'accès aux ressources

Crédits: figure by R. Bryant and D. O'Hallaron

Virtualisation

Transformer les ressources physique en ressources virtuelles

Virtualisation

Transformer les ressources physique en ressources virtuelles

- Cache les interfaces de bas niveau du matériel pour fournir des abstractions de plus haut niveau
    - Facilité d'utilisation
    - Généricité / Portabilité
        - L'interface de haut niveau reste la même avec des matériels différents
    - Mise en oeuvre d'opérations complexes
    - Sécurité:
        - Protecte contre une mauvaise utilisation du matériel par un programme utilisateur
- Masque les limites du matériel (nombre de processeurs, taille de la mémoire, etc.)
- Masque le partage des ressources entre utilisateurs et/ou programmes

Gestion de l'accès aux ressources

L'OS permet à plusieurs programmes de s'exécuter en même temps sur une machine.

Ces programmes peuvent accéder aux ressources matérielles (processeur, mémoire, disques, etc.) en même temps. L'OS doit alors assurer:

- l'**équité** dans l'accès aux ressources
- L'**efficacité** de l'accès aux ressources
- La **sécurité** de chaque application

Ceci implique:

- de gérer l'**allocation**, le **partage** et la **protection** des ressources
- de mettre en place un ensemble de **mécanismes** et de **politiques**

Les familles d'OS

Un peu d'histoire (le début)

Attention: Ce qui est présenté ici est un résumé très partiel

Les premiers OS

- Un ensemble de bibliothèques de fonctions très courantes
    - Par exemple pour accéder aux périphériques de stockage
- Un seul programme s'exécute à la fois
    - C'est un opérateur humain qui décide quel programme exécuter
- Pas de mécanismes de protection
    - On suppose qu'il n'y a pas de *mauvais* programmes

Limitations

Un peu d'histoire (le début)

Attention: Ce qui est présenté ici est un résumé très partiel

Les premiers OS

- Un ensemble de bibliothèques de fonctions très courantes
    - Par exemple pour accéder aux périphériques de stockage
- Un seul programme s'exécute à la fois
    - C'est un opérateur humain qui décide quel programme exécuter
- Pas de mécanismes de protection
    - On suppose qu'il n'y a pas de *mauvais* programmes

Limitations

- Problèmes de sécurité
    - Que ce passe-t-il si une application à accès à toutes les données présentes sur le disque?
- Problèmes d'efficacité
    - Mauvaise utilisation du matériel
        - Le processeur ne fait rien quand le programme accède au disque

Les principes de base des OS modernes

Protection

- Le code exécuté par le système d'exploitation joue un rôle central
    - Il doit être traiter différemment du code applicatif
- Les programmes applicatifs ne doivent pas accéder directement au matériel
    - Force les applications à déléguer les opérations critiques au système d'exploitation (notion d'appel système)
        - Exécution en *mode noyau* contrôlée par le matériel

Les principes de base des OS modernes

Multi-tâches

- Idée: Améliorer l'utilisation des ressources en exécutant plusieurs programmes de manière concurrente
    - Si un programme est bloqué (attente d'entrées/sorties), on en exécute un autre
- **Problèmes**
    - Que se passe-t-il si un programme monopolise le processeur?
    - Que se passe-t-il si un programme accède aux données en mémoire d'un autre programme?
- Des mécanismes de protection sont nécessaires

Un peu d'histoire (suite)

UNIX

- Créé par Ken Thompson de Bell Labs en 1969.
- Rendu disponible gratuitement pour les universités dans les années 70
- Écrit en C
- Fournit un large ensemble de commandes exécutables dans un interpréteur (le *shell*)
- De nombreux systèmes actuels basés sur les principes d'UNIX: Open BSD, Free BSD, Linux, Mac OS, Android, etc.

Idées principales

- Multi-tâches
- Design modulaire (un ensemble d'outils simples)
- Un système de fichier unifié
    - Tout est fichier
- Des processus coopérant
    - Il est simple de créer des processus et de les faire communiquer

Un peu d'histoire (la fin)

POSIX

- Dans les années 80, les vendeurs UNIX ont commencé à ajouter des fonctionnalités incompatibles
- POSIX = Portable Operating System Interface (+X for UNIX)
    - Spécification IEEE de l'API (Application Programming Interface) des systèmes d'exploitations
    - Sous-ensemble de fonctions supportées par tous les systèmes UNIX

Linux

- Implémentation d'un système UNIX from scratch
    - Par Linus Torvalds
- Publié en tant que *logiciel libre* en 1991
- S'est largement développé avec l'avènement du Web
    - Toutes les grandes entreprise du net l'utilisent
    - Domine le marché des serveurs et des smartphones (Android)

Interface d'un OS

Interface d'un OS

Typiquement 2 types d'interfaces

- Interface utilisateur
    - A la ligne de commande (Le shell)
    - Interface graphique
    - Composées d'un ensemble de commandes
    - Exemple: supprimer un fichier
        - A la ligne de commande: `rm file.txt`
        - Dans l'interface graphique: Drag and drop dans la corbeille
- Interface programmatique
    - API (Application Programming Interface)

Interface utilisateur

Interface graphique

Interface utilisateur

Interface à la ligne de commande

```
[root@localhost ~]# ping -q fa.wikipedia.org
PING text.pmtpa.wikimedia.org (208.80.152.2) 56(84) bytes of data.
^C
--- text.pmtpa.wikimedia.org ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 540.528/540.528/540.528/0.000 ms
[root@localhost ~]# pwd
/root
[root@localhost ~]# cd /var
[root@localhost var]# ls -la
total 72
drwxr-xr-x. 18 root root 4096 Jul 30 22:43 .
drwxr-xr-x. 23 root root 4096 Sep 14 20:42 ..
drwxr-xr-x.  2 root root 4096 May 14 00:15 account
drwxr-xr-x. 11 root root 4096 Jul 31 22:26 cache
drwxr-xr-x.  3 root root 4096 May 18 16:03 db
drwxr-xr-x.  3 root root 4096 May 18 16:03 empty
drwxr-xr-x.  2 root root 4096 May 18 16:03 games
drwxrwx--T.  2 root gdm  4096 Jun  2 18:39 gdm
drwxr-xr-x. 38 root root 4096 May 18 16:03 lib
drwxr-xr-x.  2 root root 4096 May 18 16:03 local
lrwxrwxrwx.  1 root root   11 May 14 00:12 lock -> ../run/lock
drwxr-xr-x. 14 root root 4096 Sep 14 20:42 log
lrwxrwxrwx.  1 root root   10 Jul 30 22:43 mail -> spool/mail
drwxr-xr-x.  2 root root 4096 May 18 16:03 nis
drwxr-xr-x.  2 root root 4096 May 18 16:03 opt
drwxr-xr-x.  2 root root 4096 May 18 16:03 preserve
drwxr-xr-x.  2 root root 4096 Jul  1 22:11 report
lrwxrwxrwx.  1 root root    6 May 14 00:12 run -> ../run
drwxr-xr-x. 14 root root 4096 May 18 16:03 spool
drwxrwxrwt.  4 root root 4096 Sep 12 23:50 tmp
drwxr-xr-x.  2 root root 4096 May 18 16:03 yp
[root@localhost var]# yum search wiki
Loaded plugins: langpacks, presto, refresh-packagekit, remove-with-leaves
rpmfusion-free-updates                                          | 2.7 kB     00:00
rpmfusion-free-updates/primary_db                               | 206 kB     00:04
rpmfusion-nonfree-updates                                       | 2.7 kB     00:00
updates/metalink                                                | 5.9 kB     00:00
updates                                                         | 4.7 kB     00:00
Updates/primary_db                    73% [=======================         ]  62 kB/s | 2.6 MB     00:15 ETA
```

Le shell

- Interface utilisateur pour accéder aux services de l'OS
    - Interpréteur de ligne de commande
    - Langage de script (permet d'exécuter un ensemble de commandes)

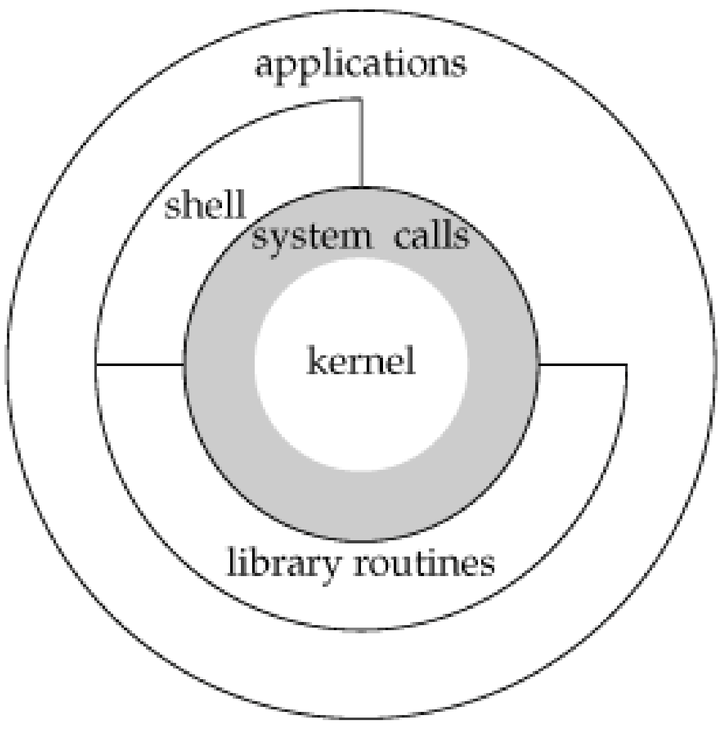
Bash

- Shell par défaut sur beaucoup de systèmes Linux
    - D'autres shell existent

Interface programmatique

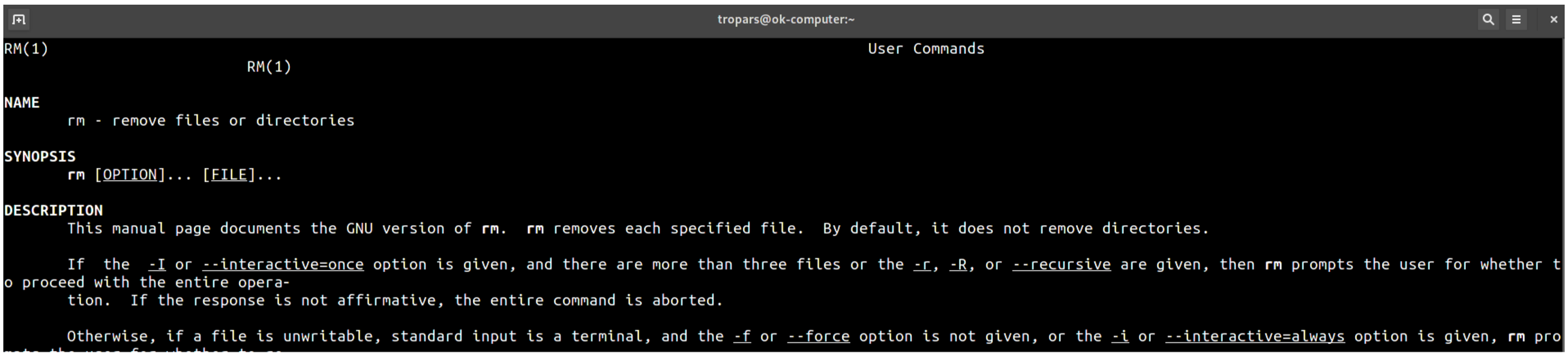
- Utilisée par les applications s'exécutant sur le système
    - Ceci inclut les programmes qui mettent en oeuvre l'interface graphique

- Composé d'un ensemble de:
    - Fonctions de bibliothèques
    - D'appels systèmes



Documentation

- Des pages de **man** (*man-pages*) fournissent de la documentation à l'utilisateur:
- Ces pages sont organisées en différentes sections:
    - `man 1 commande`: documentation des commandes accessibles via le shell
    - `man 2 fonction`: documentation des appels systèmes
    - `man 3 fonction`: documentation des fonctions de la bibliothèque C
    - Exemple: `man rm`



Résumé

- Rôles d'un système d'exploitation
    - Virtualisation
    - Gestion des ressources
- Objectifs
    - Efficacité -- Simplicité -- Portabilité -- Sécurité
- Évolution des systèmes d'exploitation
    - Des systèmes multi-tâches
    - Des mécanismes de protection
- Interface
    - Utilisateur
        - Ligne de commande (Shell)
    - Programmatique
        - Bibliothèques
        - Appels systèmes