Linux

Ivan Noyer

Lycée Thiers

Avant-propos

Ces transparents constituent une toute petite introduction forcément non exhaustive au système Linux. Ils sont présentés pour faciliter le quotidien des étudiants de MP2I dans l'élaboration de leurs programmes en cours d'année. Aucune des notions ou commandes présentées ici n'est exigible.

Crédits

- Un historique de Linux

Crédits

- Un historique de Linux
- Un cours sur Linux très complet : Linux France et un autre : Telecom Paris

Crédits

- Un historique de Linux
- Un cours sur Linux très complet : Linux France et un autre : Telecom Paris
- Des précisions sur l'architecture des fichiers sous Linux ici et la nomenclature des systèmes de fichiers là.

Crédits

- Un historique de Linux
- Un cours sur Linux très complet : Linux France et un autre : Telecom Paris
- Des précisions sur l'architecture des fichiers sous Linux ici et la nomenclature des systèmes de fichiers là.
- Un cours du mit

Historique

- Linux ou GNU/Linux : famille de systèmes d'exploitation open source de type Unix fondé sur le noyau Linux, créé en 1991 par **Linus Torvalds**.

Historique

- LINUX ou GNU/LINUX : famille de systèmes d'exploitation open source de type UNIX fondé sur le noyau LINUX, créé en 1991 par **Linus Torvalds**.
- GNU : projet informatique créé en janvier 1984 par **Richard Stallman** pour développer le système d'exploitation GNU. Chaque brique du projet est un logiciel libre utilisable en tant que tel mais dont l'objectif est de s'inscrire dans une logique cohérente. GNU/LINUX c'est donc le noyau LINUX plus les composantes de GNU.

Historique

- LINUX ou GNU/LINUX : famille de systèmes d'exploitation open source de type UNIX fondé sur le noyau LINUX, créé en 1991 par **Linus Torvalds**.
- GNU : projet informatique créé en janvier 1984 par **Richard Stallman** pour développer le système d'exploitation GNU. Chaque brique du projet est un logiciel libre utilisable en tant que tel mais dont l'objectif est de s'inscrire dans une logique cohérente. GNU/LINUX c'est donc le noyau LINUX plus les composantes de GNU.
- LINUX équipe une faible part des ordinateurs PC mais beaucoup de serveurs, téléphones portables, systèmes embarqués ou encore superordinateurs.

Historique

- Linux ou GNU/Linux : famille de systèmes d'exploitation open source de type Unix fondé sur le noyau Linux, créé en 1991 par **Linus Torvalds**.
- GNU : projet informatique créé en janvier 1984 par **Richard Stallman** pour développer le système d'exploitation GNU. Chaque brique du projet est un logiciel libre utilisable en tant que tel mais dont l'objectif est de s'inscrire dans une logique cohérente. GNU/Linux c'est donc le noyau Linux plus les composantes de GNU.
- Linux équipe une faible part des ordinateurs PC mais beaucoup de serveurs, téléphones portables, systèmes embarqués ou encore superordinateurs.
- Android : système d'exploitation pour téléphones portables. Utilise le noyau Linux mais pas GNU. Equipe 85 % des tablettes tactiles et smartphones.

Logiciel libre

- Distributions Linux (Ubuntu, Debian, Linux Mint, CentOS ...) : systèmes d'exploitation libres.

Logiciel libre

- Distributions Linux (Ubuntu, Debian, Linux Mint, CentOS ...) : systèmes d'exploitation libres.
- Les 4 libertés d'un logiciels libres telles que définies par la Free Software Foundation. On peut :

le non respect de ces règles peut conduire à des condamnations.

Logiciel libre

- Distributions Linux (Ubuntu, Debian, Linux Mint, CentOS ...) : systèmes d'exploitation libres.
- Les 4 libertés d'un logiciels libres telles que définies par la Free Software Foundation. On peut :
  - utiliser le logiciel sans restriction,

le non respect de ces règles peut conduire à des condamnations.

Logiciel libre

- Distributions Linux (Ubuntu, Debian, Linux Mint, CentOS ...) : systèmes d'exploitation libres.
- Les 4 libertés d'un logiciels libres telles que définies par la Free Software Foundation. On peut :
  - utiliser le logiciel sans restriction,
  - étudier le logiciel

le non respect de ces règles peut conduire à des condamnations.

Logiciel libre

- Distributions Linux (Ubuntu, Debian, Linux Mint, CentOS ...) : systèmes d'exploitation libres.
- Les 4 libertés d'un logiciels libres telles que définies par la Free Software Foundation. On peut :
  - utiliser le logiciel sans restriction,
  - étudier le logiciel
  - le modifier pour l'adapter à ses besoins

le non respect de ces règles peut conduire à des condamnations.

Logiciel libre

- Distributions Linux (Ubuntu, Debian, Linux Mint, CentOS ...) : systèmes d'exploitation libres.
- Les 4 libertés d'un logiciels libres telles que définies par la Free Software Foundation. On peut :
  - utiliser le logiciel sans restriction,
  - étudier le logiciel
  - le modifier pour l'adapter à ses besoins
  - le redistribuer sous certaines conditions précises

  le non respect de ces règles peut conduire à des condamnations.

Logiciel libre

- Distributions Linux (Ubuntu, Debian, Linux Mint, CentOS ...) : systèmes d'exploitation libres.
- Les 4 libertés d'un logiciels libres telles que définies par la Free Software Foundation. On peut :
  - utiliser le logiciel sans restriction,
  - étudier le logiciel
  - le modifier pour l'adapter à ses besoins
  - le redistribuer sous certaines conditions précises

  le non respect de ces règles peut conduire à des condamnations.
- Certaines licences sont conçues selon le principe du *copyleft* : une œuvre dérivée d'un logiciel sous copyleft doit à son tour être libre.

Logiciel libre

- Distributions Linux (Ubuntu, Debian, Linux Mint, CentOS ...) : systèmes d'exploitation libres.
- Les 4 libertés d'un logiciels libres telles que définies par la Free Software Foundation. On peut :
  - utiliser le logiciel sans restriction,
  - étudier le logiciel
  - le modifier pour l'adapter à ses besoins
  - le redistribuer sous certaines conditions précises

  le non respect de ces règles peut conduire à des condamnations.
- Certaines licences sont conçues selon le principe du *copyleft* : une œuvre dérivée d'un logiciel sous copyleft doit à son tour être libre.
- Licence GNU GPL : logiciel libre + copyleft. Le noyau Linux utilise cette licence.

Logiciel libre

- Distributions Linux (Ubuntu, Debian, Linux Mint, CentOS ...) : systèmes d'exploitation libres.
- Les 4 libertés d'un logiciels libres telles que définies par la Free Software Foundation. On peut :
  - utiliser le logiciel sans restriction,
  - étudier le logiciel
  - le modifier pour l'adapter à ses besoins
  - le redistribuer sous certaines conditions précises

  le non respect de ces règles peut conduire à des condamnations.
- Certaines licences sont conçues selon le principe du *copyleft* : une œuvre dérivée d'un logiciel sous copyleft doit à son tour être libre.
- Licence GNU GPL : logiciel libre + copyleft. Le noyau Linux utilise cette licence.
- Un logiciel libre n'est pas nécessairement gratuit, et inversement un logiciel gratuit n'est pas forcément libre.

Caractéristiques principales

- Linux est *multi-tâches* : plusieurs processus peuvent s'exécuter ≪ simultanément ≫.

Caractéristiques principales

- Linux est *multi-tâches* : plusieurs processus peuvent s'exécuter ≪ simultanément ≫.
- Linux est *multi-utilisateur* : Chaque personne utilisant le système dispose d'un *compte*, qui peut être vu comme une certaine zone qui lui est allouée, accessible par un nom et un mot de passe. Un mécanisme de droits un peu contraignant empêche un utilisateur d'accéder à des données dont il n'a pas les droits.

Caractéristiques principales

- Linux est *multi-tâches* : plusieurs processus peuvent s'exécuter ≪ simultanément ≫.
- Linux est *multi-utilisateur* : Chaque personne utilisant le système dispose d'un *compte*, qui peut être vu comme une certaine zone qui lui est allouée, accessible par un nom et un mot de passe. Un mécanisme de droits un peu contraignant empêche un utilisateur d'accéder à des données dont il n'a pas les droits.
- Les droits peuvent être modifiés.

Caractéristiques principales

- Linux est *multi-tâches* : plusieurs processus peuvent s'exécuter ≪ simultanément ≫.
- Linux est *multi-utilisateur* : Chaque personne utilisant le système dispose d'un *compte*, qui peut être vu comme une certaine zone qui lui est allouée, accessible par un nom et un mot de passe. Un mécanisme de droits un peu contraignant empêche un utilisateur d'accéder à des données dont il n'a pas les droits.
- Les droits peuvent être modifiés.
- Un utilisateur spécial a tous les droits le *super-utilisateur* ou *administrateur* (super user en anglais).

Noyau et processus

- *Noyau de système d'exploitation* (ou noyau, ou kernel en anglais) : partie fondamentale de certains systèmes d'exploitation. Gère les ressources de l'ordinateur et permet aux différents composants (matériels et logiciels) de communiquer entre eux.

Noyau et processus

- *Noyau de système d'exploitation* (ou noyau, ou kernel en anglais) : partie fondamentale de certains systèmes d'exploitation. Gère les ressources de l'ordinateur et permet aux différents composants (matériels et logiciels) de communiquer entre eux.
- Le noyau assure :

Noyau et processus

- *Noyau de système d'exploitation* (ou noyau, ou kernel en anglais) : partie fondamentale de certains systèmes d'exploitation. Gère les ressources de l'ordinateur et permet aux différents composants (matériels et logiciels) de communiquer entre eux.
- Le noyau assure :
  - la communication entre les logiciels et le matériel

Noyau et processus

- *Noyau de système d'exploitation* (ou noyau, ou kernel en anglais) : partie fondamentale de certains systèmes d'exploitation. Gère les ressources de l'ordinateur et permet aux différents composants (matériels et logiciels) de communiquer entre eux.
- Le noyau assure :
  - la communication entre les logiciels et le matériel
  - la gestion des divers logiciels (tâches) d'une machine (lancement des programmes, ordonnancement...)

Noyau et processus

- *Noyau de système d'exploitation* (ou noyau, ou kernel en anglais) : partie fondamentale de certains systèmes d'exploitation. Gère les ressources de l'ordinateur et permet aux différents composants (matériels et logiciels) de communiquer entre eux.
- Le noyau assure :
  - la communication entre les logiciels et le matériel
  - la gestion des divers logiciels (tâches) d'une machine (lancement des programmes, ordonnancement...)
  - la gestion du matériel (mémoire, processeur, périphérique, stockage...).

Noyau et processus

- *Noyau de système d'exploitation* (ou noyau, ou kernel en anglais) : partie fondamentale de certains systèmes d'exploitation. Gère les ressources de l'ordinateur et permet aux différents composants (matériels et logiciels) de communiquer entre eux.
- Le noyau assure :
  - la communication entre les logiciels et le matériel
  - la gestion des divers logiciels (tâches) d'une machine (lancement des programmes, ordonnancement...)
  - la gestion du matériel (mémoire, processeur, périphérique, stockage...).
- Le noyau offre ses fonctions (l'accès aux ressources qu'il gère) au travers des *appels système*. Il transmet ou interprète les informations du matériel via des interruptions (appellées les entrées/sorties)

Noyau et processus

- *Noyau de système d'exploitation* (ou noyau, ou kernel en anglais) : partie fondamentale de certains systèmes d'exploitation. Gère les ressources de l'ordinateur et permet aux différents composants (matériels et logiciels) de communiquer entre eux.
- Le noyau assure :
  - la communication entre les logiciels et le matériel
  - la gestion des divers logiciels (tâches) d'une machine (lancement des programmes, ordonnancement...)
  - la gestion du matériel (mémoire, processeur, périphérique, stockage...).
- Le noyau offre ses fonctions (l'accès aux ressources qu'il gère) au travers des *appels système*. Il transmet ou interprète les informations du matériel via des interruptions (appellées les entrées/sorties)
- *Processus* : programme en cours d'exécution par un ordinateur. Régulièrement, il a besoin d'accéder à des ressources protégées (comme une écriture en mémoire). Le noyau prend alors le relai du processus pour rendre le service demandé et lui rend le contrôle lorsque les actions voulues ont été réalisées.

Système de fichier

- système de fichiers d'un système d'exploitation : ensemble de principes et de règles selon lesquels les fichiers sont organisés et manipulés. Chaque système d'exploitation possède son système de fichier privilégié, même s'il peut en utiliser d'autres.

Système de fichier

- système de fichiers d'un système d'exploitation : ensemble de principes et de règles selon lesquels les fichiers sont organisés et manipulés. Chaque système d'exploitation possède son système de fichier privilégié, même s'il peut en utiliser d'autres.
-

OS	Système de fichier
MSDOS	FAT
Windows 95/98	FAT32 (File Allocation Table 32bits)
Windows NT	NTFS (New Technology File System) standard pour
MAC OS	HFS+ (High Performance File System)
Linux	ext4 (Extended File System)

Système de fichier

- Fichier : suite d'octets constituant un ensemble cohérent (en principe) d'information. Sous Linux, *tout* est fichier (même le clavier est considéré comme un fichier).

Système de fichier

- Fichier : suite d'octets constituant un ensemble cohérent (en principe) d'information. Sous Linux, *tout* est fichier (même le clavier est considéré comme un fichier).
- Règles de nommage Linux : 255 caractères au plus, principalement lettres et chiffres mais **. - _ ~ + %** sont aussi possibles. Les espaces ne sont pas interdits, mais je les déconseille de même que les accents.

Répertoires ou dossiers

- Si le système de fichier était une commode, les répertoires seraient des tiroirs contenant des fichiers ou des sous-tiroirs.

Répertoires ou dossiers

- Si le système de fichier était une commode, les répertoires seraient des tiroirs contenant des fichiers ou des sous-tiroirs.
- Un répertoires est un fichier dans lesquels se trouve la liste de son ≪ contenu ≫ (fichiers ou sous-répertoires) ainsi que des informations à leur propos (droits et propriétaires).

Répertoires ou dossiers

- Si le système de fichier était une commode, les répertoires seraient des tiroirs contenant des fichiers ou des sous-tiroirs.
- Un répertoires est un fichier dans lesquels se trouve la liste de son « contenu » (fichiers ou sous-répertoires) ainsi que des informations à leur propos (droits et propriétaires).
- Dans un répertoire on trouve deux fichiers particuliers notés « **.** » (répertoire courant) et « **..** » (répertoire parent).

Répertoires ou dossiers

- Si le système de fichier était une commode, les répertoires seraient des tiroirs contenant des fichiers ou des sous-tiroirs.
- Un répertoires est un fichier dans lesquels se trouve la liste de son « contenu » (fichiers ou sous-répertoires) ainsi que des informations à leur propos (droits et propriétaires).
- Dans un répertoire on trouve deux fichiers particuliers notés « **.** » (répertoire courant) et « **..** » (répertoire parent).
- Le système de fichier est représenté par un arbre dont la racine est le répertoire *root* noté « / ».

Répertoires ou dossiers

- Si le système de fichier était une commode, les répertoires seraient des tiroirs contenant des fichiers ou des sous-tiroirs.
- Un répertoires est un fichier dans lesquels se trouve la liste de son « contenu » (fichiers ou sous-répertoires) ainsi que des informations à leur propos (droits et propriétaires).
- Dans un répertoire on trouve deux fichiers particuliers notés « **.** » (répertoire courant) et « **..** » (répertoire parent).
- Le système de fichier est représenté par un arbre dont la racine est le répertoire *root* noté « / ».
- Bonne pratique pour les répertoires et fichiers persos : la première lettre des noms de répertoires en majuscules, celle des autres fichiers en minuscule : **UnNomDeRepertoire**, **unNomDeFichier**.

Exemple d'arborescence de fichiers

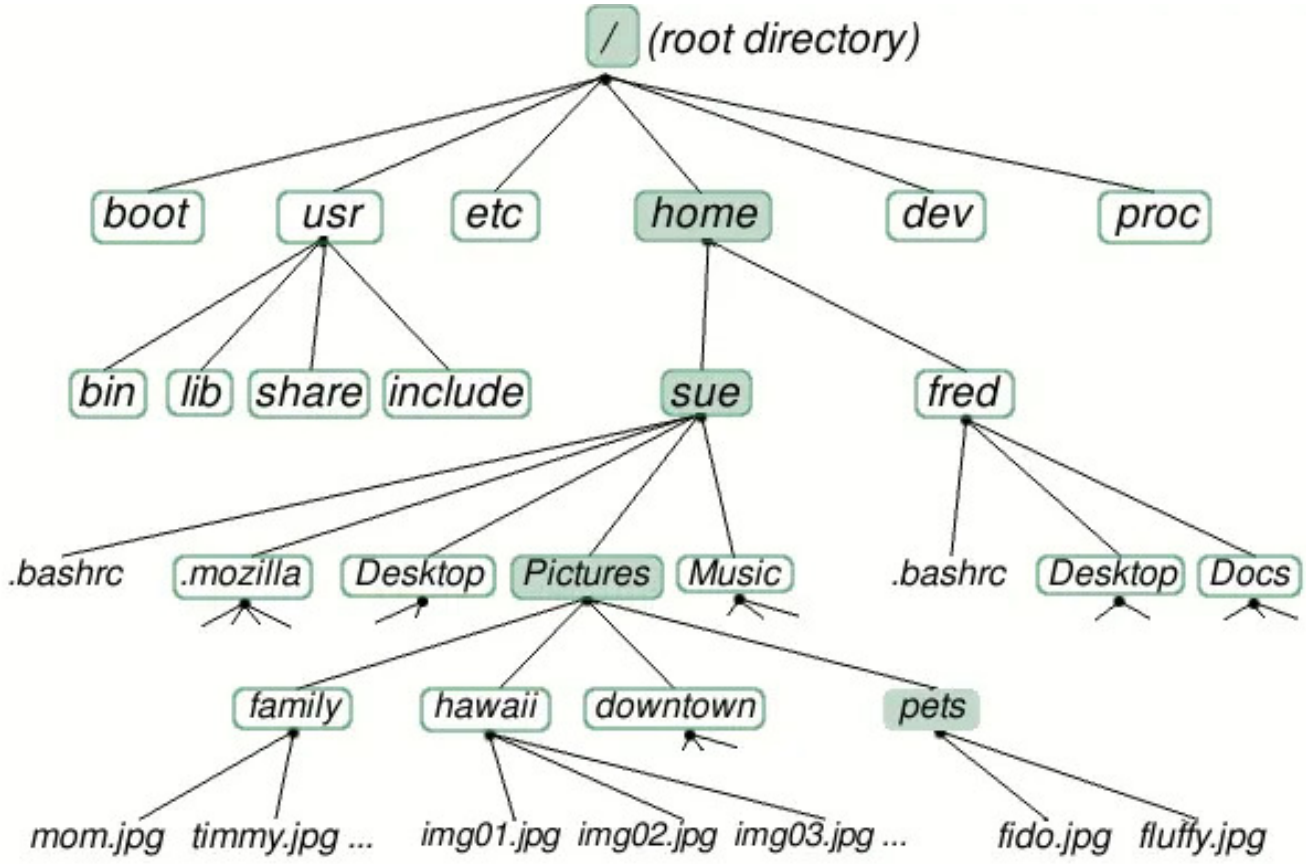


FIGURE – Le système de fichiers Linux (d'après malekal)

Aller à la page concernant les chemins

Exemple personnel

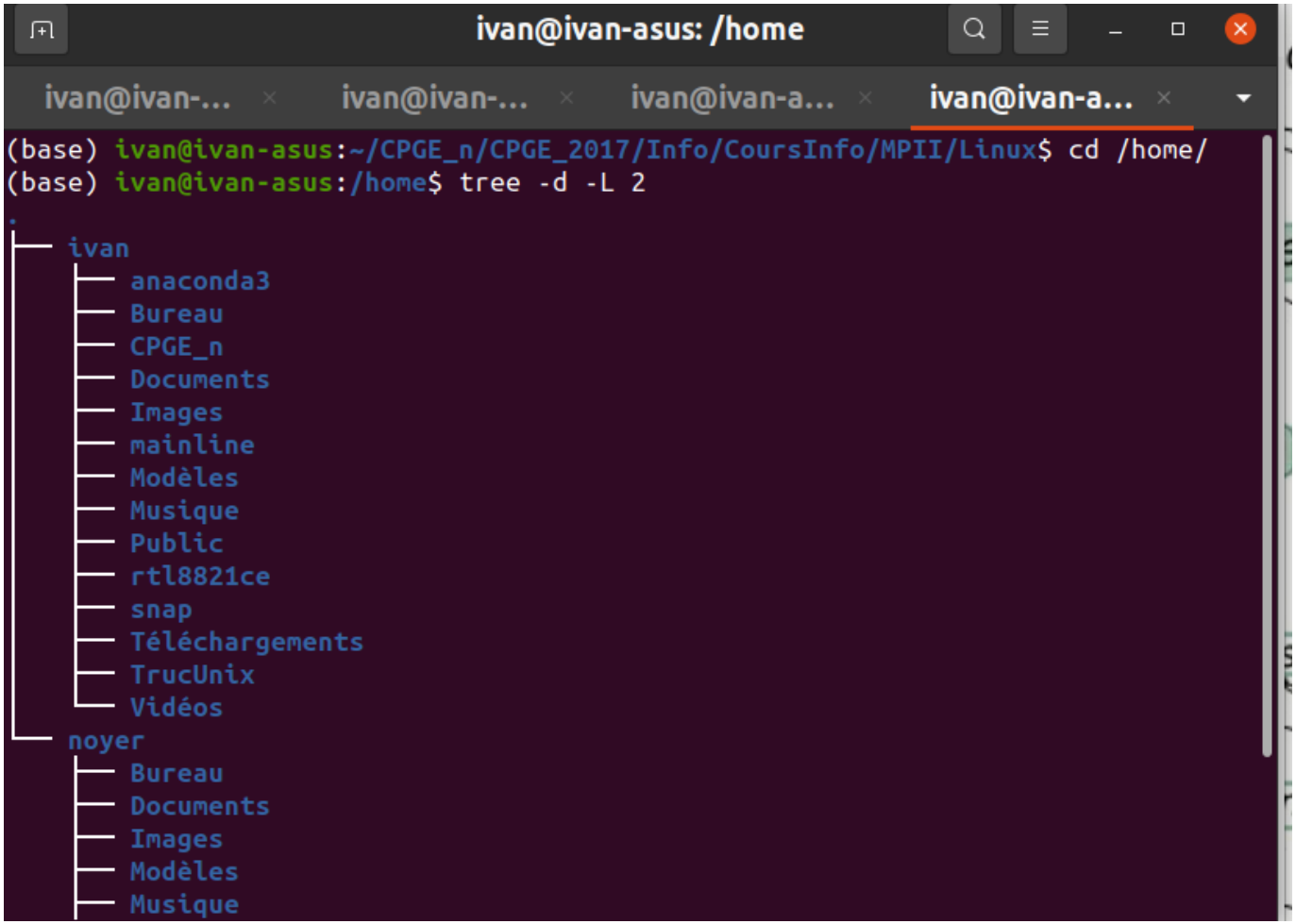


FIGURE – Une image de mon répertoire **home** donnée par la commande **tree**

Principaux répertoires

/ C'est la racine de l'arborescence

Principaux répertoires

/ C'est la racine de l'arborescence

/bin stocke les exécutables et binaires essentiels lors du démarrage. Commandes utilisables ensuite par les utilisateurs (ex : **cat** et **ls**).

Principaux répertoires

/ C'est la racine de l'arborescence

/bin stocke les exécutables et binaires essentiels lors du démarrage. Commandes utilisables ensuite par les utilisateurs (ex : **cat** et **ls**).

/sbin stocke les exécutables et binaires essentiels lors du démarrage mais réservées au *superuser* (administrateur système).

Principaux répertoires

/ C'est la racine de l'arborescence

/bin stocke les exécutables et binaires essentiels lors du démarrage. Commandes utilisables ensuite par les utilisateurs (ex : **cat** et **ls**).

/sbin stocke les exécutables et binaires essentiels lors du démarrage mais réservées au *superuser* (administrateur système).

/boot stocke les fichiers de démarrage Linux

Principaux répertoires

/ C'est la racine de l'arborescence

/bin stocke les exécutables et binaires essentiels lors du démarrage. Commandes utilisables ensuite par les utilisateurs (ex : **cat** et **ls**).

/sbin stocke les exécutables et binaires essentiels lors du démarrage mais réservées au *superuser* (administrateur système).

/boot stocke les fichiers de démarrage Linux

/dev (pour *devices*) les fichiers liés aux périphériques.

Principaux répertoires

- **/** C'est la racine de l'arborescence
- **/bin** stocke les exécutables et binaires essentiels lors du démarrage. Commandes utilisables ensuite par les utilisateurs (ex : **cat** et **ls**).
- **/sbin** stocke les exécutables et binaires essentiels lors du démarrage mais réservées au *superuser* (administrateur système).
- **/boot** stocke les fichiers de démarrage Linux
- **/dev** (pour *devices*) les fichiers liés aux périphériques.
- **/etc** Les fichiers de configuration de Linux et des applications.

Principaux répertoires

/	C'est la racine de l'arborescence
/bin	stocke les exécutables et binaires essentiels lors du démarrage. Commandes utilisables ensuite par les utilisateurs (ex : **cat** et **ls**).
/sbin	stocke les exécutables et binaires essentiels lors du démarrage mais réservées au *superuser* (administrateur système).
/boot	stocke les fichiers de démarrage Linux
/dev	(pour *devices*) les fichiers liés aux périphériques.
/etc	Les fichiers de configuration de Linux et des applications.
/home	comptes des utilisateurs

Principaux répertoires

/	C'est la racine de l'arborescence
/bin	stocke les exécutables et binaires essentiels lors du démarrage. Commandes utilisables ensuite par les utilisateurs (ex : **cat** et **ls**).
/sbin	stocke les exécutables et binaires essentiels lors du démarrage mais réservées au *superuser* (administrateur système).
/boot	stocke les fichiers de démarrage Linux
/dev	(pour *devices*) les fichiers liés aux périphériques.
/etc	Les fichiers de configuration de Linux et des applications.
/home	comptes des utilisateurs
/lib	Les librairies et bibliothèques partagés pour le fonctionnement de l'OS et des applications

Les principaux répertoires

/usr répertoire des applications partagées par différentes machines ou utilisateurs. Les programmes dans /**usr** ne sont pas nécessaires au démarrage.

Les principaux répertoires

/usr répertoire des applications partagées par différentes machines ou utilisateurs. Les programmes dans /**usr** ne sont pas nécessaires au démarrage.

/media points de montages des médias amovibles (clés USB...)

Les principaux répertoires

/usr répertoire des applications partagées par différentes machines ou utilisateurs. Les programmes dans /**usr** ne sont pas nécessaires au démarrage.

/media points de montages des médias amovibles (clés USB...)

/proc Répertoire virtuel avec les informations système (l'état du système, noyau Linux, etc) basé sur **procfs** (process file system)

Les principaux répertoires

/usr répertoire des applications partagées par différentes machines ou utilisateurs. Les programmes dans /**usr** ne sont pas nécessaires au démarrage.

/media points de montages des médias amovibles (clés USB...)

/proc Répertoire virtuel avec les informations système (l'état du système, noyau Linux, etc) basé sur **procfs** (process file system)

/root dossier personnel de l'utilisateur *root*

Les principaux répertoires

/usr répertoire des applications partagées par différentes machines ou utilisateurs. Les programmes dans /**usr** ne sont pas nécessaires au démarrage.

/media points de montages des médias amovibles (clés USB...)

/proc Répertoire virtuel avec les informations système (l'état du système, noyau Linux, etc) basé sur **procfs** (process file system)

/root dossier personnel de l'utilisateur *root*

/var Contient des données mises à jour par différents programmes durant le fonctionnement du système.

Les principaux répertoires

/usr répertoire des applications partagées par différentes machines ou utilisateurs. Les programmes dans /**usr** ne sont pas nécessaires au démarrage.

/media points de montages des médias amovibles (clés USB...)

/proc Répertoire virtuel avec les informations système (l'état du système, noyau Linux, etc) basé sur **procfs** (process file system)

/root dossier personnel de l'utilisateur *root*

/var Contient des données mises à jour par différents programmes durant le fonctionnement du système.

/tmp les fichiers temporaires.

le répertoire /**usr**

Données des applications des utilisateurs.

/usr/bin Commandes utilisables par tous les utilisateurs, et non nécessaires lors du démarrage du système.

le répertoire /**usr**

Données des applications des utilisateurs.

- /usr/bin Commandes utilisables par tous les utilisateurs, et non nécessaires lors du démarrage du système.
- /usr/sbin Commandes réservées au super-utilisateur , et non nécessaires lors du démarrage du système.

le répertoire /**usr**

Données des applications des utilisateurs.

- /usr/bin Commandes utilisables par tous les utilisateurs, et non nécessaires lors du démarrage du système.
- /usr/sbin Commandes réservées au super-utilisateur , et non nécessaires lors du démarrage du système.
- /usr/lib le dossier des librairies utilisées par les applications

le répertoire /**usr**

Données des applications des utilisateurs.

/usr/bin	Commandes utilisables par tous les utilisateurs, et non nécessaires lors du démarrage du système.
/usr/sbin	Commandes réservées au super-utilisateur , et non nécessaires lors du démarrage du système.
/usr/lib	le dossier des librairies utilisées par les applications
/usr/local	applications installées localement par l'admnistrateur système.

le répertoire /**usr**

Données des applications des utilisateurs.

- /usr/bin Commandes utilisables par tous les utilisateurs, et non nécessaires lors du démarrage du système.
- /usr/sbin Commandes réservées au super-utilisateur , et non nécessaires lors du démarrage du système.
- /usr/lib le dossier des librairies utilisées par les applications
- /usr/local applications installées localement par l'admnistrateur système.
- /usr/src Les sources des applications que l'on peut compiler

le répertoire /**usr**

Données des applications des utilisateurs.

/usr/bin	Commandes utilisables par tous les utilisateurs, et non nécessaires lors du démarrage du système.
/usr/sbin	Commandes réservées au super-utilisateur , et non nécessaires lors du démarrage du système.
/usr/lib	le dossier des librairies utilisées par les applications
/usr/local	applications installées localement par l'admnistrateur système.
/usr/src	Les sources des applications que l'on peut compiler
/usr/share	le dossier avec les fichiers qui peuvent être partagés avec toutes les architectures (i386 -INTEL-, amd64 -AMD-, etc).

le répertoire /**usr**

Données des applications des utilisateurs.

- /usr/bin Commandes utilisables par tous les utilisateurs, et non nécessaires lors du démarrage du système.
- /usr/sbin Commandes réservées au super-utilisateur , et non nécessaires lors du démarrage du système.
- /usr/lib le dossier des librairies utilisées par les applications
- /usr/local applications installées localement par l'admnistrateur système.
- /usr/src Les sources des applications que l'on peut compiler
- /usr/share le dossier avec les fichiers qui peuvent être partagés avec toutes les architectures (i386 -INTEL-, amd64 -AMD-, etc).
- /usr/local/bin c'est ici que peuvent être placés les programmes persos à partager avec d'autres utilisateurs (il faut quand même demander les droits au superuser).

le répertoire /**usr**

Données des applications des utilisateurs.

- /usr/bin Commandes utilisables par tous les utilisateurs, et non nécessaires lors du démarrage du système.
- /usr/sbin Commandes réservées au super-utilisateur , et non nécessaires lors du démarrage du système.
- /usr/lib le dossier des librairies utilisées par les applications
- /usr/local applications installées localement par l'admnistrateur système.
- /usr/src Les sources des applications que l'on peut compiler
- /usr/share le dossier avec les fichiers qui peuvent être partagés avec toutes les architectures (i386 -INTEL-, amd64 -AMD-, etc).
- /usr/local/bin c'est ici que peuvent être placés les programmes persos à partager avec d'autres utilisateurs (il faut quand même demander les droits au superuser).

...

le répertoire /**var**

/var Le répertoire /**usr**/ étant en lecture seule, tous les programmes qui ont besoin d'écrire des fichiers journaux (log), des fichiers de données (spool) ou des fichiers de blocage (lock) devraient les écrire dans /**var**.

le répertoire /**var**

/var Le répertoire /**usr**/ étant en lecture seule, tous les programmes qui ont besoin d'écrire des fichiers journaux (log), des fichiers de données (spool) ou des fichiers de blocage (lock) devraient les écrire dans /**var**.

/var/lock Fichiers de blocage, pour interdire par exemple deux utilisations simultanées d'un périphérique.

le répertoire /**var**

/var Le répertoire /**usr**/ étant en lecture seule, tous les programmes qui ont besoin d'écrire des fichiers journaux (log), des fichiers de données (spool) ou des fichiers de blocage (lock) devraient les écrire dans /**var**.

/var/lock Fichiers de blocage, pour interdire par exemple deux utilisations simultanées d'un périphérique.

/var/run Des fichiers liés aux applications en cours de fonctionnement. Par exemple, on peut y trouver le PID de l'application.

le répertoire /**var**

/var Le répertoire /**usr**/ étant en lecture seule, tous les programmes qui ont besoin d'écrire des fichiers journaux (log), des fichiers de données (spool) ou des fichiers de blocage (lock) devraient les écrire dans /**var**.

/var/lock Fichiers de blocage, pour interdire par exemple deux utilisations simultanées d'un périphérique.

/var/run Des fichiers liés aux applications en cours de fonctionnement. Par exemple, on peut y trouver le PID de l'application.

/var/log les journaux et logs du système et des applications

le répertoire /**var**

/var Le répertoire /**usr**/ étant en lecture seule, tous les programmes qui ont besoin d'écrire des fichiers journaux (log), des fichiers de données (spool) ou des fichiers de blocage (lock) devraient les écrire dans /**var**.

/var/lock Fichiers de blocage, pour interdire par exemple deux utilisations simultanées d'un périphérique.

/var/run Des fichiers liés aux applications en cours de fonctionnement. Par exemple, on peut y trouver le PID de l'application.

/var/log les journaux et logs du système et des applications

/var/cache dossiers et fichiers de cache. Par exemple apt peut y stocker les packages pour installer ou mettre à jour le système et les applications.

le répertoire /**var**

/var	Le répertoire /**usr**/ étant en lecture seule, tous les programmes qui ont besoin d'écrire des fichiers journaux (log), des fichiers de données (spool) ou des fichiers de blocage (lock) devraient les écrire dans /**var**.
/var/lock	Fichiers de blocage, pour interdire par exemple deux utilisations simultanées d'un périphérique.
/var/run	Des fichiers liés aux applications en cours de fonctionnement. Par exemple, on peut y trouver le PID de l'application.
/var/log	les journaux et logs du système et des applications
/var/cache	dossiers et fichiers de cache. Par exemple apt peut y stocker les packages pour installer ou mettre à jour le système et les applications.
/var/spool	Pour stocker les fichiers de données des programmes.

le répertoire /**etc**

stocke les fichiers de configurations du système ainsi que des applications.
Un sous-répertoire par application

/etc/init.d et /etc/default : les fichiers liés aux daemons Linux

le répertoire /**etc**

stocke les fichiers de configurations du système ainsi que des applications.
Un sous-répertoire par application

/etc/init.d et /etc/default : les fichiers liés aux daemons Linux

/etc/password, /etc/group, /etc/shadow : les fichiers de configuration des utilisateurs Linux.

le répertoire /**etc**

stocke les fichiers de configurations du système ainsi que des applications. Un sous-répertoire par application

/etc/init.d et /etc/default : les fichiers liés aux daemons Linux

/etc/password, /etc/group, /etc/shadow : les fichiers de configuration des utilisateurs Linux.

/etc/hosts : le fichier HOSTS de Linux (liens entre adresses IP et littérales)

le répertoire /**etc**

stocke les fichiers de configurations du système ainsi que des applications. Un sous-répertoire par application

/etc/init.d et /etc/default : les fichiers liés aux daemons Linux

/etc/password, /etc/group, /etc/shadow : les fichiers de configuration des utilisateurs Linux.

/etc/hosts : le fichier HOSTS de Linux (liens entre adresses IP et littérales)

/etc/sudoers et /etc/sudoers.d : la configuration de sudo.

le répertoire /**etc**

stocke les fichiers de configurations du système ainsi que des applications. Un sous-répertoire par application

/etc/init.d et /etc/default : les fichiers liés aux daemons Linux

/etc/password, /etc/group, /etc/shadow : les fichiers de configuration des utilisateurs Linux.

/etc/hosts : le fichier HOSTS de Linux (liens entre adresses IP et littérales)

/etc/sudoers et /etc/sudoers.d : la configuration de sudo.

/etc/sysctl.conf et /etc/sysctl.d les fichiers de configuration de démarrage du noyau Linux.

le répertoire /**etc**

stocke les fichiers de configurations du système ainsi que des applications. Un sous-répertoire par application

/etc/init.d et /etc/default : les fichiers liés aux daemons Linux

/etc/password, /etc/group, /etc/shadow : les fichiers de configuration des utilisateurs Linux.

/etc/hosts : le fichier HOSTS de Linux (liens entre adresses IP et littérales)

/etc/sudoers et /etc/sudoers.d : la configuration de sudo.

/etc/sysctl.conf et /etc/sysctl.d les fichiers de configuration de démarrage du noyau Linux.

...

Des fichiers sensibles

Quelques fichiers particulièrement importants, sur lesquels repose une grande partie de la stabilité du système, voire de son simple fonctionnement

/etc/passwd : fichier des utilisateurs et leurs mots de pass. Suppression $\Longrightarrow$ impossible d'utiliser le système.

Des fichiers sensibles

Quelques fichiers particulièrement importants, sur lesquels repose une grande partie de la stabilité du système, voire de son simple fonctionnement

/etc/passwd : fichier des utilisateurs et leurs mots de pass. Suppression $\Longrightarrow$ impossible d'utiliser le système.

/etc/fstab Liste des partitions utilisées par le système et selon quelle méthode il les utilise.

Des fichiers sensibles

Quelques fichiers particulièrement importants, sur lesquels repose une grande partie de la stabilité du système, voire de son simple fonctionnement

/etc/passwd : fichier des utilisateurs et leurs mots de pass. Suppression $\Longrightarrow$ impossible d'utiliser le système.

/etc/fstab Liste des partitions utilisées par le système et selon quelle méthode il les utilise.

/boot/vmlinuz Se situe généralement, soit sous la racine (/), soit sous /boot. En fait, il s'agit du système lui-même ! Ultra sensible !

Des fichiers sensibles

Quelques fichiers particulièrement importants, sur lesquels repose une grande partie de la stabilité du système, voire de son simple fonctionnement

/etc/passwd : fichier des utilisateurs et leurs mots de pass. Suppression $\Longrightarrow$ impossible d'utiliser le système.

/etc/fstab Liste des partitions utilisées par le système et selon quelle méthode il les utilise.

/boot/vmlinuz Se situe généralement, soit sous la racine (/), soit sous /boot. En fait, il s'agit du système lui-même ! Ultra sensible !

...

Avertissement

On ne donne ici que quelques indications forcément incomplètes sur le shell **bash** :

- des principes généraux d'écritures des commandes

Avertissement

On ne donne ici que quelques indications forcément incomplètes sur le shell **bash** :

- des principes généraux d'écritures des commandes
- quelques commandes forcément incomplètes

Avertissement

On ne donne ici que quelques indications forcément incomplètes sur le shell **bash** :

- des principes généraux d'écritures des commandes
- quelques commandes forcément incomplètes
- pas d'instruction de programmation pour les scripts **bash** : on a assez à faire avec le langage C et OCAML.

Terminal et console

Terminal : environnement dans lequel on écrit et qui donne le retour des commandes. Cet environnement peut être fourni par les serveur graphique, et peut disposer de fenêtres, menus, et autres boutons, ou sans fenêtre, comme lorsque on fait Ctrl+Alt+F3 (c'est Ctrl+Alt+F7 pour retourner au serveur graphique -pour moi, c'est F2-). Peut très bien être une imprimante avec un clavier comme un téléscripteur.

Terminal et console

Terminal : environnement dans lequel on écrit et qui donne le retour des commandes. Cet environnement peut être fourni par les serveur graphique, et peut disposer de fenêtres, menus, et autres boutons, ou sans fenêtre, comme lorsque on fait Ctrl+Alt+F3 (c'est Ctrl+Alt+F7 pour retourner au serveur graphique -pour moi, c'est F2-). Peut très bien être une imprimante avec un clavier comme un téléscripteur.

shell : interpréteur de commande ; programme lancé juste après la procédure de login et qui traite les commandes passées. Le shell le plus répandu sous **Linux** est le *bash* (Bourne again shell).

Terminal et console

Terminal : environnement dans lequel on écrit et qui donne le retour des commandes. Cet environnement peut être fourni par les serveur graphique, et peut disposer de fenêtres, menus, et autres boutons, ou sans fenêtre, comme lorsque on fait Ctrl+Alt+F3 (c'est Ctrl+Alt+F7 pour retourner au serveur graphique -pour moi, c'est F2-). Peut très bien être une imprimante avec un clavier comme un téléscripteur.

shell : interpréteur de commande ; programme lancé juste après la procédure de login et qui traite les commandes passées. Le shell le plus répandu sous **Linux** est le *bash* (Bourne again shell).

Console : combinaison d'un terminal et d'un shell.

Terminal et console

Terminal : environnement dans lequel on écrit et qui donne le retour des commandes. Cet environnement peut être fourni par les serveur graphique, et peut disposer de fenêtres, menus, et autres boutons, ou sans fenêtre, comme lorsque on fait Ctrl+Alt+F3 (c'est Ctrl+Alt+F7 pour retourner au serveur graphique -pour moi, c'est F2-). Peut très bien être une imprimante avec un clavier comme un téléscripteur.

shell : interpréteur de commande ; programme lancé juste après la procédure de login et qui traite les commandes passées. Le shell le plus répandu sous **Linux** est le *bash* (Bourne again shell).

Console : combinaison d'un terminal et d'un shell.

Ouverture d'un terminal sous Ubuntu : Ctr+Alt+T.

Un terminal

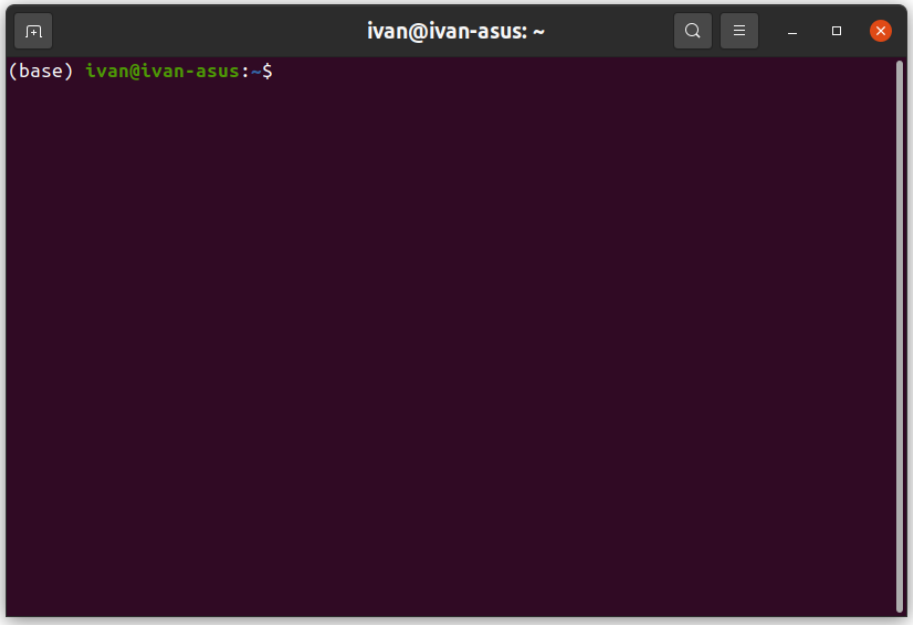


FIGURE – Un terminal dans lequel on n'a encore rien écrit

- Ouverture avec Ctrl + Alt + T
- Avant le prompt **$** : nom d'utilisateur @ nom de machine puis répertoire courant
- ~ désigne le répertoire racine de mon Home

Syntaxe générale des commandes

```
nom [-options] [argument1...]
```

- Explications :

Syntaxe générale des commandes

```
nom [-options] [argument1...]
```

- Explications :

  nom nom de la commande

Syntaxe générale des commandes

```
nom [-options] [argument1...]
```

- Explications :

nom	nom de la commande
options	représente une ou plusieurs options

Syntaxe générale des commandes

```
nom [-options] [argument1...]
```

- Explications :

nom	nom de la commande
options	représente une ou plusieurs options
argument1	est le 1er argument

Syntaxe générale des commandes

```
nom [-options] [argument1...]
```

- Explications :

nom	nom de la commande
options	représente une ou plusieurs options
argument1	est le 1er argument

- les options sont écrite le plus souvent sous la forme d'un caractère accolé à un tiret (**ls -l**) mais pas toujours, par exemple **gcc –version**.
  Les options courtes sont introduites par un tiret (**-a**), les longues par deux tirets (**–all**).

Syntaxe générale des commandes

```
nom [-options] [argument1...]
```

- Explications :

nom	nom de la commande
options	représente une ou plusieurs options
argument1	est le 1er argument

- les options sont écrite le plus souvent sous la forme d'un caractère accolé à un tiret (**ls -l**) .
- si paramètre demandé, il est séparé par un espace :
  gcc essai.c -o sortie (2 paramètres : **essai.c** et **sortie**)

Syntaxe générale des commandes

```
nom [-options] [argument1...]
```

- Explications :

nom	nom de la commande
options	représente une ou plusieurs options
argument1	est le 1er argument

- les options sont écrite le plus souvent sous la forme d'un caractère accolé à un tiret (**ls -l**) .
- si paramètre demandé, il est séparé par un espace : **gcc essai.c -o sortie** (2 paramètres : **essai.c** et **sortie**)
- les crochets indiquent un élément facultatif

Syntaxe générale des commandes

```
nom [-options] [argument1...]
```

- Explications :

nom	nom de la commande
options	représente une ou plusieurs options
argument1	est le 1er argument

- les options sont écrite le plus souvent sous la forme d'un caractère accolé à un tiret (**ls -l**) .
- si paramètre demandé, il est séparé par un espace : **gcc essai.c -o sortie** (2 paramètres : **essai.c** et **sortie**)
- les crochets indiquent un élément facultatif
- les points de suspension indiquent la possibilité de répéter un argument, par exemple **ls /etc /usr/bin**

Syntaxe générale des commandes

```
nom [-options] [argument1...]
```

- Explications :

nom	nom de la commande
options	représente une ou plusieurs options
argument1	est le 1er argument

- les options sont écrite le plus souvent sous la forme d'un caractère accolé à un tiret (**ls -l**) .
- si paramètre demandé, il est séparé par un espace : **gcc essai.c -o sortie** (2 paramètres : **essai.c** et **sortie**)
- les crochets indiquent un élément facultatif
- les points de suspension indiquent la possibilité de répéter un argument, par exemple **ls /etc /usr/bin**
- séparation par un espace ou une tabulation

Commandes internes vs externes

- Une commande *externe* est un fichier présent dans l'arborescence. Exemple : Quand un utilisateur exécute la commande **ls**, le shell demande au noyau Linux d'exécuter le fichier **/bin/ls**

```
$file /bin/ls
/bin/ls: ELF 64-bit LSB shared object, x86-64,
version 1 (SYSV), dynamically linked, [...]
```

Commandes internes vs externes

- Une commande *externe* est un fichier présent dans l'arborescence. Exemple : Quand un utilisateur exécute la commande **ls**, le shell demande au noyau Linux d'exécuter le fichier **/bin/ls**

```
$file /bin/ls
/bin/ls: ELF 64-bit LSB shared object, x86-64,
version 1 (SYSV), dynamically linked, [...]
```

- Une commande **interne** est intégrée au processus shell. Elle n'a aucune correspondance avec un fichier sur le disque. L'accès à une commande interne est plus rapide que pour une externe.

Commandes internes vs externes

- Une commande *externe* est un fichier présent dans l'arborescence. Exemple : Quand un utilisateur exécute la commande **ls**, le shell demande au noyau Linux d'exécuter le fichier **/bin/ls**

```
$file /bin/ls
/bin/ls: ELF 64-bit LSB shared object, x86-64,
version 1 (SYSV), dynamically linked, [...]
```

- Une commande **interne** est intégrée au processus shell. Elle n'a aucune correspondance avec un fichier sur le disque. L'accès à une commande interne est plus rapide que pour une externe.
- La commande **type** indique si une commande est interne ou externe.

```
$ type ls
ls est un alias vers << ls --color=auto >>
$ type cd
cd est une primitive du shell
```

Commandes internes vs externes

- Une commande *externe* est un fichier présent dans l'arborescence. Exemple : Quand un utilisateur exécute la commande **ls**, le shell demande au noyau Linux d'exécuter le fichier **/bin/ls**

```
$file /bin/ls
/bin/ls: ELF 64-bit LSB shared object, x86-64,
version 1 (SYSV), dynamically linked, [...]
```

- Une commande **interne** est intégrée au processus shell. Elle n'a aucune correspondance avec un fichier sur le disque. L'accès à une commande interne est plus rapide que pour une externe.
- La commande **type** indique si une commande est interne ou externe.

```
$ type ls
ls est un alias vers << ls --color=auto >>
$ type cd
cd est une primitive du shell
```

- certaines commande ont une version externe et une interne.

la commande **man**

- Pour connaître le mode d'emploi d'une commande, taper **man nomDeLaCommande** (ex : **man ls** pour connaître le manuel de **ls**)

la commande **man**

- Pour connaître le mode d'emploi d'une commande, taper **man nomDeLaCommande** (ex : **man ls** pour connaître le manuel de **ls**)
- Consulter la documentation Ubuntu

la commande **man**

- Pour connaître le mode d'emploi d'une commande, taper **man nomDeLaCommande** (ex : **man ls** pour connaître le manuel de **ls**)
- Consulter la documentation Ubuntu
- Le manuel est décomposé en plusieurs sections 1 : Programmes exécutables ou commandes de l'interpréteur de commandes (shell) ; 2 : Appels système (Fonctions fournies par le noyau) ; 3 : Appels de bibliothèque (fonctions fournies par des bibliothèques) ; 4 : Fichiers spéciaux (situés généralement dans /dev) ; 5 : Formats des fichiers et conventions (Par exemple /etc/passwd) ;...

la commande **man**

- Pour connaître le mode d'emploi d'une commande, taper **man nomDeLaCommande** (ex : **man ls** pour connaître le manuel de **ls**)
- Consulter la documentation Ubuntu
- Le manuel est décomposé en plusieurs sections 1 : Programmes exécutables ou commandes de l'interpréteur de commandes (shell) ; 2 : Appels système (Fonctions fournies par le noyau) ; 3 : Appels de bibliothèque (fonctions fournies par des bibliothèques) ; 4 : Fichiers spéciaux (situés généralement dans /dev) ; 5 : Formats des fichiers et conventions (Par exemple /etc/passwd) ;...
- Parfois deux pages de manuel ont le même nom comme **printf** (en section 1 et 3). Entrer **man 1 printf** ou **man 3 printf** pour spécifier.

la commande **man**

- Pour connaître le mode d'emploi d'une commande, taper **man nomDeLaCommande** (ex : **man ls** pour connaître le manuel de **ls**)
- Consulter la documentation Ubuntu
- Le manuel est décomposé en plusieurs sections 1 : Programmes exécutables ou commandes de l'interpréteur de commandes (shell) ; 2 : Appels système (Fonctions fournies par le noyau) ; 3 : Appels de bibliothèque (fonctions fournies par des bibliothèques) ; 4 : Fichiers spéciaux (situés généralement dans /dev) ; 5 : Formats des fichiers et conventions (Par exemple /etc/passwd) ;...
- Parfois deux pages de manuel ont le même nom comme **printf** (en section 1 et 3). Entrer **man 1 printf** ou **man 3 printf** pour spécifier.
- Entrer **q** pour quitter le manuel.

Chemins

consulter l'exemple d'arborescence

- Un élément de l'arborescence est repéré par son nom (par exemple **nom.jpg**) précédé de :

/ : sépare les noms de fichiers

Chemins

consulter l'exemple d'arborescence

- Un élément de l'arborescence est repéré par son nom (par exemple **nom.jpg**) précédé de :
  - son chemin absolu depuis la racine (ex : **/home/sue/Pictures/family/nom.jpg**)

/ : sépare les noms de fichiers

Chemins

consulter l'exemple d'arborescence

- Un élément de l'arborescence est repéré par son nom (par exemple **nom.jpg**) précédé de :
  - son chemin absolu depuis la racine (ex : **/home/sue/Pictures/family/nom.jpg**)
  - son chemin relatif depuis le répertoire courant. Par exemple, si je suis dans **pets** : **../../family/nom.jpg**

/ : sépare les noms de fichiers

Chemins

consulter l'exemple d'arborescence

- Un élément de l'arborescence est repéré par son nom (par exemple **nom.jpg**) précédé de :
  - son chemin absolu depuis la racine (ex : **/home/sue/Pictures/family/nom.jpg**)
  - son chemin relatif depuis le répertoire courant. Par exemple, si je suis dans **pets** : **../../family/nom.jpg**

  / : sépare les noms de fichiers
- ~ : répertoire racine du Home

Commentaire

Les commentaires ne sont pas interprétés :

- La commande **kalin** n'existe pas, elle soulève une erreur :

```
$ kalin
La commande <<kalin>> n'a pas été trouvée, ...
```

Commentaire

Les commentaires ne sont pas interprétés :

- La commande **kalin** n'existe pas, elle soulève une erreur :

```
$ kalin
La commande <<kalin>> n'a pas été trouvée, ...
```

- Le caractère **#** n'est pas interprété. On peut écrire **kalin** sans erreur :

```
$ # kalin
$
```

Contenu d'un répertoire

La commande **ls** donne le contenu d'un répertoire :

- sans argument : les entrées du répertoire courant **ls**

Contenu d'un répertoire

La commande **ls** donne le contenu d'un répertoire :

- sans argument : les entrées du répertoire courant **ls**
- avec argument : les entrées repérées par le (ou les) argument(s) : **ls myFile, ls myRep, ls /etc /usr/bin**

Contenu d'un répertoire

La commande **ls** donne le contenu d'un répertoire :

- sans argument : les entrées du répertoire courant **ls**
- avec argument : les entrées repérées par le (ou les) argument(s) : **ls myFile, ls myRep, ls /etc /usr/bin**
- pour afficher les fichiers cachés **ls -a** (indique notamment **.bashrc** si je suis en ~)

Contenu d'un répertoire

La commande **ls** donne le contenu d'un répertoire :

- sans argument : les entrées du répertoire courant **ls**
- avec argument : les entrées repérées par le (ou les) argument(s) : **ls myFile, ls myRep, ls /etc /usr/bin**
- pour afficher les fichiers cachés **ls -a** (indique notamment **.bashrc** si je suis en ~)
- pour pour tous les attributs (type, droits, liens physiques, propriétaire, groupe, taille, date, nom) **ls -l**

Contenu d'un répertoire

La commande **ls** donne le contenu d'un répertoire :

- sans argument : les entrées du répertoire courant **ls**
- avec argument : les entrées repérées par le (ou les) argument(s) : **ls myFile, ls myRep, ls /etc /usr/bin**
- pour afficher les fichiers cachés **ls -a** (indique notamment **.bashrc** si je suis en ~)
- pour pour tous les attributs (type, droits, liens physiques, propriétaire, groupe, taille, date, nom) **ls -l**
- **ls -al** au lieu de **ls -a -l**.

Affichage d'une chaîne de caractère

La commande **echo** affiche une ligne de texte

- (Le caractère **#** indique le début d'un commentaire

```
$ echo 'coucou' # afficher coucou
coucou
```

Affichage d'une chaîne de caractère

La commande **echo** affiche une ligne de texte

- (Le caractère **#** indique le début d'un commentaire

```
$ echo 'coucou' # afficher coucou
coucou
```

- le choix des guillements est important : « ' » (touche 4), « " » (touche 3) et « ` » (ALT GR + 7) n'ont pas le même sens.

Affichage d'une chaîne de caractère

La commande **echo** affiche une ligne de texte

- (Le caractère **#** indique le début d'un commentaire

```
$ echo 'coucou' # afficher coucou
coucou
```

- le choix des guillements est important : « ' » (touche 4), « " » (touche 3) et « ` » (ALT GR + 7) n'ont pas le même sens.
- Pour afficher le contenu d'une variable d'environnement :

```
$ echo $LANG
fr_FR.UTF-8
```

Les métacaractères

Les métacaractères du shell permettent :

- de construire des chaînes de caractères génériques

Les métacaractères

Les métacaractères du shell permettent :

- de construire des chaînes de caractères génériques
- de modifier l'interprétation d'une commande

Métacaractères de construction

- * désigne une chaîne de caractères quelconque

Métacaractères de construction

- * désigne une chaîne de caractères quelconque
- ? désigne un caractère quelconque

Métacaractères de construction

- * désigne une chaîne de caractères quelconque
- **?** désigne un caractère quelconque
- **[...]** désigne les caractères entre crochets, définis par énumération ou par un intervalle :

Métacaractères de construction

- * désigne une chaîne de caractères quelconque
- **?** désigne un caractère quelconque
- **[...]** désigne les caractères entre crochets, définis par énumération ou par un intervalle :
  - **[Aa]** désigne les caractères A ou a,

Métacaractères de construction

- * désigne une chaîne de caractères quelconque
- **?** désigne un caractère quelconque
- **[...]** désigne les caractères entre crochets, définis par énumération ou par un intervalle :
  - **[Aa]** désigne les caractères A ou a,
  - **[0-9a-zA-Z]** désigne un caractère alphanumérique quelconque.

Métacaractères de construction

- * désigne une chaîne de caractères quelconque
- **?** désigne un caractère quelconque
- **[...]** désigne les caractères entre crochets, définis par énumération ou par un intervalle :
  - **[Aa]** désigne les caractères A ou a,
  - **[0-9a-zA-Z]** désigne un caractère alphanumérique quelconque.
  - **[!0-9]** désigne l'ensemble des caractères sauf les chiffres.

Métacaractères de construction

Voici le contenu du répertoire courant :

```
$ ls
alain Ali tata titi toto tutu zut
```

- **ls t[ao]t[ao]** retourne **tata** et **toto**,

Métacaractères de construction

Voici le contenu du répertoire courant :

```
$ ls
alain Ali tata titi toto tutu zut
```

- **ls t[ao]t[ao]** retourne **tata** et **toto**,
- **ls ? ? ?** retourne les noms de 3 lettres donc **zut** et **Ali**

Métacaractères de construction

Voici le contenu du répertoire courant :

```
$ ls
alain Ali tata titi toto tutu zut
```

- **ls t[ao]t[ao]** retourne **tata** et **toto**,
- **ls ? ? ?** retourne les noms de 3 lettres donc **zut** et **Ali**
- **ls A*** retourne les noms qui comanecent par A donc **Ali**

Métacaractères de construction

Voici le contenu du répertoire courant :

```
$ ls
alain Ali tata titi toto tutu zut
```

- **ls t[ao]t[ao]** retourne **tata** et **toto**,
- **ls ? ? ?** retourne les noms de 3 lettres donc **zut** et **Ali**
- **ls A*** retourne les noms qui comanecent par A donc **Ali**
- **t ? ?o** retourne les noms de 4 lettres qui terminent par o et commencent par t donc **toto**

Métacaractères de construction

Voici le contenu du répertoire courant :

```
$ ls
alain Ali tata titi toto tutu zut
```

- **ls t[ao]t[ao]** retourne **tata** et **toto**,
- **ls ? ? ?** retourne les noms de 3 lettres donc **zut** et **Ali**
- **ls A*** retourne les noms qui comanecent par A donc **Ali**
- **t ? ?o** retourne les noms de 4 lettres qui terminent par o et commencent par t donc **toto**
- **ls [!b-z]*** désigne les noms qui ne commencent pas par une lettre entre b et z donc **alain** et **Ali**.

Positionnement/recherche dans l'arborescence

pwd affiche le nom absolu du répertoire de travail

Positionnement/recherche dans l'arborescence

pwd affiche le nom absolu du répertoire de travail

cd change le répertoire de travail.

Positionnement/recherche dans l'arborescence

pwd affiche le nom absolu du répertoire de travail

cd change le répertoire de travail.

- Avec argument : se rend à la destination indiquée ;

Positionnement/recherche dans l'arborescence

pwd affiche le nom absolu du répertoire de travail

cd change le répertoire de travail.

- Avec argument : se rend à la destination indiquée ;
- sans argument : retourne au répertoire de connexion du user.

Positionnement/recherche dans l'arborescence

pwd affiche le nom absolu du répertoire de travail

cd change le répertoire de travail.

- Avec argument : se rend à la destination indiquée ;
- sans argument : retourne au répertoire de connexion du user.

ls liste les entrées d'un répertoire (déjà vu).

Positionnement/recherche dans l'arborescence

pwd affiche le nom absolu du répertoire de travail

cd change le répertoire de travail.

- Avec argument : se rend à la destination indiquée ;
- sans argument : retourne au répertoire de connexion du user.

ls liste les entrées d'un répertoire (déjà vu).

find pour trouver un ou plusieurs fichiers dans une arborescence. Beaucoup d'options (consulter le manuel)

Rechercher

- **find** cherche récursivement dans l'arborescence à partir du point indiqué. **find /usr -name "ls*"** cherche les fichiers dont le nom commence par **ls** dans le répertoire /**usr** et ses sous-répertoires. Il y en a beaucoup !

Rechercher

- **find** cherche récursivement dans l'arborescence à partir du point indiqué. **find /usr -name "ls*"** cherche les fichiers dont le nom commence par **ls** dans le répertoire /**usr** et ses sous-répertoires. Il y en a beaucoup !
- L'option **-type** permet de ne chercher que les fichiers (**f**) ou les répertoires (**d**). **find /var/log/ -type d -name "*sm*"** : chercher les répertoires dont le nom contient sm

Rechercher

- **find** cherche récursivement dans l'arborescence à partir du point indiqué. **find /usr -name "ls*"** cherche les fichiers dont le nom commence par **ls** dans le répertoire /**usr** et ses sous-répertoires. Il y en a beaucoup !
- L'option **-type** permet de ne chercher que les fichiers (**f**) ou les répertoires (**d**). **find /var/log/ -type d -name "*sm*"** : chercher les répertoires dont le nom contient sm
- recherche par taille : **find /Téléchargements -size +20M -size -40M** cherche les fichiers dont la taille est comprise entre 20 Mo et 40Mo.

Rechercher

- **find** cherche récursivement dans l'arborescence à partir du point indiqué. **find /usr -name "ls*"** cherche les fichiers dont le nom commence par **ls** dans le répertoire /**usr** et ses sous-répertoires. Il y en a beaucoup !
- L'option **-type** permet de ne chercher que les fichiers (**f**) ou les répertoires (**d**). **find /var/log/ -type d -name "*sm*"** : chercher les répertoires dont le nom contient sm
- recherche par taille : **find /Téléchargements -size +20M -size -40M** cherche les fichiers dont la taille est comprise entre 20 Mo et 40Mo.
- Recherche par utilisateur : **find /tmp -user adrien** cherche dans /**tmp** les fichiers dont le propriétaire est **adrien**.

Rechercher

- **find** cherche récursivement dans l'arborescence à partir du point indiqué. **find /usr -name "ls*"** cherche les fichiers dont le nom commence par **ls** dans le répertoire /**usr** et ses sous-répertoires. Il y en a beaucoup !
- L'option **-type** permet de ne chercher que les fichiers (**f**) ou les répertoires (**d**). **find /var/log/ -type d -name "*sm*"** : chercher les répertoires dont le nom contient sm
- recherche par taille : **find /Téléchargements -size +20M -size -40M** cherche les fichiers dont la taille est comprise entre 20 Mo et 40Mo.
- Recherche par utilisateur : **find /tmp -user adrien** cherche dans /**tmp** les fichiers dont le propriétaire est **adrien**.
- Beaucoup d'autres options : par date de création, date de dernière modification, par type de permissions, recherche de fichiers vides etc...

Consulter le contenu d'un fichier texte

Commandes de base :

- **cat monfichier**, **more monfichier** : affichage simple et page par page ;

Consulter le contenu d'un fichier texte

Commandes de base :

- **cat monfichier**, **more monfichier** : affichage simple et page par page ;
- **head monfichier**, **head -n monfichier** : affichage des *n* premières lignes ;

Consulter le contenu d'un fichier texte

Commandes de base :

- **cat monfichier**, **more monfichier** : affichage simple et page par page ;
- **head monfichier**, **head -n monfichier** : affichage des *n* premières lignes ;
- **tail monfichier**, **tail -n monfichier** : affichage des *n* dernières lignes ;

Consulter le contenu d'un fichier texte

Commandes de base :

- **cat monfichier**, **more monfichier** : affichage simple et page par page ;
- **head monfichier**, **head -n monfichier** : affichage des *n* premières lignes ;
- **tail monfichier**, **tail -n monfichier** : affichage des *n* dernières lignes ;
- **wc monfichier** : affichage du nombre de lignes, de mots, de caractères. Options **-l**, **-w** et **-c** pour les nombres de lignes, de mots et de caractères.

Historique

- Le shell **bash** enregistre toutes les commandes tapées et permet de les rappeler pour les ré-exécuter soit telles quelles, soit modifiées.

Historique

- Le shell **bash** enregistre toutes les commandes tapées et permet de les rappeler pour les ré-exécuter soit telles quelles, soit modifiées.
- La commande **history** permet de lister le contenu de l'historique des commandes, de façon numérotée. Le caractère **!** permet de rappeler une commande.

Historique

- Le shell **bash** enregistre toutes les commandes tapées et permet de les rappeler pour les ré-exécuter soit telles quelles, soit modifiées.
- La commande **history** permet de lister le contenu de l'historique des commandes, de façon numérotée. Le caractère **!** permet de rappeler une commande.

  !! rappelle la dernière commande

Historique

- Le shell **bash** enregistre toutes les commandes tapées et permet de les rappeler pour les ré-exécuter soit telles quelles, soit modifiées.
- La commande **history** permet de lister le contenu de l'historique des commandes, de façon numérotée. Le caractère **!** permet de rappeler une commande.

  !! rappelle la dernière commande

  !n rappelle la commande numéro *n*

Historique

- Le shell **bash** enregistre toutes les commandes tapées et permet de les rappeler pour les ré-exécuter soit telles quelles, soit modifiées.
- La commande **history** permet de lister le contenu de l'historique des commandes, de façon numérotée. Le caractère **!** permet de rappeler une commande.

!!	rappelle la dernière commande
!n	rappelle la commande numéro *n*
!chaine	rappelle la dernière commande commençant par *chaine*

Historique

- Le shell **bash** enregistre toutes les commandes tapées et permet de les rappeler pour les ré-exécuter soit telles quelles, soit modifiées.
- La commande **history** permet de lister le contenu de l'historique des commandes, de façon numérotée. Le caractère **!** permet de rappeler une commande.

!!	rappelle la dernière commande
!n	rappelle la commande numéro *n*
!chaine	rappelle la dernière commande commençant par *chaine*

- On peut aussi utiliser les flèches haut et bas pour naviguer dans l'historique des commandes.

Inode

- Un *nœud d'index* ou *inode* (contraction de l'anglais index et node) est une structure de données contenant des informations à propos d'un fichier ou répertoire.

Inode

- Un *nœud d'index* ou *inode* (contraction de l'anglais index et node) est une structure de données contenant des informations à propos d'un fichier ou répertoire.
- Les inodes contiennent toutes les informations sur les fichiers à part le (ou les) nom(s).

Inode

- Un *nœud d'index* ou *inode* (contraction de l'anglais index et node) est une structure de données contenant des informations à propos d'un fichier ou répertoire.
- Les inodes contiennent toutes les informations sur les fichiers à part le (ou les) nom(s).
- Pour connaître le numéro d'inode d'un fichier :

```
$ ls -i asup # num d'inode du fichier asup
10224747 asup
```

Nombre d'inodes utilisées

Informations sur les inodes utilisés par les partitions avec **df -i** :

```
$ df -i
Sys. de fichiers Inœuds IUtil. ILibre IUti% Monté sur
udev 2016836 580 2016256 1% /dev
tmpfs 2032479 1076 2031403 1% /run
/dev/nvme0n1p5 31227904 736385 30491519 3% /
tmpfs 2032479 1 2032478 1% /dev/shm
tmpfs 2032479 5 2032474 1% /run/lock
tmpfs 2032479 18 2032461 1% /sys/fs/cgroup
/dev/loop1 10803 10803 0 100% /snap/core18/2074
/dev/loop2 293 293 0 100% /snap/discord/122
...
```

Répertoire

Le contenu d'un répertoire est un catalogue qui met en correspondance des noms des fichiers et des sous-répertoires avec des numéros d'inode.

- La commande **mkdir Rep** crée un répertoire de nom **Rep** dans le répertoire courant. Ce répertoire n'est pas tout à fait vide car il contient deux entrées : La référence au répertoire parent (du répertoire créé) noté « **..** » La référence (autoréférence) au répertoire lui-même noté « **.** »
- La suppression d'un répertoire vide s'effectue par **rmdir rep**

Répertoire

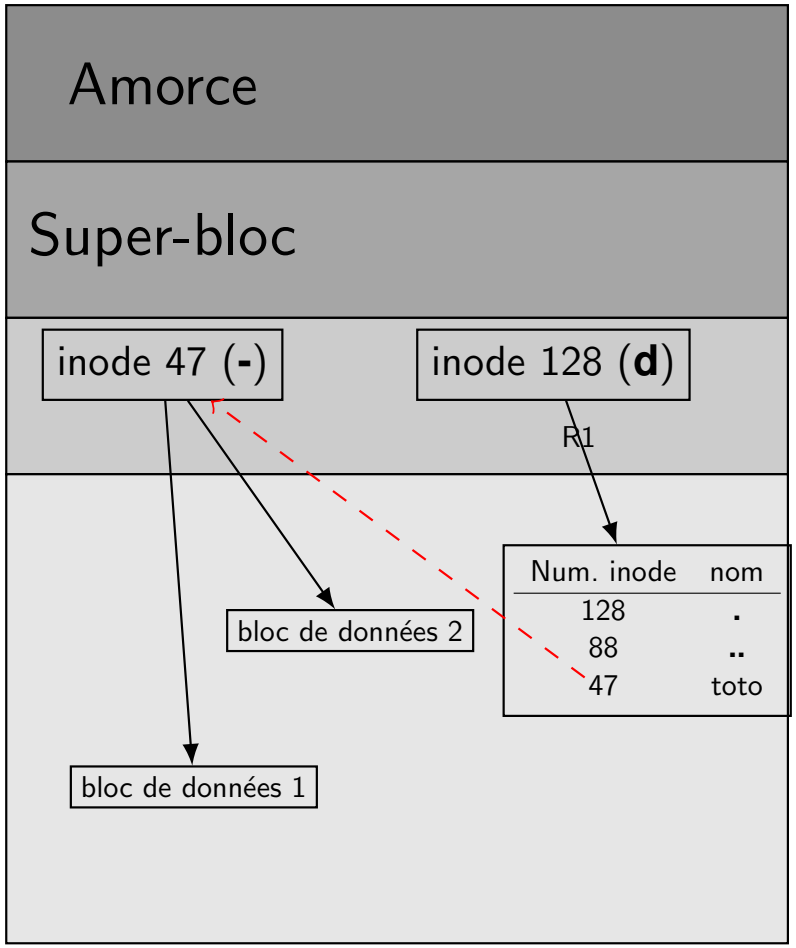


FIGURE – Un répertoire contenant un fichier toto

Taille du contenu d'un répertoire

- **du -h -d 1 monRepertoire** : taille des fichiers et sous-répertoires.

Taille du contenu d'un répertoire

- **du -h -d 1 monRepertoire** : taille des fichiers et sous-répertoires.
  - L'option **-h** force un affichage « human readable » (par exemple, 1k, 236M, 2G).

Taille du contenu d'un répertoire

- **du -h -d 1 monRepertoire** : taille des fichiers et sous-répertoires.
  - L'option **-h** force un affichage « human readable » (par exemple, 1k, 236M, 2G).
  - L'option **-d** affiche la taille totale du répertoire exploré et pas seulement la taille de ses constituants. Et le paramètre 1 indique la profondeur de l'exploration (ici, on s'arrête aux fils, avec 2 comme paramètre , ce serait aux petits-fils)

Taille du contenu d'un répertoire

- **du -h -d 1 monRepertoire** : taille des fichiers et sous-répertoires.
  - L'option **-h** force un affichage « human readable » (par exemple, 1k, 236M, 2G).
  - L'option **-d** affiche la taille totale du répertoire exploré et pas seulement la taille de ses constituants. Et le paramètre 1 indique la profondeur de l'exploration (ici, on s'arrête aux fils, avec 2 comme paramètre , ce serait aux petits-fils)
- La commande **ncdu nomDuRepertoire**, plus conviviale, permet de connaître la place prise par les fichiers et dossiers en navigant dans l'arborescence. Par exemple **ncdu /home** indique les tailles des différents répertoires utilisateurs.

Créer, remplir, vider, supprimer un répertoire

Exemple concret

- ```
  $ mkdir Asup # Creer répertoire Asup
  ```
  ```

Créer, remplir, vider, supprimer un répertoire

Exemple concret

- ```
  $ mkdir Asup # Creer répertoire Asup
  ```

- ```
  $ cd Asup # aller au répertoire Asup
  $ ls # pas de contenu
  ```

Créer, remplir, vider, supprimer un répertoire

Exemple concret

```
$ mkdir Asup # Creer répertoire Asup
```

```
$ cd Asup # aller au répertoire Asup
$ ls # pas de contenu
```

```
$ touch asup # créer le fichier vide asup
$ ls -al # afficher fichiers cachés
total 8
drwxrwxr-x 2 ivan ivan 4096 août 19 15:28 .
drwxrwxr-x 3 ivan ivan 4096 août 19 15:28 ..
-rw-rw-r-- 1 ivan ivan 0 août 19 15:28 asup
```

Créer, remplir, vider, supprimer un répertoire

Exemple concret

- ```
  $ mkdir Asup # Creer répertoire Asup
  ```

- ```
  $ cd Asup # aller au répertoire Asup
  $ ls # pas de contenu
  ```

- ```
  $ touch asup # créer le fichier vide asup
  $ ls -al # afficher fichiers cachés
  total 8
  drwxrwxr-x 2 ivan ivan 4096 août 19 15:28 .
  drwxrwxr-x 3 ivan ivan 4096 août 19 15:28 ..
  -rw-rw-r-- 1 ivan ivan 0 août 19 15:28 asup
  ```

- ```
  $ rm asup # supprimer asup
  $ ls # plus de contenu
  ```

Créer, remplir, vider, supprimer un répertoire

Exemple concret

```
$ cd .. # revenir au père
$ rmdir Asup # supprime rep Asup
```

Créer, remplir, vider, supprimer un répertoire

Exemple concret

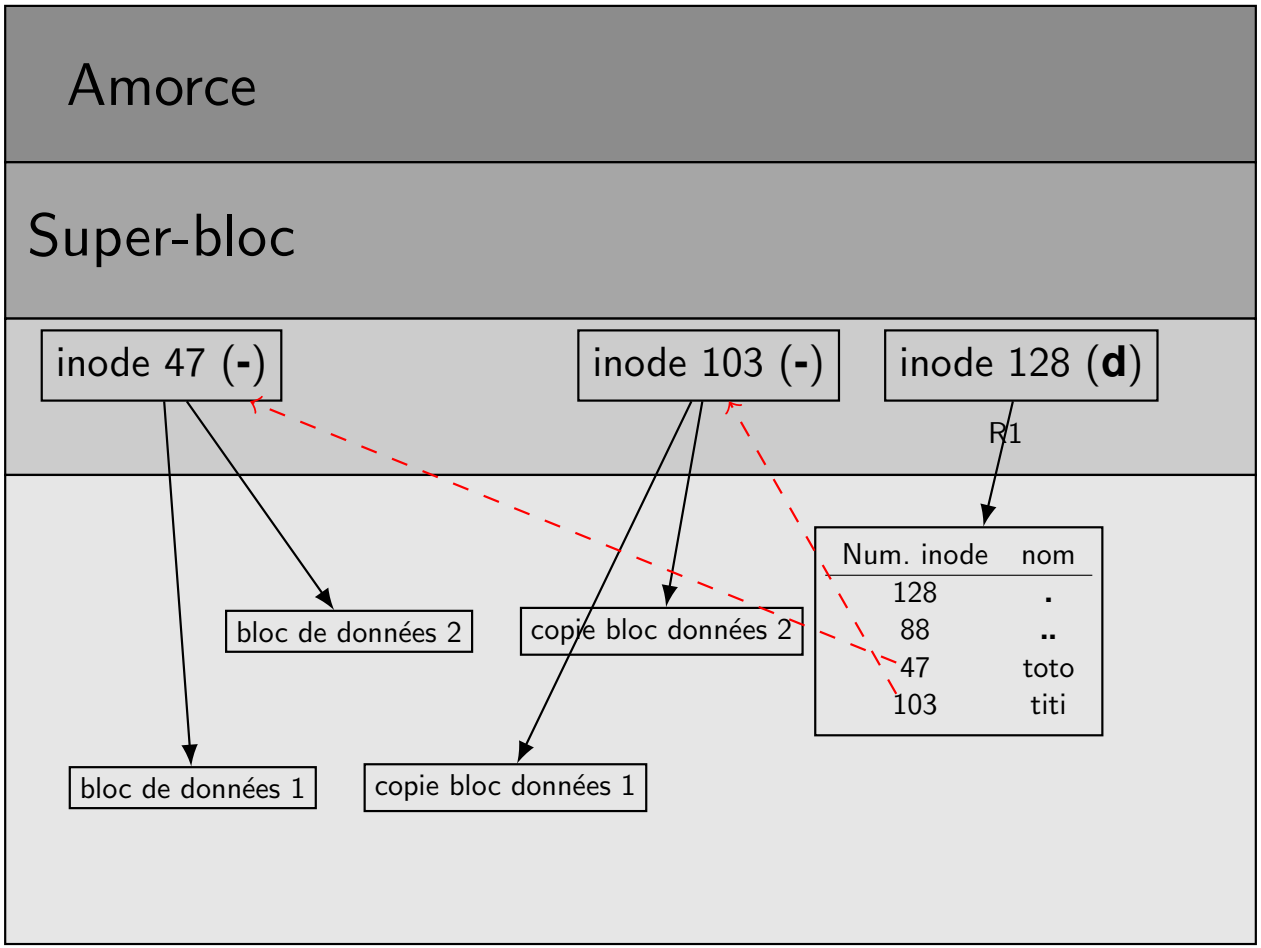
```
$ cd .. # revenir au père
$ rmdir Asup # supprime rep Asup
```

- Les répertoires **Nouveau** et son fils **AutreNouveau** sont créés simultannéments puis supprimés de même

```
$ mkdir -p Nouveau/AutreNouveau # creer un répertoire et son fils
$ rmdir -p Nouveau/AutreNouveau # supprimer un répertoire et son fils
```

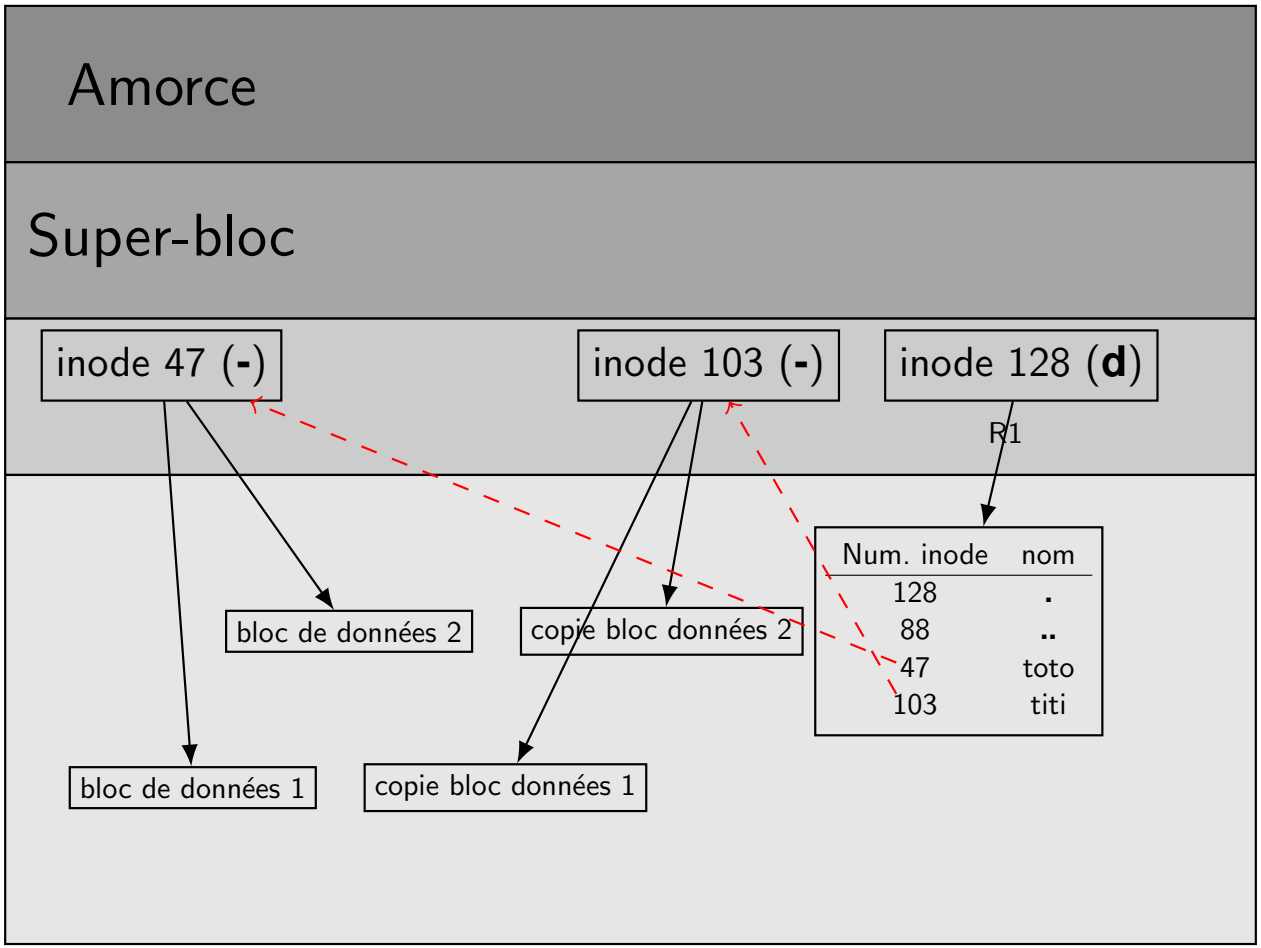
Copier

- copier un fichier : **cp** <**source**> <**but**>.



Copier

- copier un fichier : **cp** <**source**> <**but**>.
- **cp toto titi** duplique les données de **toto**. Un inode autre que celui de **toto** pointe sur les données dupliquées.



Renommage ou déplacement

La commande **mv** déplace et/ou renomme un fichier ou un répertoire.

- **mv R1/toto R2/titi** : Déplace le fichier **toto** en **titi** et change de répertoire.

Renommage ou déplacement

La commande **mv** déplace et/ou renomme un fichier ou un répertoire.

- **mv R1/toto R2/titi** : Déplace le fichier **toto** en **titi** et change de répertoire.
- **mv R1/toto R2/** : Déplace **toto** dans le répertoire **R2**.

Renommage ou déplacement

La commande **mv** déplace et/ou renomme un fichier ou un répertoire.

- **mv R1/toto R2/titi** : Déplace le fichier **toto** en **titi** et change de répertoire.
- **mv R1/toto R2/** : Déplace **toto** dans le répertoire **R2**.
- **mv R1 R2** Renomme le répertoire **R1** en **R2**.

Renommage ou déplacement

La commande **mv** déplace et/ou renomme un fichier ou un répertoire.

- **mv R1/toto R2/titi** : Déplace le fichier **toto** en **titi** et change de répertoire.
- **mv R1/toto R2/** : Déplace **toto** dans le répertoire **R2**.
- **mv R1 R2** Renomme le répertoire **R1** en **R2**.
- **mv toto titi** Renomme le fichier **toto** en **titi** en restant dans le répertoire courant..

Renommage ou déplacement

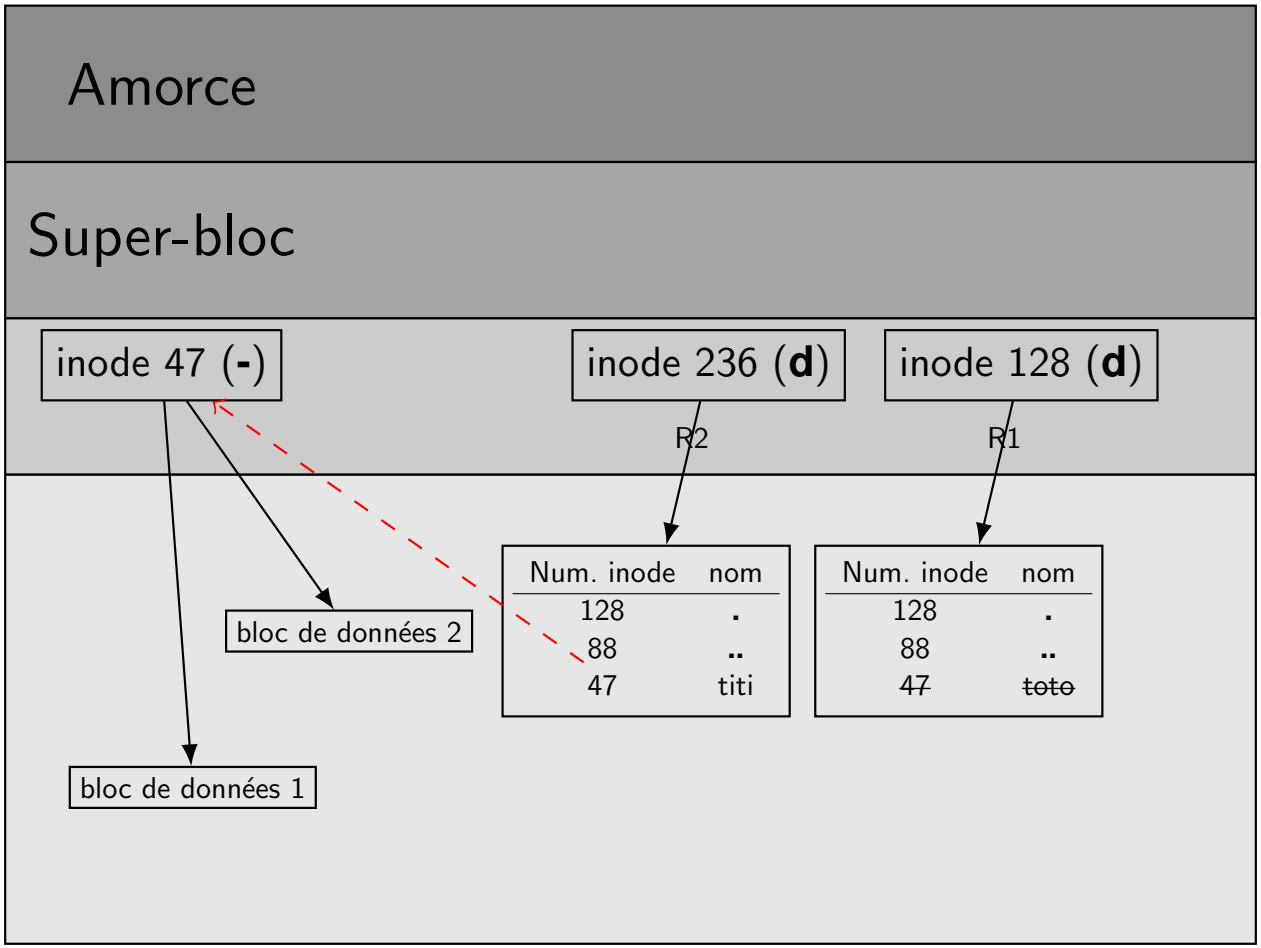


FIGURE – Déplacement **mv toto R2/titi** : par rapport à la figure 4, **titi** de **R2** est l'ancien **toto** de **R1**

Lien physique

- Un *lien physique* est une référence directe à un fichier via son inode.

Lien physique

- Un *lien physique* est une référence directe à un fichier via son inode.
- Avantage : On peut changer le contenu ou la localisation du fichier original, le lien physique restera valide.

Lien physique

- Un *lien physique* est une référence directe à un fichier via son inode.
- Avantage : On peut changer le contenu ou la localisation du fichier original, le lien physique restera valide.
- Inconvénient : il faut rester dans le même volume physique (la même partition).

Lien physique

- Un *lien physique* est une référence directe à un fichier via son inode.
- Avantage : On peut changer le contenu ou la localisation du fichier original, le lien physique restera valide.
- Inconvénient : il faut rester dans le même volume physique (la même partition).
- syntaxe : **ln toto ../R2/titi**. Voir figure 8.

Lien physique

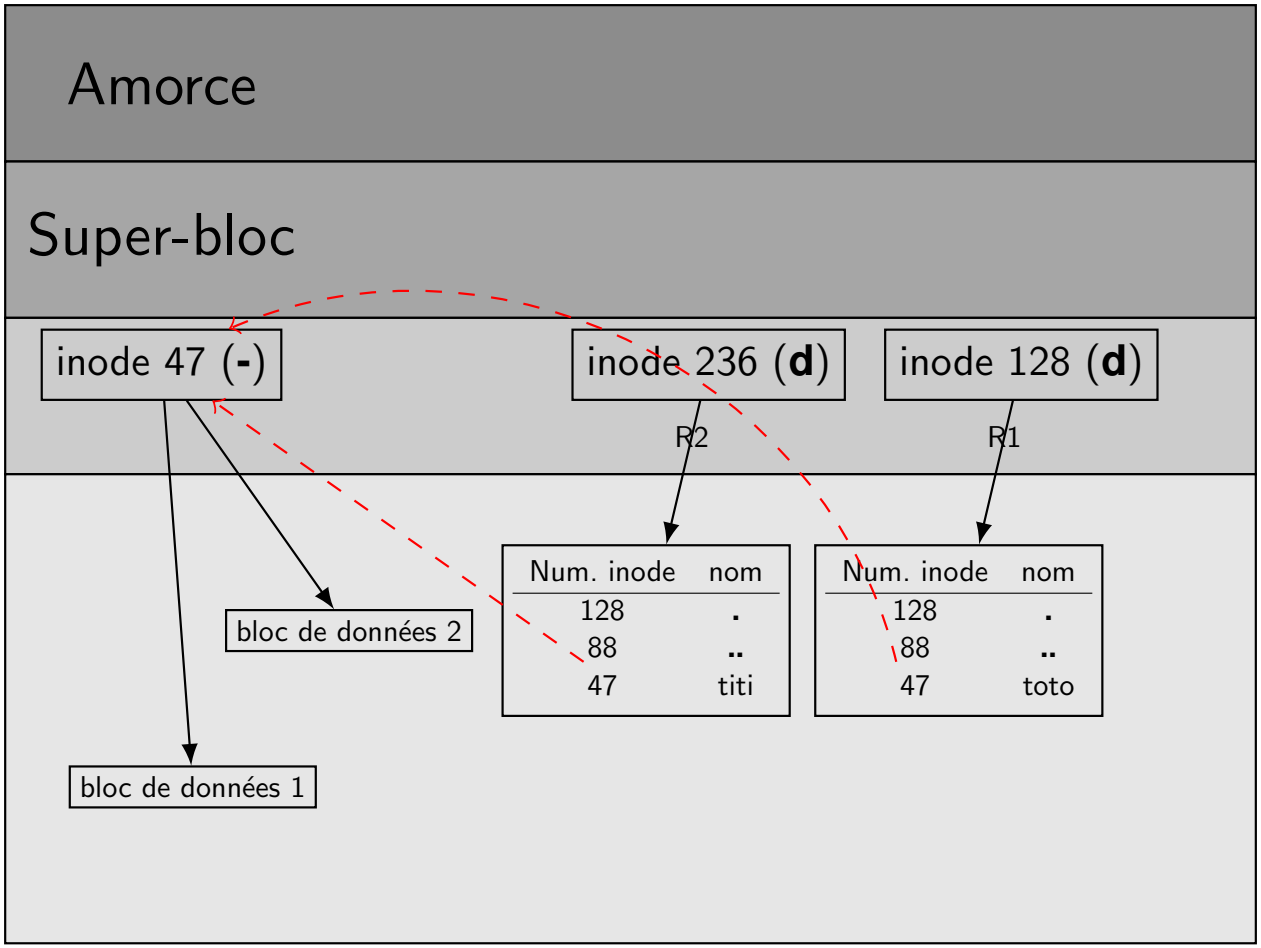


FIGURE – Lien **ln toto R2/titi** : **titi** et **toto** pointent vers le même inode

Lien physique

- Un *lien physique* est une référence directe à un fichier via son inode.

Lien physique

- Un *lien physique* est une référence directe à un fichier via son inode.
- Avantage : On peut changer le contenu ou la localisation du fichier original, le lien physique restera valide.

Lien physique

- Un *lien physique* est une référence directe à un fichier via son inode.
- Avantage : On peut changer le contenu ou la localisation du fichier original, le lien physique restera valide.
- Inconvénient : il faut rester dans le même volume physique (la même partition).

Lien physique

- Un *lien physique* est une référence directe à un fichier via son inode.
- Avantage : On peut changer le contenu ou la localisation du fichier original, le lien physique restera valide.
- Inconvénient : il faut rester dans le même volume physique (la même partition).
- On ne peut pas faire de lien physique avec un répertoire.

Lien physique

- Un *lien physique* est une référence directe à un fichier via son inode.
- Avantage : On peut changer le contenu ou la localisation du fichier original, le lien physique restera valide.
- Inconvénient : il faut rester dans le même volume physique (la même partition).
- On ne peut pas faire de lien physique avec un répertoire.
- syntaxe : **ln toto ../R2/titi**. Voir figure 8.

Lien physique

- Un *lien physique* est une référence directe à un fichier via son inode.
- Avantage : On peut changer le contenu ou la localisation du fichier original, le lien physique restera valide.
- Inconvénient : il faut rester dans le même volume physique (la même partition).
- On ne peut pas faire de lien physique avec un répertoire.
- syntaxe : **ln toto ../R2/titi**. Voir figure 8.
- Changer le contenu de **titi** revient à changer celui de **titi** et réciproquement.

Lien physique

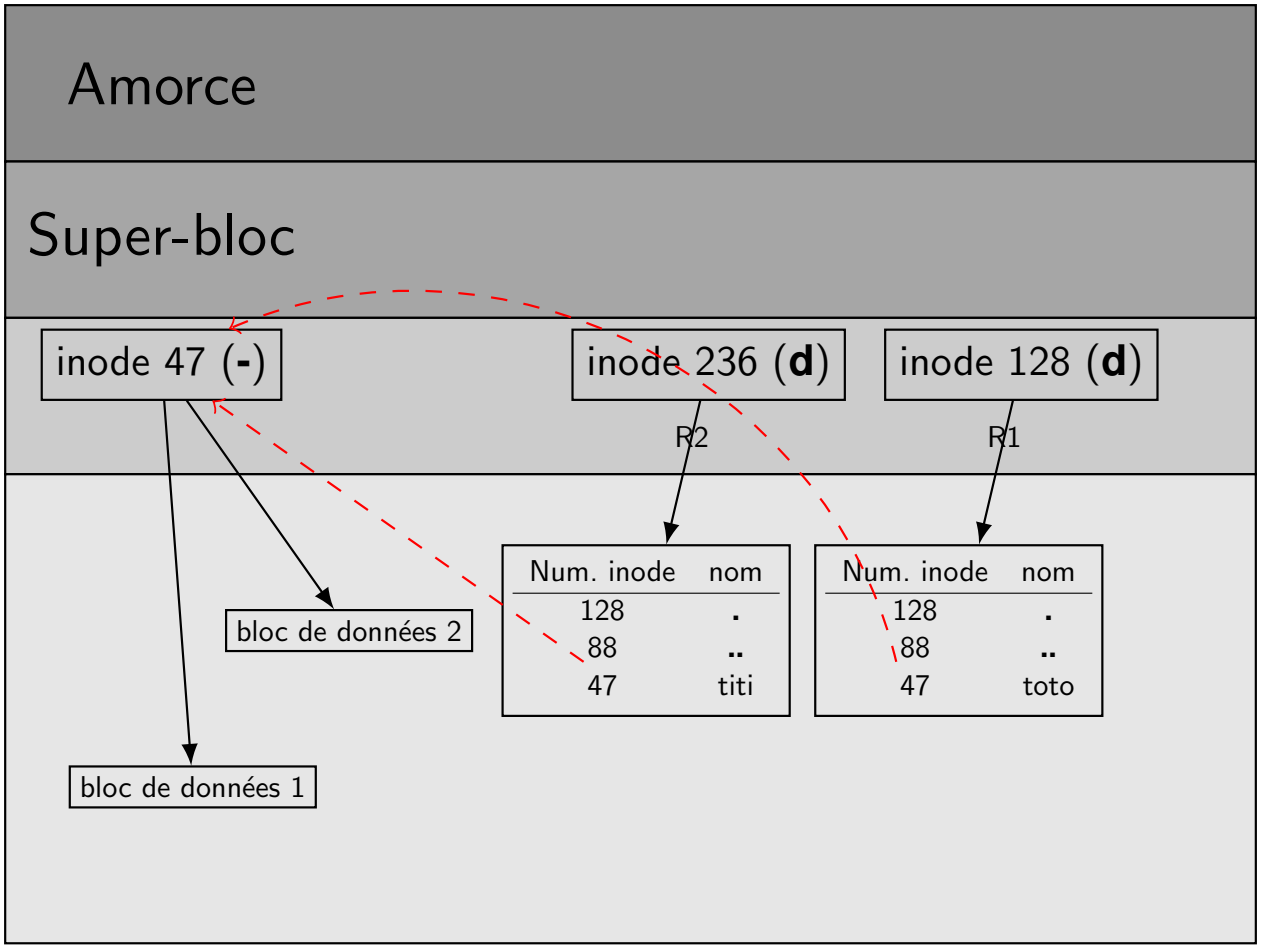


FIGURE – Lien **ln toto R2/titi** : **titi** et **toto** pointent vers le même inode

Suppression et inode

- En supprimant un fichier (**rm** pour remove), tout ce qui se passe est la suppression de l'un des noms pointant vers un numéro d'inode spécifique.

Suppression et inode

- En supprimant un fichier (**rm** pour remove), tout ce qui se passe est la suppression de l'un des noms pointant vers un numéro d'inode spécifique.
- Les données resteront jusqu'à la suppression de tous les noms associés au même numéro d'inode.

Suppression et inode

- En supprimant un fichier (**rm** pour remove), tout ce qui se passe est la suppression de l'un des noms pointant vers un numéro d'inode spécifique.
- Les données resteront jusqu'à la suppression de tous les noms associés au même numéro d'inode.
- Les systèmes Linux se mettent à jour sans nécessiter de redémarrage du système en grande partie à cause du fonctionnement des inodes.

Suppression et inode

- En supprimant un fichier (**rm** pour remove), tout ce qui se passe est la suppression de l'un des noms pointant vers un numéro d'inode spécifique.
- Les données resteront jusqu'à la suppression de tous les noms associés au même numéro d'inode.
- Les systèmes Linux se mettent à jour sans nécessiter de redémarrage du système en grande partie à cause du fonctionnement des inodes.
- Dans la situation de la figure 8, entrons **rm toto** : l'inode de **toto** n'est pas libéré puisque le lien de **titi** existe toujours. Voir figure 9.

Suppression et inode

- En supprimant un fichier (**rm** pour remove), tout ce qui se passe est la suppression de l'un des noms pointant vers un numéro d'inode spécifique.
- Les données resteront jusqu'à la suppression de tous les noms associés au même numéro d'inode.
- Les systèmes Linux se mettent à jour sans nécessiter de redémarrage du système en grande partie à cause du fonctionnement des inodes.
- Dans la situation de la figure 8, entrons **rm toto** : l'inode de **toto** n'est pas libéré puisque le lien de **titi** existe toujours. Voir figure 9.
- Entrons **rm titi**. Les blocs de données et l'inode correspondant sont libérés : le système pourra les attribuer à d'autres fichiers. Voir figure 10.

Suppression et inode

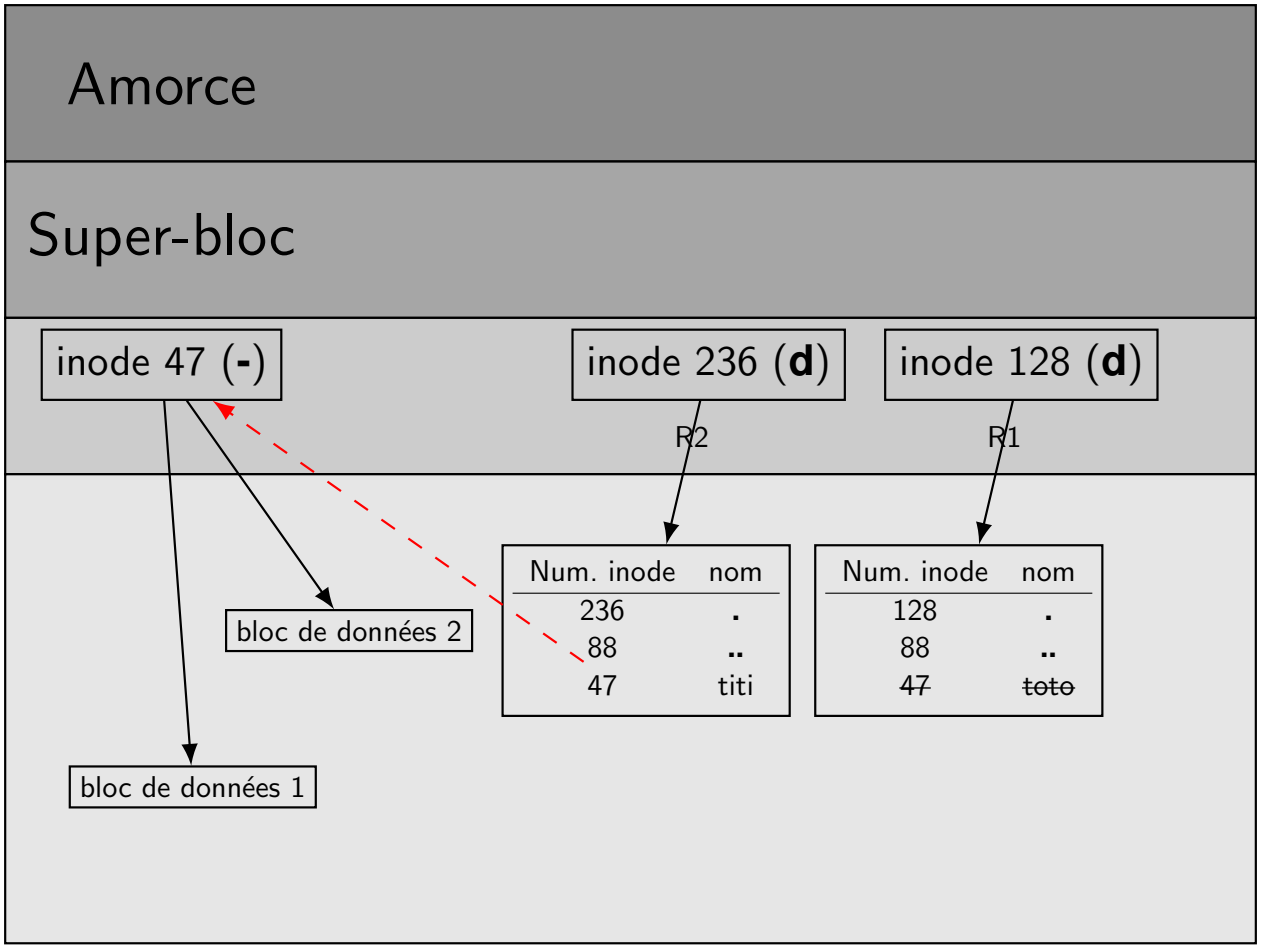


FIGURE – Suppression **rm toto** : l'inode 47 et ses données ne sont pas libérées

Suppression et inode

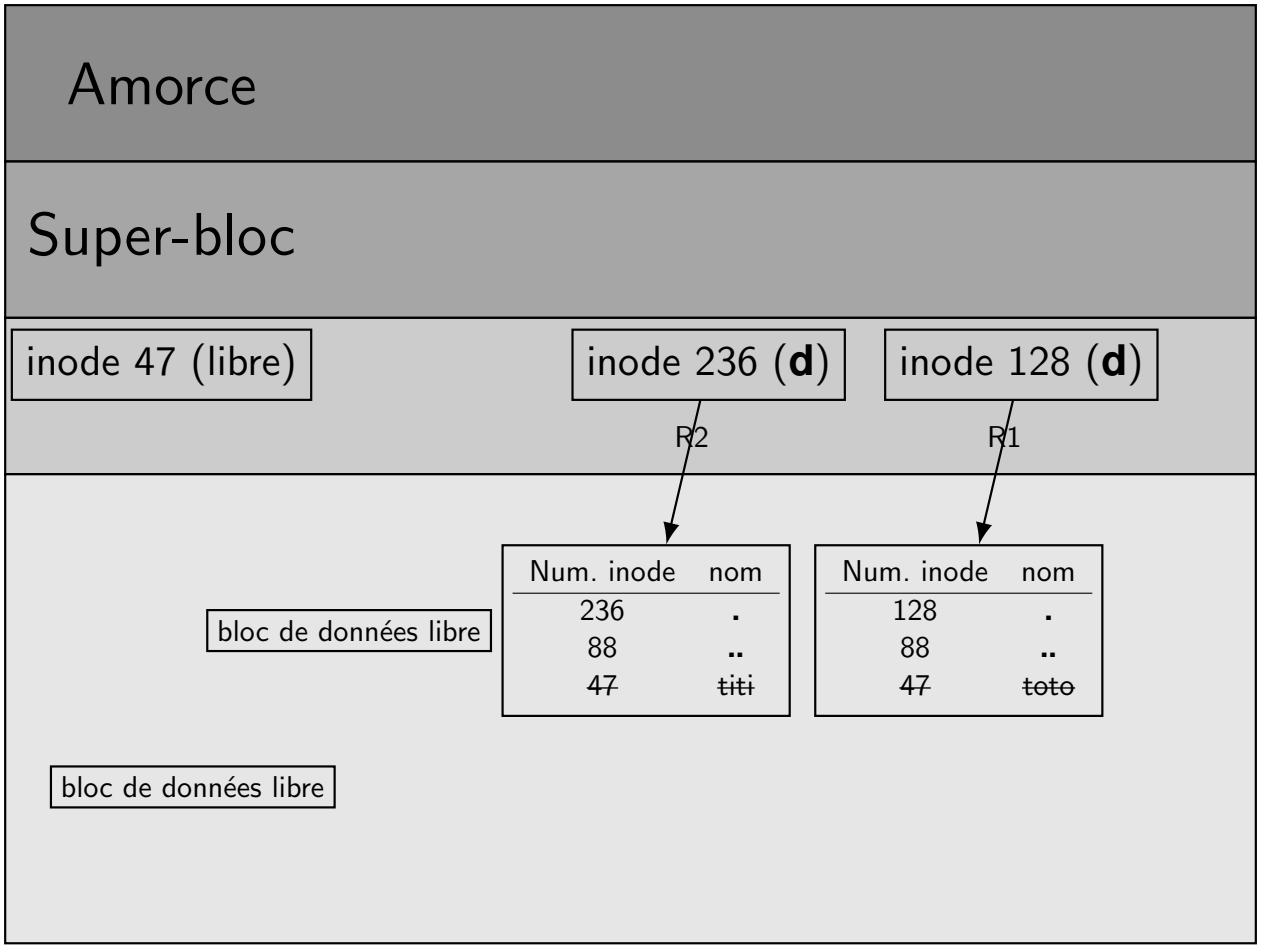


FIGURE – Suppression **rm tii** : l'inode 47 et ses données sont libérés

Lien symbolique

- Un *lien symbolique* est une référence à un nom (*i.e.* un chemin absolu ou relatif). Tandis qu'un lien physique réfère plutôt un inode.
- Avec un lien symbolique on peut traverser les partitions plus de risque que deux inodes différents aient le même numéro).
- Création avec l'option **s** de **ln** : **ln -s /path/to/original symlink**. Par exemple **ln -s R1/toto titi** crée un lien symbolique entre **titi** du répertoire courant et **toto** du répertoire **R1**.
- Par défaut, le lien symbolique doit être créé dans le répertoire courant. Si on veut spécifier aussi un chemin relatif pour le lien symbolique, utiliser l'option **r** : **ln -s R1/toto R2/titi**. Voir figure 11.

Lien symbolique