Généralités

- rôles d'un système d'exploitation (OS ou opérating System)
  - machine virtuelle qui cache la machine physique
  - gestion des ressources matérielles (mémoire, périphériques, ...)
- Unix
  - multi tâches et multi utilisateurs
  - axiome: « tout est fichier »
  - interface standard: POSIX
  - unix: plusieurs famille d'OS : BSD, SYS V, Linux, ...
    - cf http://fr.wikipedia.org/wiki/Tableau_synoptique_des_syst%C3%A8mes_d'exploitation

Rôle d'un administrateur système

- ajouts et suppressions d'utilisateurs
- ajout, suppression, configuration de matériel
- sauvegarde et restauration
- installation, mise à jour de logiciels
- surveillance du système:
  - sécurité
  - monitoring
- documentation locale
- rédaction de fiches de procédures, cahier des charges, ...
- aide aux utilisateurs

Principes de base de l'administration

- tout système doit avoir un administrateur
- complexité :
  - de plus en plus de machines
  - des systèmes hétérogènes
- méthodes de travail
  - rigueur (doc, gestion de version, validation, ...)
  - automatiser les tâches qui peuvent l'être
  - documentation: fiches de procédure, cahier des charges, ...
  - pas de travail de fond: le vendredi soir, après un pot :-)

Droits d'accès aux fichiers et aux processus

- fichiers: possède un propriétaire (UID) et un groupe propriétaire (GID)
- le propriétaire seul à pouvoir modifier les permissions du fichiers
- processus:
  - UID et GID réel: ceux de l'utilisateur du programme (et pas ceux du propriétaire du fichier sur disque)
  - UID et GID effectifs: pour déterminer les droits d'accès
- bits SETUID ou SETGIUD

Root

- root (superutilisateur): tout compte d'UID 0
- peut exécuter toute opération **valide** sur n'importe quel fichier ou processus
- devenir root
  - connexion directe en tant que root: déconseillé, notamment via réseau
  - su : changement d'identité
  - sudo: exécuter certaines commande en tant que root
- il est important de journaliser les actions effectuées en tant que root (sudo, snoopylogger, ...)

les autres pseudo-utilisateurs

- daemon : propriétaire des logiciels systèmes (uid 1)
- nobody: utilisateur sans droit
  - utilisé par nfs notamment
  - utilisé par certains daemon
  - ne doit possèder aucun fichier
- squid, bind, ...: pratique actuelle

méthodes d'administration

- installation de logiciels
  - des packages binaires via l'un des outils de gestion de packages du système d'exploitation :
  - des logiciels livrés en source à recompiler
  - des logiciels non livrés avec le système d'exploitation
- configuration:
  - via l'édition de fichier de configuration (=> il faut maîtriser un outil d'édition de texte par plateforme)
  - via des commandes d'administration
  - utilisation d'outils intégrés (smit (AIX), sam (HP UX), webmin, admintool (solaris)
  - via des scripts ou des extension d'outil (webmin) maison

7

Documentation

- man: les pages de manuel d'unix, sections du manuel
  - man commande
  - exemples :
    - man 1 kill: kill commande utilisateur (section 1 du man)
    - man 2 kill: kill appel système (section 2 du man)
    - apropos ou man -k
- documentation du système d'exploitation, /usr/doc, /usr/share/doc, ...
- WeB (google est votre ami notamment pour les messages d'erreurs)
- groupe de news USENET, forums WeB
  - fr.comp.os.* par exemple

8

le démarrage des PC

- MBR:
  - premier bloc de données du periph d'amorçage
  - 446 octets pour le gestionnaire d'amorçage
  - 64 octets pour la table des partitions (16 par partitions)
  - 2 octets à AA55 (valeur fixe)
- PBR: partition boot loader: en début de partition, chargé par le chargeur d'amorçage du MBR ou un autre chargeur d'amorçage
- MBR microsoft : charge le pbr de la première partition principale active
- gestionnaire d'amorçage: lilo, grub & Co

9

démarrage du système

- chargement et initialisation du noyau
- détection des périphériques
- création des processus systèmes, montage de la partition racine /en lecture seule, création du processus init
- intervention de l'opérateur (en cas de démarrage en mode monoutilisateur)
- exécution des scripts de démarrage
- passage en mode multiutilisateur

10

init

- cf man telinit
- Rôle de init
  - exécution ds scripts d'initialisation
  - gestion des terminaux
  - ancêtre de tous les processus
- 2 types de démarrage :
  - BSD
  - SYSV

11

mode monoutilisateur

- init lance un processus qui lance un shell root
- le processus de démarrage reprend après
- ce modecorrespond aux runlevel 1 ou S (le sens dépend de la distribution de l'OS)
  - S: réparation des systèmes de fichiers ou ~: pas de daemon, racine en lecture seule est la seule partition montée
    - il faut la remonter en lecture/écriture et monter les autres partitions si nécessaire
  - 1: administration, certains daemon sont actifs, les partitions sont toutes montées (Sys V, Suse, ...ràf: à préciser sous debian)

12

états du système

- BSD: monoutilisateur ou multiutilisateur
- Sys V, Linux:
  - plusieurs états (Run Level)
  - sens (dépend du système, voire de la distribution) :
    - 0: arrêt de la machine (halt)
    - 1: monoutilisateur (administration)
    - 2 à 5 : multiutilisateur
    - 6: redémarrage (reboot)
    - s ou S: monoutilisateur (maintenance)
  - le « run level » par défaut est défini dans /etc/inittab
  - connaître son « Run level »: who -r, runlevel (linux)
  - changer de « run level »: telinit No

13

scripts de démarrage BSD

- varient selon le système BSD
- l'idée de base: deux scripts (/etc/rc et /etc/rc.local) pilotent le démarrages du sytème
- Scripts de démarrage FreeBSD et autres BSD libres (ràf: dans une version ultérieure de ce document)

14

scripts de démarrage Sys V et Linux

- le système le plus répandu
- /etc/inittab:
  - indique à init ce qu'il faut faire pour chaque niveau d'exécution
  - au démarrage: init passe par tous les niveaux entre 0 et le niveau d'exécution par défaut;
  - idem dans l'autre sens à l'arrêt
  - les scripts de démarrages situées dans /etc/rcN.d sont démarrés via inittab
  - en pratique: de nos jours, on ne modifie que rarement inittab, on s'appuie sur les scripts (cf ci-dessus)

15

inittab

- Demo:
  - syntaxe du fichier
  - man inittab

16

Scripts de démarrage Sys V (suite)

- /etc/init.d ou /sbin/init.d (HP-UX):
  - contient la copie d'origine des scripts de démarrage
  - un script par daemon ou aspect du système
  - les scripts comprennent les arguments start et stop (et parfois d'autres arguments comme restart)
- /etc/rcN.d (N=0, 1, 2, ... : run level)
  - contient un lien symbolique vers le scripts d'origine. Le nom du lien peut avoir deux formes:
    - KXX: le service doit être arreté dans ce run level
    - SXX: le service doit être démarré dans ce « run level »
  - outils de gestion des scripts de démarrage : update-rc.d (linux debian)

17

Scripts de démarrage Sys V (suite)

- processus d'appel des scripts par init
  - recherche de /etc/rcN.d
  - si init augmente son niveau d'exécution: execution des scripts commençant par S dans l'ordre des No
  - si init diminue son niveau d'exécution: execution des scripts commençant par K dans l'ordre des No
- les scripts peuvent être utilisés pour arrêter ou redémarrer manuelle un service
  - exemple: /etc/inid.d/networking restart
- DEMO: exemple d'ajout de script avec update-rc.d

18

Scripts de démarrage: la relève

- Scripts SYS-V : supposent un système complètement opérationnel au boot : difficilement compatible avec les périphériques amovibles
- Gestion minimale des dépendances entre actions :
  - exemple: le montage d'une entrée de /etc/fstab peut nécessiter un outil présent dans /usr qui peut être monté via réseau. Il faut donc attendre que le réseau soit opérationnel
- la configuration des processus à lancer est dispersée :
  - dans /etc/init.d & Co, dans la crontab, dans

Scripts de démarrage: initng

- initing : « init new generation »
- permet le lancement asynchrone des scripts
- gestion des dépendances entre scripts
  - un script est lancé quand tous les scripts dont il dépend ont été lancés
- Exemple :

```
service system/my_service {
  need = system/initial net/all;

  exec start = /sbin/my_service --start –option;
  exec stop = /sbin/my_service --stop --option;
}
```

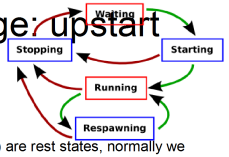

Scripts de démarrage: Sun Solaris Service Management Facility (SMF)

- dans une version ultérieure de ce document
- cf http://www.sun.com/bigadmin/content/selfheal/

Scripts de démarrage: upstart

- Un outil unique qui s'appuie sur des événements : les scripts sont lancés lors de certaines évènements
  - le système a démarré;
  - un système de fichier (SGF) a été monté
  - le SGF racine est en lecture/écriture
  - un périphérique block a été ajouté au système
  - il est une certaine heure (remplace cron)
  - un processus vient d'être lancé ou s'est arrêté
  - un fichier a été modifié
  - un périphérique réseau a été détecté
  - une connexion réseau a lieu (remplace inetd)
  - ...

Scripts de démarrage: upstart



- The two states shown in red (“waiting” and “running”) are rest states, normally we expect the job to remain in these states until an event comes in, at which point we need to take actual to get the job into the next state.
- The other states are temporary states; these allow a job to run shell script to prepare for the job itself to be run (“starting”) and clean up afterwards (“stopping”). For services that should be respawned if they terminate before an event that stops them is received, they may run shell script before the process is started again (“respawning”).
- Jobs leave a state because the process associated with them terminates (or gets killed) and move to the next appropriate state, **following the green arrow if the job is to be started or the red arrow if it is to be stopped**. When a script returns a non-zero exit status, or is killed, the job will always be stoped. When the main process terminates and the job should not be respawned, the job will also always be stopped.
- As already covered, events generated by the init daemon or received from other processes cause jobs to be started or stopped; also manual requests to start or stop a job may be received.
- The communication between the init daemon and other processes is bi-directional, so the status of jobs may be queries and even changes of state to all jobs be

Scripts de démarrage: upstart

- Exemple (version simplifiée du script de démarrage des scripts d'init SYSV de niv. 2 sur ubuntu 6.10)

```
start on startup
stop on shutdown
stop on runlevel-3

script
  set $(runlevel --set 2 || true)
  exec /etc/init.d/rc 2
end script
```

Scripts de démarrage: lauchd

- lauchd est fourni avec MacOS X
- similaire à upstart
  - des scripts sont lancés quand certains évènements se produisent
- différence avec upstart: les évènements possibles sont très limités:
  - démarrage du système
  - fichier modifié ou placé dans une file d'attente
  - date/heure atteinte (remplace cron)
  - connexion sur un port donné (remplace inetd)

25

Scripts de démarrage: la relève

- upstart (ubuntu) :
  - http://www.netsplit.com/blog/articles/2006/08/26/upstart-in-universe
  - http://upstart.ubuntu.com/
- launchd (MacOS-X) : http://developer.apple.com/macosx/launchd.html
- Sun Service Management Facility : http://www.sun.com/bigadmin/content/selfheal/
- synthese: http://www-128.ibm.com/developerworks/library/l-boot-faster/index.html?ca=dgr-lnxw0

26

arrêt du système

- init est à la source de tout
  - arrêt du système: demander à init de tuer proprement ses processus fils
  - shutdown [-r|-h] heure message : pour arrêter ou rebooter le système à l'heure indiqué
  - reboot: équivaut à shutdown -r ou à telinit 6
  - halt: équivaut à shutdown -h ou à telinit 0

27

processus d'arrêt du système

- écriture de l'évènement dans les journaux
- exécute les scripts KXX (init SYSV)
- init tue ses processus fils
- vide les tampons disque (sync)
- démonte les systèmes de fichiers quand les écritures sont terminées
- arrête la machine

28

cycle de vie d'un processus

29

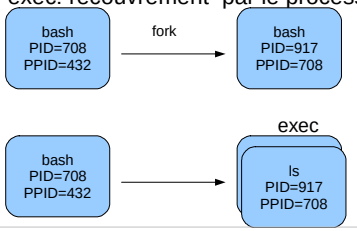
processus

- un programme: un fichier sur disque
- un processus: un programme en cours d'exécution
  - le code exécutable du programme
  - les données de l'instance en train de s'exécuter
- programme réentrant:
  - deux instances du même programme partagent le même code exécutable
  - elles ont par contre chacune leurs données
- processus système (daemon)/utilisateur

30

hiérarchie de processus, recouvrement

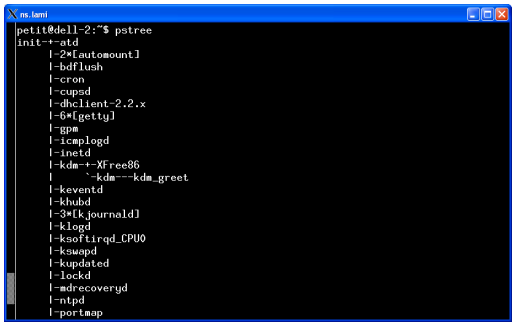
- un processus (processus fils) est toujours créé par un autre processus (processus père):
  - fork: création d'une copie du processus père
  - exec: recouvrement par le processus fils



Hiérarchie de processus

- tout processus a un processus parent sauf le processus initial
- processus initial : init (pid 1)
- arrêter la machine: demander à init d'arrêter tous ses processus fils

pstree



caractéristiques des processus

- statiques
  - PID
  - PPID
  - propriétaire réel (UID, GID)
  - terminal d'attache pour les entreés-sorties
- dynamique
  - propriétaire effectif (EUID, EGID)
  - priorité
  - nice
  - consommation cpu/mémoire
  - dossier de travail

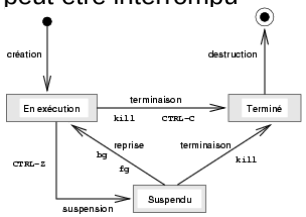
commande ps

- 2 syntaxes (Linux):
  - syntaxe Systeme V: option précédées de -
  - syntaxe BSD: options NON précédées de -
  - quelques options SysV:
    - -e ou -A: tous les processus
    - -a: tous les processus associés à un terminal
    - -H: représentation hiérarchique (forêt)
    - -f: format complet;-l: format long (encore plus détaillé)
    - -o: pour modifier le format de sortie (cf manuel)
    - -g, -p, -t, -u: n'affiche que les processus des groupe (-g), processus (-p), terminaux (-t) ou utilisateurs (-u) listés.

commande ps: exemple

Etat d'un processus

- R: exécution
- Z: zombi: il est mort mais son père ne le sait pas
- D: le processus ne peut être interrompu
- S: suspendu
- T: terminé



gestion de processus & shell

- commande & : lancement de commande en tâche de fond
- bg: reprise de commande en tâche de fond
- fg: reprise de commande en avant plan
- jobs: liste des commandes lancées
- Ctrl-C: arrêt (SIGTERM) du processus
- Ctrl-Z: suspension (SIGSTOP) du processus

Signaux

- permettent au système de communiquer avec les processus
- signaux utiles
  - STOP: suspendre
  - CONT: reprendre
  - HUP (1): souvent: relecture configuration
  - KILL(9): tuer sans possibilité de traitement
  - INT(2): équivalent à Ctrl-C: interruption gérable. permet au processus de gérer son interruption
- kill -signal PID

priorité des processus

- l'exécution des divers processus est gérée par un ordonnanceur (scheduler)
- une priorité est définie dynamiquement
- but: que chaque processus puisse avancer son exécution tout en respectant des priorités
- nice/renice: permet d'influer sur la priorité des processus
  - de 0 à 19 pour un utilisateur
  - de -20 à 19 pour root
  - plus le chiffre est élevé, moins le processus est prioritaire

code de retour

- valeur à laquelle le processus père peut accèder
- 0: terminaison normale
- autre valeur: situation anormale

systèmes de fichiers

- différents systèmes de fichiers:
  - exemples
  - journalisation
  - création d'un système de fichier
  - exemple sous linux
- partition
  - associée à un système de fichier et un point de montage
  - de swap
- gestionnaire de volume logique
  - développer LVM sous Linux

acl POSIX

- dans une version ultérieure de ce document
  - Cf TD
- cas particuliers:
  - linux debian
  - FreeBSD

43

Systèmes de gestion de fichiers (SGF)

- SGF: mode d'organisation et de stockage des données sur disque;
- Exemples: FAT32, NTFS, ext2fs, ext3fs, reiserfs, UFS, ...
- Les SGF ont des propriétés et fournissent des services variés
- Exemple:
  - les SGF Unix (ext2fs, UFS, ...) : droits sur les fichiers.
  - FAT32: pas de droits d'accès aux fichiers

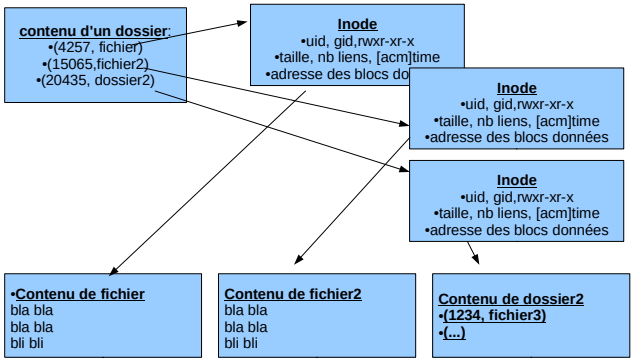
44

SGF (suite)

- Les SGF unix fournissent un sous-ensemble commun de fonctionnalités: celui dont nous parlerons.
- Chaque SGF peut fournir plus que ce sous-ensemble
- Fichier unix: fichier disque mais aussi ressource du système (pilote de périphérique, terminal, mémoire, ...)
  - /dev/hda1 : partition 1 du disque 1 (Linux)
  - /dev/kmem: mémoire virtuelle du noyau

45

Fichiers



SGF : inode

- Inode: attributs + localisation des blocs contenant les données du fichier
- Inode:
  - Id. du disque logique où est le fichier,
  - numéro du fichier sur ce disque
  - Type de fichier et droits d'accès
  - Nombre de liens physiques
  - Propriétaire, groupe propriétaire
  - Taille
  - Dates :
    - De dernier accès (y compris en lecture): atime
    - Date de dernière modification des données: mtime
    - Date de dernier modification de l'inode: ctime

47

Dossier/répertoire

- Deux grandes classes de fichiers :
  - Fichier ayant un contenu sur disque : fichiers réguliers, dossiers, liens symboliques, tubes
  - Ressources : Fichiers spéciaux (pilotes de périphériques, ràf, ...)
- Dossiers: listes de couples (nom, No inode)
- Un couple est appelé « lien physique » (hardlink)
- Du point de vue de l'utilisateur, un dossier contient des fichiers (fichiers réguliers, dossiers, ...)

48

Inodes/Nom: conséquences

- Créer/détruire un fichier: ajouter/retirer un couple dans le dossier
- opération nécessitant un droit au niveau du dossier pas du fichier
- Le système travaille avec des No d'inode, l'utilisateur avec les noms de fichiers :
  - les dossiers font le lien entre les deux :
    - On trouve le couple (nom, inode) du dossier où est le fichier
    - Pour trouver ce dossier, on applique le même principe (pour Unix, un dossier est aussi un fichier)

Fichiers: résumé:

- ce que l'utilisateur perçoit comme un fichier identifié par un nom peut se décomposer en trois notions sous unix :
  - un inode: informations (taille, dates, uid, gid, droits) et localisation des données sur disque
  - le contenu du fichier: les données qui y sont stockées
  - un lien physique: associe un nom à un inode. Un même inode peut avoir plusieurs lien.

Droits d'accès aux fichiers

- 3 types d'accès: lecture (R), écriture (W) et exécution (X)

Objet/Droit	R (lecture)	W (écriture)	X (exécuter)
fichier régulier	lire le contenu	modifier le fichier	exécuter le fichier
dossier	lister le contenu du dossier	modifier le contenu du dossier (y compris destruction de fichier)	utiliser le dossier dans un chemin ou s'y positionnner

- 3 classes d'utilisateurs: le propriétaire du fichier, le Groupe du propriétaire du fichier, les Autres utilisateurs.

type fichier	Propriétaire			Groupe du proprio			Autres utilisateurs		
-	R	W	X	R	-	X	R	-	X

- informations dans l'inode, affichage avec « ls », changement avec chmod, chgrp et chown

Droits d'accès : algorithme

- Si l'uid réel du processus est égal à l'uid du propriétaire du fichier: ce sont les droits du propriétaire qui détermine l'accès (y compris les refus d'accès)
- Sinon si le gid du processus est égal au gid du propriétaire du fichier: ce sont ces les droits du groupe propriétaires qui déterminent l'accès (y compris les refus d'accès)
- sinon, c'est l'entrée other qui est utilisée

Droits d'accès : algorithme

- Exemple (tordu) : soit un fichier f dont les permissions sont : R— RWX RWX et un utilisateur qui en est propriétaire et qui appartient au groupe propriétaire du fichier
  - L'utilisateur n'a que le droit de lecture alors qu'il appartient au groupe
  - Sous unix, les permissions ne sont pas cumulatives.
  - En résumé, on peut dire que sous unix, le particulier (utilisateur) l'emporte sur le général (groupe)

ACL posix: algorithme

- Si l'uid réel du processus est égal à l'uid du propriétaire du fichier ou à l'IUD d'une entrée utilisateur des ACL :ce sont les droits du propriétaire qui détermine l'accès (y compris les refus d'accès)
- Sinon si le gid du processus est égal au gid du propriétaire du fichier ou à l'IUD d'une entrée groupe des ACL : ce sont ces les droits du groupe propriétaires qui déterminent l'accès (y compris les refus d'accès)
- sinon, c'est l'entrée other qui est utilisée

ACL posix: conseils

- Les ACL POSIX vérifient le principe selon lequel le particulier l'emporte sur le général
- Utiliser des ACL utilisateur et des ACL groupe limite la lisibilité des ACL :
  - Pour savoir si un utilisateur a un droit, il faut vérifier si ce droit ne lui est pas refusé dans une ACL utilisateur et s'il existe une ACL utilisateur ou groupe ou other le lui donnant
- Conseil : n'utiliser que des ACL groupe (quitte à créer un groupe pour un seul utilisateur si nécessaire)

55

Droits d'accès (2): suid, sgid, sticky bit

- 3 autres « droits » spéciaux:
  - bit SUID: le programme s'exécute avec les droits de son propriétaire (au lieu de ceux de l'utilisateur qui le lance)
  - bit SGID: le programme s'exécute avec les droits du groupe propriétaires du fichier
  - sticky bit :
    - sur un fichier exécutable : (obsolète) maintient le fichier en mémoire après l'exécution pour diminuer le temps de la prochaine invocation
    - sur un dossier: seul le propriétaire du fichier a le droit de le supprimer. Exemple: /tmp/

56

Commandes de base: chmod

- chmod [-R] mode fichier ...
- -R: fichier est un dossier, chmod agit récursivement sur fichier et sur son contenu
- mode:
  - forme numérique: 644
    - pour u: 400 (r), 200 (w) et 100 (x)
    - pour g: 40 (r), 20 (w) et 10 (x)
    - pour o: 4 (r), 2 (w) et 1 (x)
  - forme symbolique: [ugo][+-=][rwxXstguo]

57

chmod: exemples

58

commande de base: umask

- définit les droits d'accès par défaut d'un fichier
- les droits sont le complément du paramètre d'umask: on laisse tout sauf les droits précisés
- Exemple:
  - umask 002 : mode par défaut: RWXRWXR-X (tout sauf 002)
  - umask 026: mode par défaut: RWXR-X--X (tout sauf 026)
  - umask a=rx,gu+w: mode par défaut: RWXRWXR-X
  - umask -S : affiche le l'état courant sous forme symbolique : u=rwx,g=rwx,o=rw dans notre exemple.

59

Commandes de base: chown, chgrp

- chown -R [-H | -L | -P] proprio[:groupe] fichier
- chgrp -R [-H | -L | -P] groupe fichier ...

60

chown/chgrp: exemples

61

Commandes de base: ls

62

Commandes de base: cat

63

Commande de base: stat

64

Exemples

- Stat fichier (noter ctime, mtime et atime)
- Cat fichier
- Stat fichier (atime a changé)
- Chmod fichier
- Stat fichier (ctime a changé)
- Modif fichier
- Stat fichier (mtime a changé)

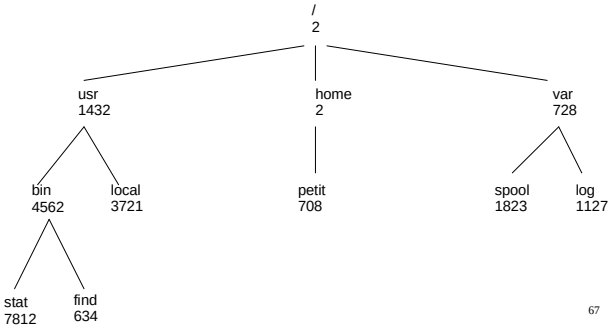
65

Arborescence

- Sous unix, on a une arborescence unique (donc pas de C:\, D:\, ...comme sous windows)
- Le disque système contient la racine absolue /
- toute l'arborescence est sous cette racine absolue
- Les systèmes de fichiers des autres partitions s'intègrent dans l'arborescence en prenant la place d'un dossier existant
- la racine d'un système de fichier a 2 comme numéro d'inode

66

arborescence



monter un système de fichier

- commande mount/umount
- /etc/fstab
  - des associations système de fichier/point de montage
  - notamment: les partitions à monter au démarrage du système
  - l'ordre des lignes est important pour mount, umount et fsck
  - syntaxe: periph pointDeMontage typeSGF options fsfreq fspassNo
  - demo: exemple de fstab
- fsck, df, du

algo de recherche

- /usr/bin/stat
- algo de localication:
  - examiner le contenu du dossier d'inode 2 pour trouver le No d'inode du dossier usr : 1432 par exemple.
  - examiner le contenu de dossier d'inode 1432 pour trouver le No d'inode du dossier bin. 4562 par exemple
  - examiner le contenu de dossier d'inode 4562 pour trouver le No d'inode du fichier stat. 7812 par exemple
  - exécuter le fichier d'inode 7812

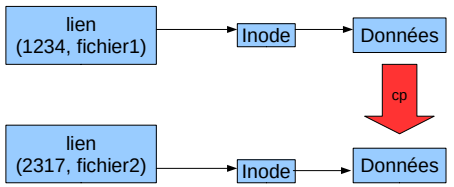
Chemin

- /usr/bin/stat: chemin absolu du fichier stat
- chemin absolu: chemin depuis la racine absolue
- notion de dossier courant
- chemin relatif: chemin depuis le dossier courant

Commandes de base:

- pwd : indique le dossier courant
- cd : changer de dossier courant
- mkdir: pour créer un dossier
- rmdir: détruit les dossiers vides

commande de base: cp



cp fichier1 fichier2

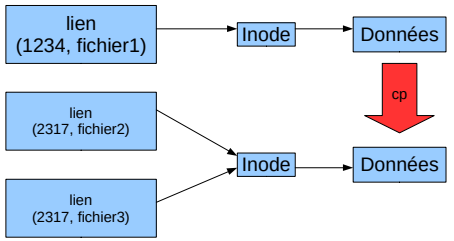
Commandes de base: cp

- copie des données d'un fichier (source) dans un autre (cible) :
  - la cible n'existe pas : création d'un nouvel inode et recopie des données du fichier
  - la cible existe: inode destination inchangée, recopie des données du fichier dans la cible

cp (2)

- gnu cp: en standard sous Linux, installable facilement ailleurs
- fournit des options non standard mais pratiques
- ràf

cp et liens physiques



cp fichier1 fichier2

Commandes de base: rm

- suppression d'un lien d'un fichier ou plusieurs fichiers: `rm [-fiRr] fichier1 ...`
- options:
  - -i: demande de confirmation pour tout fichier à supprimer (aff sur stderr et lecture sur stdin)
  - -f: supprime les messages d'erreur lorsqu'un fichier n'existe pas et supprime la demande d'acquittement si l'utilisateur de rm n'a pas les droits d'écriture sur le fichier à supprimer
  - -R ou -r: supprime récursivement le contenu d'un dossier avant d'appliquer rmdir au dossier.

•rm :exemples

Commandes de base: mv

- mv [-fi] source destination
  - renomme un lien. Source et fichier sont des fichiers réguliers
- mv [-fi] source1 ... destination
  - renomme les sources en les déplaçant **dans** le dossier destination
- la commande fonctionne aussi si sources et destinations sont dans des systèmes de fichiers différents.
- la seconde forme est utilisée si la destination est un dossier existant

mv: exemples

- mv [-fi] source destination
  - renomme un lien. Source et fichier sont des fichiers réguliers
- mv [-fi] source1 ... destination
  - renomme les sources en les déplaçant **dans** le dossier destination
- la commande fonctionne aussi si sources et destinations sont dans des systèmes de fichiers différents.
- la seconde forme est utilisée si la destination est un dossier existant

Commandes de base: ln

- ln fichier nouveau_lien_physique
- ln -s fichier lien_symbolique
- options:
  - -s: crée un lien symbolique
  - -f: force la création même si la destination existe déjà
  - --: fin des options (pour permettre le traitement d'un fichier dont le nom commence par « « - »

Exemples

Sauvegarde versus Archivage: sauvegarde

- sauvegarde: c'est dupliquer et externaliser les données pour se protéger :
  - des erreurs humaines
  - des crash matériel
  - des erreurs logicielles
  - des sinistres (incendie, inondation, ...)
  - de la malveillance (vol, virus, ...)
- la durée de vie d'un jeu de sauvegarde est limitée dans le temps (quelques mois au plus)
- Sauvegarde des données / du système

Sauvegarde versus Archivage: sauvegarde

- on sauvegarde certaines données pour se protéger de certains risques
  - faire la liste des risques dont il faut se protéger
  - adapter le cahier des charges des sauvegardes et des autres mesures en fonction des ces risques
  - les sauvegardes ne sont qu'un des éléments permettant de garantir la continuité du service à côté d'autres : disques raid, redondance des serveurs, ...

Sauvegarde versus Archivage: archivage

- l'archivage, c'est le stockage hors ligne de données peu utilisées
  - le temps d'accès a souvent peu d'importance
  - on peut avoir une hiérarchie de mode de stockage (du plus rapide au moins rapide (bande à aller chercher) suivant la fréquence de l'utilisation des données concernées)
- la durée de vie d'une archive se compte au minimum en années, voire en décennies
- il faut donc penser à l'obsolescence des matériels, des supports, des logiciels, des standards utilisés

Solutions qui ont prouvé leur inefficacité

- Faire faire les sauvegardes par les utilisateurs : pour dégager sa responsabilité mais aucune garantie qu'elles seront faites
- Sauvegardes sur des supports peu fiables (disquettes, DAT, ...)
- Sauvegarde en écriture seule : il faut valider sauvegarde et procédures de restauration
- Sauvegarde d'un système en cours d'exécution
- Un seul support de sauvegarde
- Sauver sur une partition du même disque
- Pas de sauvegarde hors site (incendie, ...)

85

Sauvegardes : procédure/planification

- Planifier les sauvegardes, tester leur réalisation
  - il faut avoir l'assurance que les sauvegarde prévues ont eu lieu
  - les procédures de sauvegardes doit être écrites
  - procédures testées
  - procédures et planification validées par les utilisateurs/propriétaires des données

86

Sauvegardes : planification

- choix des données à sauvegarder :
  - données des utilisateurs (y compris : boîtes aux lettres, profil, ...), fichiers de configuration, ..;
    - retrouver un système en état suppose de réinstaller le système puis de restaurer les données
    - volumétrie plus faible
  - Système entier avec procédure de redémarrage:
    - restauration directe et rapide d'un système opérationnel
    - volumétrie plus importante
- périodicité, choix des données ont un impact fort sur la volumétrie et donc sur le coût =>compromis

87

Problèmes liés à la volumétrie:

- Coût
- Charge réseau
- Durée des sauvegardes
- Indisponibilité des serveurs/applications dans le cas où le logiciel de sauvegarde impose l'arrêt des logiciels.

88

Restauration

- procédures de restauration
  - écrites
  - au résultat validé par les propriétaires des données (pour ne pas en oublier)
  - **testées régulièrement en grandeur réelle** de façon à garantir :
    - que toutes les données pertinentes ont été sauvegardées
    - d'être capable de tout restaurer correctement

89

Fiablisation

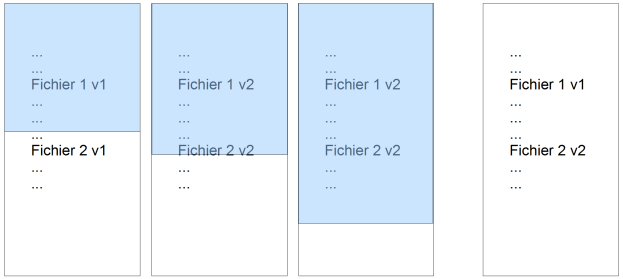
- utiliser des média fiables
  - éviter disquettes, CD RW, DAT, ..et leur préférer CD R (bof), DLT, AIT, ...
  - varier les marques de media pour palier un défaut de fabrication dans une série, ...)
- gérer le vieillissement des média de sauvegarde/archivage (reprise sur des média récents, ...)

90

Fiabilisation (2)

- fiabiliser l'environnement des media de sauvegarde:
  - inondation, incendie, vol (coffre ignifugé, ...)
  - sauvegarde hors site (penser à la confidentialité des données, au risque de vol, à l'interception des données, ...)
- indexer les données pour s'y retrouver dans le volume total des jeux de sauvegardes
  - indexer les données
  - étiqueter les media

Sauvegarde live: problématique



Si l'application modifie plusieurs fichiers durant la sauvegarde, les versions sauvées peuvent ne pas être cohérentes.

Sauvegarde

Sauvegarde live: solutions

- Arrêt de l'application pendant la sauvegarde:
  - garantit de plus qu'aucun verrou n'empêchera l'accès à un fichier
- Instantané (Snapshot): image des blocs du sgf, en cas d'effacement de fichier, les blocs ne sont pas réalloués tant que la sauvegarde n'est pas finie
  - Suppose un support dans le sgf
  - Proposé par les NAS, par afs, LVM, UFS2 (FreeBSD, ...), ...
  - Application supplémentaire : proposer aux utilisateurs une image des états antérieurs de leur compte à moindre coût.

Sauvegarde live: snapshot

- Là, on mettra une illustration du fonctionnement des snapshot
- Fait en live au tableau.

Sauvegardes incrémentales/différentielles

- Sauvegarde incrémentales : fichiers créés ou modifiés depuis la sauvegarde précédente
  - Diminue le volume à sauver
  - Restauration nécessite toutes les sauvegardes, restaure toutes les versions d'un même fichier
  - Peu adapté si la totalité des fichiers changent
- Sauvegarde différentielle : fichiers créés/modifiés depuis la dernière sauvegarde de référence
  - Diminue le volume à sauver
  - Plus de volume qu'en incrémental
  - Restauration nécessite la sauvegarde de référence et la dernière sauvegarde différentielle

Exemple:

- Comparer la taille des sauvegardes et les procédures de restauration dans les cas suivants : (ST: sauvegarde totale, I: sauvegarde incrémentale, D: sauvegarde Différentielle)
- Cas 1) ST, I1, I2, I3, I4, I5, I5
- Cas 2) ST, D1, D2, D3, D4, D5
- Cas 3) ST, I1, I2, I3, D1, I4, I5

Dump: un outil de sauvegarde sous unix

- Dump est un outil unix
- Il est efficace (travail directement au niveau du sgf)
- Sauvegarde non portable d'un unix à un autre (lié au sgf)
- Niveau (0 à 9) permettant un gestion très souples des sauvegarde incrémentales/differentielles:
  - Une sauvegarde niveau *n* sauvegarde tous les fichiers modifiés depuis la dernière sauvegarde de niveau *n-1*

Ntbackup: un outil de sauvegarde sous windows

- Un outil apparu avec windows 2000
- Pratique et polyvalent
- Supporte les sauvegarde incrémentales et différentielles
- Permet de sauver les données (au choix) et/ou les fichiers systèmes
- Utile pour des sauvegardes « artisanales »
- Pour des sauvegardes lourdes, la pratique est d'utiliser des logiciels dédiés (cf cours sauvegarde AF) qui géreront les verrous sur les fichiers, les sauvegardes live de certaines applications, ...

Ntbackup

- La restauration nécessite que w2k soit installé
- NT Backup est installé par défaut sous windows 2000/XP pro et serveur.
- Ntbackup est fourni avec windows Xp home (cf VALUEADD\MSFT\NTBACKUP)

Windows: droit requis pour les sauvegardes

- Dans la prochaine version de ce document

Syslog

- syslogd: daemon chargé de gérer les journaux d'une machine
  - journaux: /var/log/*.log (en général)
- peut gérer les journaux d'hôtes distants
  - option « -r » à positionner explicitement
  - rfc 3164: BSD Syslog protocol
  - udp port 514
  - supporté par de nombreux type d'équipement réseau: un standard incontournable

Syslog

- sécurité:
  - pas d'authentification, de filtrage des sources,
  - pas de chiffrement des informations
  - udp: non connecté, pas d'assurance de délivrance

Syslog

- gestion des journaux:
  - gaffe classique: un disque plein à cause de journaux accumulés
  - outils de gestion des journaux : logrotate, newsyslog, ...: compresser, déplacer, effacer, ...

Syslog

- analyse des journaux:
  - pour détecter un problème et/ou en déterminer les causes **après coup**
  - pour alerter d'un problème **en cours**
  - des rapport d'analyse de journaux trop long ne sont pas (plus) lus. Il faut :
    - réagir rapidement aux choses graves
    - extraire les informations pertinentes de la masse d'information
  - Deux types d'outils
    - outils d'analyse de journaux: logcheck, logsurfer, swatch, sec, ...
    - via un ids: système de détection d'intrusion

syslog-ng:

- configuration plus souple
- classement des messages par leur contenu, par l'hôte d'origine
- meilleure redirection des messages sur le réseau
- possibilité de chroot
- peut utiliser UDP et TCP
- chiffrement et authentification du trafic réseau
- portable
- export des journaux vers un sgbd

configuration: syslog.conf

- facilité.niveau<tab>action
- facilité: type de service source

Action
fichier
terminal
pipe |
@machineDistante
utilisateur1,utilisateur2,...
*

Niveau	
emerg (panic)	Situations de panique.
alert	Situations urgentes.
crit	Situations critiques.
err (error)	Erreurs.
warning (warn)	Messages de WARNING.
notice	Messages divers.
info	Messages d'informations.
debug	Débogage.

Facilités	
kern	Le noyau.
user	Process des utilisateurs.
mail	Système de courrier.
daemon	Démons systèmes.
auth	Authentification.
lpr	Système de spooling d'imprimante.
news	Usenet.
uucp	UUCP.
cron	Démon cron.
mark	Messages générés à intervals réguliers.
local0-7	Huit niveaux de messages locaux.
syslog	Messages internes à syslogd.
authpriv	Messages privés auth.
*	Toutes les facilités sauf mark.

Syslog : demo

- lister le syslog d'un système existant
- lister un journal de /var/log, montrer les entrées "MARK" insérées par syslogd
- tester son comportement avec la commande logger
  - logger -p mail.crit "boîte au lettre en feu :-) »
  - logger -p news.err "pas de nouvelles, bonne nouvelle"
  - comparer l'effet avec le contenu de syslog.conf et notamment que le message est stocké si son niveau est supérieur ou égal à celui de la règle
- le modifier en y insérant une entrée
- tester l'entrée insérée avec logger

Bibliographie sur la supervision et sur syslog

- « unix, guide de l'administrateur » de Nemeth, Snyder & Al, Campus press
- « MISC No 22 » (revue): superviser sa sécurité
- Ntsyslog: http://ntsyslog.sourceforge.net/
- http://www.linux-kheops.com/line/html/line/line-dec1996/datas/syslog.htm
-

comptes utilisateurs: création

- uid
- modifier /etc/passwd & Co
- mot de passe
- dossier personnel
- fichier d'initialisation dans $HOME
- donner les bons droit au dossier perso (chgrp, chown)
- déclarer l'utilisateur dans les services usuels (mail, ...)
- tester le compte

comptes utilisateurs

- structure d'un fichier /etc/passwd
- passwd: pour changer son mot de passe
- shadows passwords: /etc/shadow
- commande d'administration :
  - dépend du système d'exploitation
  - exemples:
    - useradd/adduser
    - userdel

groupes

- /etc/group
- chaque utilisateur a un groupe initial (/etc/passwd) et des groupes secondaires (/etc/group)
- groups: liste les groupes de l'utilisateur
- groupes sous BSD:
  - l'utilisateur appartient à tous les groupes
  - création de dossier/fichier: groupe du dossier père
  - gestion des groupes: pw (création/suppression, ajout d'utilisateurs, ...)

groupes

- groupe sous SysV et Linux
  - l'utilisateur appartient à un instant donné à un seul groupe => newgrp pour changer de groupe
  - création de dossier/fichier: groupe du dossier père ou groupe de l'utilisateur (Linux, autorisé par SysV)
  - gestion des groupes
    - groupadd, groupmod, groupdel: ajout/suppression de groupes
    - usermod -G group,... login: ajoute login au(x) groupe(s)

planification de tâches: cron et atd

- cron: tâches planifiées régulières
- atd: exécution unique
- cron et arrêt systèmes/chgt d'heures
- commande crontab:
  - crontab -l : lister
  - crontab -r : supprimer
  - crontab -e : modifier
- dossier daily, monthly, ...: (dépend de l'OS)

format du fichier crontab

- régles communes:
  - # en début de ligne indique un commentaire
  - les champs sont séparés par des espaces
  - les espaces de la commandes sont laissés inchangés. commande exécutée par sh
  - dans la commande, % indique un saut de ligne
  - contenu des champs :
    - *, entier, entier-entier, des entiers/intervalles séparés par des virgules
- crontab utilisateur :

```
minute heure jourDuMois jourDeLaSemaine commande
```

- crontab sytème (souvent : /etc/crontab)

```
minute heure jourDuMois jourDeLaSemaine utilisateur commande
```

crontab: exemples

- commandes valides :

```
echo date courante: `date` >> /tmp/test
mutt -s "coucou Pascal" petit@shayol.org % coucou
  % courrier de test
find / -xdev -name core -atime +7 -exec /bin/rm -
  f {} \;
```

- spec de temps valides:

```
*0 * * * * : toutes les 10 mn
10 2 * * * : tous les jours à 2h10
0 23 * * 0 : tous les dimanches à 23h00
0 20-23,0-7,10,12,14,16,18 * * * : toutes les
  heures entre 20h00 et 7h00 puis toutes les deux
  heures
```


cron : sécurité

- contrôle d'accès :
  - cron.allow: seuls utilisateurs habilités à programmer des tâches
  - cron.deny: seuls utilisateur NON autorisés à programmer des tâches (suppose l'absence de cron.allow)
  - si ni cron.(allow|deny): seul root y a droit
- contrôle d'accès réalisé par la commande crontab
  - => les fichiers crontab doivent avoir les bons droits

Chiffrement : définitions

services offerts par le chiffrement:

- confidentialité
- intégrité: chiffrer une empreinte du message
- signature numérique
- authentification (ex.: ssh qui authentifie les machines)
- kerberos: authentification centralisée unique
- non répudiation: prouver qui a créé un message: utilisation de tiers de confiance, chiffrement à clef publique

Chiffrement: robustesse

- cryptanalyse: analyser une information chiffrée pour la déchiffre (dont des méthodes en force brute, ...)
- algo public
- la sécurité repose sur :
  - la non divulgation de la clef
  - la robustesse de l'algorithme
  - la taille de la clef (gare aux comparaisons entre algo différents)
  - l'utilisation de clefs différentes pour chiffrer des messages différents limite la quantité d'information à la disposition de l'attaquant

chiffrement: taille des clefs

- attaques en force brute: tenter une partie importante de l'espace des clefs
- temps dépend du nombre de clefs possibles et donc de la taille de la clef:
  - 10 bits : 1024 clefs possibles
  - 56 bits: 2^56=7 10^16
  - dépendance exponentielle en fonction de la taille de la clef: 1 bit de plus = 2 fois plus de temps
- la taille critique dépend de l'algo (et de sa vitesse, de ses faiblesses, ...)

algorithme de chiffrement

- chiffrement symétrique/asymétrique
  - symétrique: les algo classiques sont rapides
    - la même clef sert au chiffrement et au déchiffrement
    - souvent utilisé via une clef de session
      - clef de session: transmise via algo asymétrique (on parle d'enveloppe digitale)
      - session: chiffrée par un algo symétrique et la clef transmise
  - asymétrique: les algo classiques sont lents
    - couple de clef publique/clef privée
      - clef publique: peut être connue de tous
      - clef privée: tenue cachées
      - ce qui est chiffré avec l'une ne peut être déchiffré qu'avec l'autre

- Algorithmes de chiffrement symétrique:
  - DES (1976): standard américain (1977), clef de 56 bits sur des blocs de 64 bits. dépassé de nos jours.
  - triple DES (1978): variante via une triple application de DES permettant d'avoir des clefs entre 128 et 192 bits sur des blocs de 64 bits.
  - RC2, RC4, RC5 (1994) et RC6:
  - IDEA (1992): clef 128 bits sur des blocs de 64 bits
  - blowfish: clef 32 à 448 bits sur des blocs de 64 bits. Algo très analysé,considéré comme solide. utilisation libre.
  - AES (1998): clefs 128, 192 ou 256 bits sur blocs de 128 bits. standard américain. utilisation libre.

algorithmes classiques

- asymétriques:
  - RSA s'appuyant sur la factorisation de nombres premiers
  - Diffie-Hellman et El Gamal s'appuyant sur le calcul des logarithmiques discrets
  - des algorithmes nouveaux s'appuyant sur les courbes elliptiques

durée de vie des clefs

- dépend de sa taille
- dépend de son taux d'utilisation
- dépend du contexte d'utilisation
- hiérarchie de clef (clef maîtresse, clef de session par ex.)
- révocation de clef
- une utilisation intensive du chiffrement nécessite la mise en place d'une IGC (infrastructure de gestion de clef ou PKI – Public Key Infrastructure en anglais)

hachage/ empreinte

- principe:
  - une fonction non réversible H:
    - connaissant H(x), il est très difficile de trouver y tel que H(y)=H(x)
  - telle que deux empreintes différentes correspondent forcément à deux textes différents
  - la probabilité d'avoir deux empreintes identique est très faible

hachage: applications

- authentification des utilisateurs:
  - on stocke la version hachée du mot de passe
  - un grain de sel permet d'éviter que deux personnes qui ont le même mot de passe aient la même empreinte
- copie optimisée de fichiers
- vérification de l'intégrité de fichiers

Hachage: algo classiques

- MD4 (mdp windows NT & Co)
- MD5 (mdp unix): empreinte de 128 bits, considéré comme faible (collisions)
- sha-1: empreintes de 160 bits (solidité mise en doute actuellement)
- sha-2: empreintes de 256, 384 ou 512 bits au choix
- utilisation d'un algo de chiffrement: le mot de passe est transformé en clef pour chiffrer un texte connu. ex. connu: DES modifié itéré 25 fois pour les mots de passe unix.

Identification et authentification

- **identification**: définir l'identité de l'utilisateur
- **authentification**: permet de vérifier l'identité fournie (authentification simple vs authentification forte)
  - via un élément que l'utilisateur connait (mot de passe, ...)
  - via un élément que l'utilisateur possède (carte à puce, certificat, ...)
  - via biometrie

authentification

- élément clef pour assurer :
  - la confidentialité et l'intégrité des données via un contrôle d'accès: seules les personnes identifiées, authentifiées et habilités à le faire peuvent accèder/ modifier les données
  - la non-répudiation et l'imputabilité (preuve d'une transaction, ...)
- Authentification unique (SSO: Single Sign On)
  - l'utilisateur s'authentifie une fois
  - il a accès à toutes les ressources du réseau
  - cf partie technique (keberos, ...)

Authentification de base sous unix

- authentification par login/mot de passe
- l'emprunte du mot de passe (+ un peu de sel): stockée dans /etc/passwd ou /etc/shadow ou ~
- Algo: des, md5, blowfish
- lorsqu'un utilisateur s'authentifie
  - on calcule l'emprunte (+ le sel) du mot de passe qu'il fournit
  - on compare le résultat à l'emprunte stockée

Authentification sous unix: PAM

- PAM: pluggable authentification modules
- mécanismes permettant d'intégrer des modes d'authentification variés via une interface unique
- via la configuration de PAM (et l'existence du module concerné), on peut faire supporter à de nombreux unix des systèmes d'authentifications variés (carte à puce, mot de passe jetables, annuaire LDAP, ...)

Authentification de base sous windows

- 2 algo de chiffrement: LanMAN (faible) et NTLM
  - pour rester compatible avec un parc ancien
  - mdp < 15 car. chiffrés en LanMAN et en NTLM
- attaques sur LanMAN: de quelques secondes à quelques heures pour trouver un mot de passe alphanumérique par force brute
- LanMAN:
  - désactivable sur les windows 2000sp2+
  - désactivé sur windows Vista

LanMAN: algorithme

- mot de passe tronqué à 14 caractères ou mis à 14 (bourrage avec des caractères nuls)
- mis en majuscule et coupé en deux parties de 7 caractères
- chaque partie:
  - utilisée comme clef de chiffrement DES à 56 bits pour chiffrer la chaîne « KGS!@#$% »
  - on concatène les deux résultats de 8 octets pour obtenir une emprunte LanMAN de 16 octets

133

attaque par force brute

- on calcule l'emprunte de tout ou partie de l'espace des mots de passe et on compare à l'emprunte stockée
- attaques utilisant des jeux d'empruntes totalement ou partiellement pré-calculées
- attaque par dictionnaires (+ modifications classiques)
- outils: lc4 (windows), john the ripper

134

exemples d'autres attaques

- espionnage du réseau:
  - pour récupérer les mots de passe en clair
- remplacement d'une machine par une autre:
  - l'utilisateur s'authentifie sur la machine du pirate en croyant s'authentifier sur un serveur
- compromission d'une machine
  - sur le serveur distante; on remplace les programmes de login et autres
  - sur le poste client: on met en place un keylogger (il en existe de compatibles avec les claviers virtuels)

135

SSH

- ssh est à la fois
  - un protocole
  - une commande
  - un ensembles d'outils dont il existe diverses version de diverses origines

136

SSH

- ssh permet de relier
  - des machines sûres et non compromises
  - à travers un réseau non sûr
    - but: éviter l'écoute passive ou active de la communication
  - l'ensemble des échanges est chiffré
  - les machines sont authentifiées

137

SSH

- authentification des machines
- chiffrement de session
- authentification des utilisateurs
- tunneling
- boîte à outil ssh

138

authentification des machines

- chaque machine a un couple clef privée/publique
- chaque machine doit avoir la clef publique de l'autre
- quand ce n'est pas le cas, cette clef peut être fournie par l'une des machines à l'autre qui la sauvera localement
  - dans ce cas, l'authentification de l'autre machine ne peut être garantie lors de cette première connexion
  - compromis pour faciliter l'adoption du protocole ssh face à la difficulté de diffuser les clefs de façon simple et sûre

Authentification des machine: processus

- les deux machines échangent des informations sur les protocoles de chiffrement qu'ils supportent (algo de chiffrement symétrique, à clef pub/priv, algo de hash, algo de signature de messages)
- le client génère une d'une clef de session pour algorithme symétrique
- il la transmet au serveur en la chiffrant avec la clef publique du serveur et indique l'algo de chiffrement utilisé
- le serveur envoie un message de confirmation chiffré avec le clef de session
- le reste de la communication est chiffrée avec la clef de session et l'algorithme de chiffrement symétrique choisi

Authentification des utilisateurs

- authentification par pam (mdp, one time password, ...)
- authentification par clef publique
  - l'utilisateur possède un couple clef privée/publique
  - la clef privée est sur la machine cliente protégée par une phrase d'accès
  - la clef publique est transférée par un moyen sur sur le serveur dans le fichier authorized_keys de l'utilisateur

authentification par clef publique

- l'utilisateur fournit la phrase d'accès à sa clef privée
- la machine client déchiffre la clef privée de l'utilisateur et l'utilise pour générer une signature qui est envoyée au serveur
- le serveur tente de valider cette signature à l'aide des clefs publiques présentes dans le fichier authorized_keys de l'utilisateur
- en cas de succès, l'accès est autorisé

processus du point de vue de l'utilisateur

- générer un couple clef publique/privée sur le poste client (ex.: ssh-keygen -t dsa. clef privée: id_dsa, publique: id_dsa.pub)
- tranférer la clef PUBLIQUE sur le serveur et l'ajouter au fichier contenant les clefs publiques de l'utilisateur (ex.: ~/.ssh/authorized_keys
- la connexion est ensuite possible sans mot de passe (si la stratégie de sécurité du serveur l'autorise)
- il est possible de placer des restrictions (IP d'origine, commande autorisée, ...) pour chaque clef présente dans le authorized_keys.

agents d'authentification: ssh-agent

- agent d'authentification ssh: mémorise les clefs en mémoire vive pour éviter à l'utilisateur de taper une clef à chaque utilisation
- principe: ssh-agent est le processus père (ou un ancètre) du processus qui réalise la connexion ssh
- en pratique:
  - ssh-agent est lancé au démarrage de la session graphique X
  - on lance à la main « ssh-agent bash » ou « ssh-agent xterm »

agents d'authentification: ssh-add

- ssh-add: commande utilisateur pour ajouter une clef en mémoire

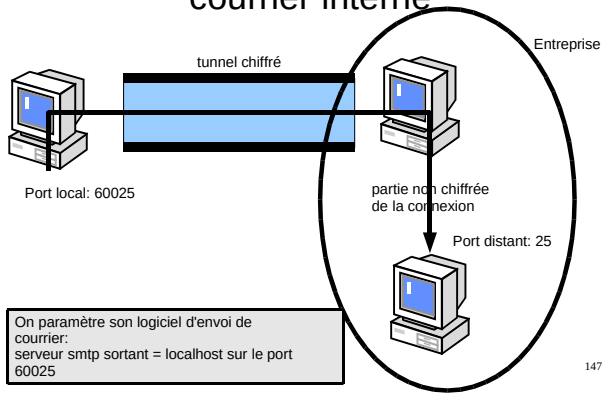
145

tunnel SSH

- ssh permet de rediriger des connexions tcp effectuées sur un port donné du client vers un port donné d'une machine accessible depuis le serveur
- il permet de faire de même d'un port du serveur vers le client
- utilisation traditionnelle (option -X): redirection X11
- vpn du pauvre : accès à un intranet depuis internet

146

tunnel SSH: accès à un serveur de courrier interne



147

réseau

- tcp/ip est supporté depuis mathusalem par tous les Unix
- configuration:
  - adresse IP
  - routage
  - services réseau utilisés par la machine
  - services réseaux fournis par la machine

148

réseau : interface réseau

- une adresse ip peut-être affectée à chaque interface réseau
- nom des interfaces réseau
  - Linux: eth0, eth1, eth0:0 (alias: ràf)
  - OpenBSD, FreeBSD: nom spécifique au pilote de la carte (ex.: pcn0, vr0, fxp0, ...)
- interface spécifique:
  - interface de bouclage: lo sous Linux
  - liaison point à point, ppp, ...: ppp, tun0, ...
- le noyau doit contenir directement ou via modules:
  - le pilote de la carte
  - les pilotes des protocoles réseau utilisés

149

Configuration d'une interface réseau: ifconfig

- ifconfig: configurer une interface réseau
  - syntaxe dépendant de l'OS: ifconfig interface options
  - options:
    - up/down,
    - adresse ip, masque, mtu, ...
    - media (10/100/..., half/full duplex), adresse ethernet, ...
- ifconfig: exemples
  - ifconfig -a : affiche toutes les interfaces (+ informations)
  - ifconfig eth0 192.168.24.85 netmask 255.255.255.0 up: configure et active eth0

150

Configuration d'une interface réseau: via des scripts/fichiers de configuration

- Linux debian: /etc/network/interfaces: adresse IP, masque, ...

```
auto eth0
iface eth0 inet static
        address 195.221.165.248
        netmask 255.255.255.0
        network 195.221.165.0
        broadcast 195.221.165.255
        gateway 195.221.162.249
```

- OpenBSD: /etc/hostname.nomIF

```
inet 192.168.197.55 255.255.255.0 NONE
```

- FreeBSD: /etc/rc.conf

```
ifconfig_vx0="inet 195.159.221.165  netmask
  255.255.255.0"
```

151

Etat d'une interface réseau

- ifconfig nomInterface

```
fxp0:
  flags=8943<UP,BROADCAST,RUNNING,PROMISC,SIMPLEX,MULTICAST>
  mtu 1500
        inet 192.168.161.1 netmask 0xffffff00 broadcast
  192.168.161.255
        inet6 fe80::2a0:c9ff:fe9e:dad2%fxp0 prefixlen 64
  scopeid 0x1
        ether 00:a0:c9:9e:da:d2
        media: Ethernet autoselect (100baseTX <full-duplex>)
        status: active
```

- netstat -i:

```
$netstat -i -I fxp0
Name  Mtu   Network       Address            Ipkts Ierrs    Opkts
  Oerrs  Coll
fxp0  1500  <Link#1>    00:a0:c9:9e:da:d2   918366     0   952442
  0    0
fxp0  1500  192.168.161   192.168.161.1       1916     -    65737
  -    -
fxp0  1500  fe80:1::2a0 fe80:1::2a0:c9ff:        0     -        0
  -    -
```

152

test de connectivité: ping

- ping: envoie un paquet icmp echo request et attend un paquet icmp echo response
- si ça ne passe pas, il est possible que ça soit le paquet retour qui n'arrive pas
- test à compléter par une analyse de trames (tcpdump, ethereal, ...) pour voir où est le problème
- nmap, hping permet de faire de même via tcp ou udp en choisissant le port source (pour éviter certains filtres)
- arp: gestion du cache arp

153

Demo:

- demo où l'on teste la connectivité entre deux postes séparés par un routeur
- test entre les machines directement connectées
- test entre les deux machines extrêmes
- le second poste aura un routeur par défaut incorrect
  - les paquets ne revienne pas
  - mettre en évidence
    - que le paquet part (analyse de trame)
    - que la paquet arrive
    - que le paquet retour ne part pas (pb arp)

154

routage

- le routage permet à deux machines non directement reliées de communiquer via des machines intermédiaires appelés routeurs.
- un poste a en général une configuration simple: routeur par defaut
- cas plus complexes:
  - routage statique
  - routage dynamique (sort du contexte de cet enseignement)
- machine routeur:
  - accepte les paquets destinés à d'autres hôtes
  - le routage ip doit être activé

155

routage : configuration

- routes statiques: via la commande route ou fichier de configuration
- fichiers de configuration
  - Debian Gnu Linux:
    - /etc/network/interfaces : adresse IP, **routeur par défaut** & Co
    - debian: /etc/network/options: active le routage
    - ubuntu: /etc/sysctl.conf pour l'activation du routage
  - FreeBSD:
    - /etc/rc.conf: routeur par défaut, routes statiques
  - OpenBSD:
    - /etc/mygate: routeur par défaut

156

netstat -r: table de routage

- affiche la table de routage
  - une entrée pour chaque sous-réseau de chaque interface réseau (le champ passerelle est à 0.0.0.0)
  - une entrée pour le routeur par défaut (le champ destination est à 0.0.0.0)
  - une entrée par route statique.
- option -n : pas de conversion des valeurs numériques en valeurs litérales (évite l'utilisation du dns)

```
petit@sarge-test:~$ netstat -rn
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic   MSS Fenêtre irtt Iface
192.168.100.0   192.168.244.60  255.255.255.0   UG        0 0          0 eth0
192.168.244.0   0.0.0.0         255.255.255.0   U         0 0          0 eth0
0.0.0.0         192.168.244.2   0.0.0.0         UG        0 0          0 eth0
```

Netstat

- obtenir des informations sur la configuration/les logiciels réseau d'un ordinateur
- des options dépendant du système d'exploitation
- exemple d'utilisation:
  - option commune: -n: désactive la résolution des adresses numériques (dns, ports, ...)
  - netstat -a: surveillance de l'état des connexion réseau
  - netstat -i : stat. trafic des interfaces réseau
  - netstat -r: table de routage
  - netstat -s: stat. par protocole tcp/ip

158

netstat -a: surveillance de l'état des connexion réseau

- « netstat -taupe » :
  - t: tcp
  - a, --all
  - u: udp
  - p: pid et programme auquel appartient la socket
  - e ou –extended (on peut aussi mettre -ee pour plus de détail)

159

netstat -s: stat. par protocole tcp/ip

160

services réseau, super-serveurs

- notion de socket
- numéros de ports
- démarrage via script
- démarrage via inetd/xinetd
- tcpd: tcp wrapper
- rpc et portmapper

161

inetd/xinitd

- pb: beaucoup de services potentiels qui ne servent pas tous ou rarement
- Solution pour les services réseau : on ne lance les services peu utilisés que lorsqu'une connexion se présente
- inetd: daemon qui gère les autres daemon
- inetd.conf:

```
ftp    stream  tcp     nowait  root    /usr/libexec/ftpd
      ftpd -l
auth   stream  tcp     wait    root
  /usr/local/sbin/identd  identd -w -t120
pop3    stream  tcp     nowait  root    /usr/sbin/tcpd
  /usr/sbin/ipop3d
```

162

tcp wrappers

- But: interdire l'accès à des services en fonction de la machine demandeuse
- depuis inetd via tcpd
- via bibliothèque dynamique ad hoc
- ràf: la syntaxe du fichier de configuration sera détaillée dans la prochaine version de ce document

163

rpc/portmapper

- Principe:
  - un serveur qui démarre indique à portmap sur quel port il écoute et quel service il rend (/etc/rcp)
  - un client qui veut se connecter à un serveur demande au portmapper (port 111)sur quel port écoute le serveur qu'il veut joindre
- application: nfs, nis
- commandes utilisateur : rpcinfo
- rcp et sécurité
  - tcp-wrapper
  - fixer le port utilisé par les serveurs (nfs, nis le permettent)

164

Demo:

- donner un exemple de capture de trame avec nis ou nfs pour montrer le processus (ràf: préciser le contexte de l'exemple dans la prochaine version de ce document)
- Le but est de montrer la connexion sur le port 111 (portmapper) pour trouver le port sur lequel écoute réellement le service.

165

Partage de fichiers systèmes

- gérer de façon centralisée les fichiers de configuration d'un parc entier
- quels fichiers partager ?
  - utilisateurs, groupes et autres informations communes à un parc/domaine
- comment les partager ?
  - par diffusion d'un fichier maître
    - push: gestion centralisée, accès RW du maître aux client (sécurité)
    - pull:mode plus décentralisé, sécurité (accès R suffisent)
  - en remplaçant/complétant les fichiers par la consultation en temps réel d'un serveur central:
    - NIS, LDAP

166

NIS: gestion des utilisateurs dans un domaine

- partage de bases de données d'informations
- De nos jours, on lui préfèrera ldap (sera vu en M1)
- NIS s'appuie sur rpc
- NIS et la sécurité:
  - repérage des serveurs par diffusion (corrigé): usurpation
  - diffusion publique d'informations critiques (empreintes des mots de passe): attaque en force brute
- NIS+: même but mais conception très différente. Sécurisé mais lourd, peu utilisé.

167

NIS

- sélection de la source d'informations administratives
  - +
  - nsswitch.conf
  - pam
- Fonctionnement

168

NSS, name service switch: problèmatique

- Historiquement, les données de certains services étaient dans des fichiers situés dans /etc. Exemples:
  - Noms/adresses de machines : /etc/hosts
  - Utilisateurs: /etc/passwd
  - Groupes: /etc/group
- De nos jours, certaines de ces informations sont totalement ou partiellement obtenues du réseau:
  - DNS, NIS, LDAP, ...

NSS: cahier des charges

- Cahier des charges:
  - Avoir un système évolutif
  - Permettant de sélectionner la sources des données d'un service
  - Capable d'intégrer facilement de nouvelles sources de données
  - La liste des services concernées est figée (on ne peut pas faire gérer un service non prévu à l'origine)
- Philosophie proche de celle de PAM pour l'authentification

NSS: implémentation

- La liste des services est cablée dans la libc
- Un fichier de configuration permet de préciser pour chaque service une ou plusieurs sources de données
- Une interface standardisée permet de créer des greffons pour de nouvelles sources de données sans avoir à modifier la libc

NSS: liste des services concernés

Nom service	description	fonctions de la libc utilisant la base de donnée
aliases	les alias de courrier électronique (obsolete)	
ethers	adresses ethernet et les adresses IP correspondantes	
group	liste des groupes auxquels appartiennent les utilisateurs du système	get grent
hosts	noms et adresses IP de machines	gethostbyname
networks	noms et masques de réseaux	getnetent
passwd	comptes utilisateur du système + informations sur ces comptes (UID, GID, ...)	getpwent
protocols	les protocoles internet disponibles	getprotoent
publickey	utilisé par les secure rpc (sert à NFS et NIS+)	
rpc	noms et numéros de programmes rpc	getrpcbyname
services	correspondance entre nom d'un service et protocole/port normalisé utilisé	getservent
shadow	mots de passe chiffrés des utilisateurs présents dans passwd	getspnam

/etc/nsswitch.conf

- Là, on donne exemple de fichier et on l'explique

Exemples de sources de données

service	description	bibliotheque correspondante
compat	équivaut à « files, nis » mais permet en plus l'utilisation de la syntaxe +/-user dans /etc/passwd	/lib/libnss_compat.so.X
db	en utilisant des fichiers au format DB	/lib/libnss_db.so.X
dns	Via réseau en interrogeant un serveur DNS	/lib/libnss_dns.so.X
files	en utilisant les fichiers présents sur la machine (/etc/passwd, ...)	/lib/libnss_files.so.X
ldap	Via réseau en interrogeant un serveur LDAP	/lib/libnss_ldap.so.X
mdns	Via réseau en utilisant les paquets multicast DNS (cf zeroconf, dnsext)	/lib/libnss_mdns4.so.X et /lib/libnss_mdns6.so.X
nis	Via réseau en interrogeant un serveur NIS	/lib/libnss_nis.so.X
nisplus	Via réseau en interrogeant un serveur NIS+	
wins	Via réseau en interrogeant un contrôleur de domaine windows	

Commande getent

- Interroge une base de données
- getent utilise les bases de données précisées par nsswitch.conf
- Outil pratique pour tester la mise en service d'une nouvelle base de données
- Exemple:

```
$ getent passwd petit
petit:x:2028:2002:Pascal Petit:/nhome/fs2/petit:/bin/bash
```


NFS: généralités

- permet le partage de dossier
  - exporte tout dossier du système
  - export limité par les SGF
  - fiable, performance améliorables, sans état (cookie)
- s'appuie sur rpc (mais port 2049 réservé et utilisé de plus en plus pour nfsd)
- principe:
  - un dossier distant exporté est monté sur un dossier local comme on le ferait d'un SGF
  - les utilisateurs et groupes locaux sont censés être les mêmes sur le serveur et le client
  - nis est une solution traditionnelle pour garantir cette correspondance entre UID-GID serveur et clients

NFS

- les différentes versions de nfs
  - 1985: NFSV2 (première version publique)
    - réseaux locaux, udp
    - fichier posix 32 bits
    - performance en écriture médiocre (impossibilité de bénéficier du cache du serveur)
  - 1994: NFS V3
    - fichiers posix 64 bits
    - réseaux locaux, tcp ou upd
    - performances en écriture correctes
  - NFS V4:
    - RFC 2624, 3010, 3530.
    - dans une version ultérieure de ce document

NFS

- sécurité: avant la V4, un désastre :-)
  - pas d'authentification des postes clients
  - pas de chiffrement des données
  - root sur un poste client peut obtenir l'accès à toutes les données via une manipulation simple
  - rootsquash (par défaut), nosuid
  - ports: non fixe par défaut => difficile à filtrer
  - solutions (peu utilisées): secure RPC, kerberos
- verrous: un problème usuel non résolu (sera détaillé dans la prochaine version de ce document)
- serveur nfs dédiés (appliances)

NFS côté serveur

- fichiers de configuration
  - /etc/exports:
    - sur le serveur
    - contient les options et machine autorisées
    - utilisé par mountd et par nfsd
    - certains systèmes d'exploitation imposent la construction d'une version binaire de exports à l'aide de la commande exportfs (share sous Solaris)
- daemons
  - mountd: montage des fichiers
  - nfsd: accès au fichiers

NFS côté client

- fichiers de configuration
  - /etc/fstab: SGF montés (y compris nfs)
- Daemons
  - biosd et nfsd: fournissent un cache au niveau du client (nfs v2+). nb nfsd joue sur les perfs.
- commande:
  - mount/umount
  - options de montage classiques:
    - soft (retour erreur en cas de srv HS), intr
    - hard (blocage si srv HS)
    - rsize=8192, wsize=8192 (tampons en lecture et écriture)
    - tcp, nosuid, nodev:
- ports privilégiés: exigés par certains serveurs

NFS: demo

- sur un client nfs:
  - df et mount pour voir les systèmes de fichiers montés par nfs
  - /etc/fstab
- sur le serveur
  - /etc/exports pour voir les systemes de fichiers autorisés à l'export

181

NFS V4

- inspiré d'AFS, changement complet de philosophie:
  - incompatible avec les versions précédentes
  - à état,
  - récupération des sessions en cas de crash serveur ou client
  - support du chiffrement,
  - cache client agressif
  - support des ACL, d'utf8
  - performances correctes même sur un lien internet à haute latence
  - regroupement des requêtes réseau
  - supporte la réplication et la migration (rediriger les requêtes d'un serveur saturé vers un autre peu chargé)

185

Bibliographie sur NFS

- « Unix, guide de l'administrateur » de Nemeth, Snyder et Al, Campus press
- NFS V4: http://www.ietf.org/html.charters/nfsv4-charter.html
- http://www.linux-france.org/prj/inetdoc/cours/admin.reseau.fs/admin.reseau.fs.single.html
-

186

Bibliographie sur NFS

- RFC:
  - 1094: NFS protocol specification
  - 1813: NFS V3
  - 2054: webnfs client spec.
  - 2055: webnfs server spec
  - 2224: NFS URL scheme
  - 2623: NFS V2 et V3 security Issues
  - 2624: NFS V4 Design consideration
  - 3010: NFS V4
  - 3530: NFS V4 protocol

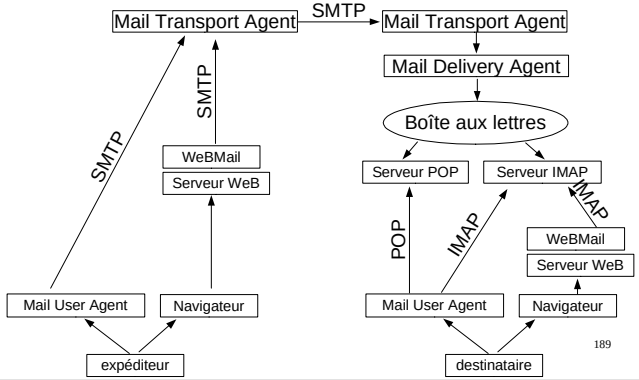
187

SMTP: notions de base

- L'envoi et la réception d'un courrier mettent en jeu de nombreux outils et protocoles. Notamment :
  - MUA: Mail User Agent ou agent utilisateur (mail, mutt, thunderbird (mozilla), eudora, voire même Outlook express si n'a peur de rien),
  - MTA: Mail Transport Agent ou agent de transport (serveur smtp: sendmail, postfix, qmail, exim, voire exchange),
  - MDA: Mail Delivery Agent ou agent de délivrance du courrier, chargé par le MTA de déposer le courrier dans la boîte aux lettre de l'utilisateur (exemple: procmail, ...)
  - POP3, IMAP : protocole permettant au MUA d'accèder aux boîtes aux lettres

188

SMPT: architecture



189

SMTP:

- Rfc821 puis 2821: protocole SMTP
- Rfc822 puis 2822: format des courriers
- D'autres documents concernent le courrier: POP3, IMAP, ETRN, MIME, ...

190

SMTP: commandes

- Contrôle de session
  - HELO/EHLO nomServeurExpéditeur
  - RSET/QUIT
  - NOOP
- Traitement des courriers
  - MAIL From:<adresse>
  - RCPT To:<adresse>
  - DATA
  - VRFY/EXPN

191

SMTP: DEMO (1)

- On se connecte sur le port 25 d'un serveur existant et on tape quelques commandes pour jouer
- On commente les capacités annoncées par le MTA
- On commente les codes d'acquittement (il est de bon ton de faire quelques fautes de frappe pour générer des erreurs)

192

SMTP: DEMO (2)

- On se connecte sur le port 25 d'un serveur existant et on envoie un courrier:
  - Un courrier raisonnable
  - Un courrier avec un expéditeur bidon et des entêtes bidon
- On commente encore les codes d'acquittement
- On montre un refus de relais

193

SMTP: acquittement (rfc 2821)

- un code normalisé suivi par du texte (non normalisé)
- Codes:
  - 1xy: acquittement positif préliminaire
  - 2xy: acquittement positif
  - 3xy: acquittement positif intermédiare : commande acceptée mais attente de données pour compléter (ex: DATA)
  - 4xy: acquittement négatif provisoire
  - 5xy: acquitement négatif définitif

194

SMTP: acquittement

- Codes: signification du deuxième caractère
  - x0z: erreur de syntaxe:
  - x1z: Information: statut ou aide
  - x2z Connexions: réponse ayant trait au canal de communication
  - x5z Système de courrier : réponses indiquant l'état du système de courrier suite à la commande
- Codes: 3e caractères: précise le deuxième

195

SMTP: exemples d'acquittements

500 Syntax error, command unrecognized

501 Syntax error in parameters or arguments

502 Command not implemented

214 Help message

220 <domain> Service ready

250 Requested mail action okay, completed

450 Requested mail action not taken: mailbox unavailable (e.g., mailbox busy)

550 Requested action not taken: mailbox unavailable (e.g., mailbox not found, no access, or command rejected for policy reasons)

452 Requested action not taken: insufficient system storage

552 Requested mail action aborted: exceeded storage allocation

SMTP: demo (3)

- Un serveur mal configuré qui retourne un code en 4xy en cas d'utilisateur inexistant
- Conséquence: pas de courrier d'erreur pour l'expéditeur dans que son MTA n'a pas fini d'essayer d'envoyer le courrier

Message: entête et corps

- une ligne vide sépare le corps des entêtes
- une entête logique peut se poursuivre sur plusieurs lignes :
  - les lignes de continuation commencent par un espace (espace, tabulation, ...)
- les entêtes comprennent des entêtes placées par le MUA et des entêtes de traçage placées par le MTA
- Enveloppe: information échangées entre MUA et MTA ou entre MTA, pas dans le message

Entête des messages (rfc 2822)

- Notion d'enveloppe
- Champs de traçage
  - Return-path
  - Received
- Adresses et champs utilisateur
  - From:,Sender,Reply-To:
  - To:, CC:, BCC:
- Champs informationnels souvent optionnels
  - Date, Subject, X-...
  - Message-ID
  - In-Reply-To:, References:

format des adresses (RFC 819)

```
<mailbox> ::= <local-part> "@" <domain>

<local-part> ::= <string> | <quoted-string>

<string> ::= <char> | <char> <string>

<quoted-string> ::=  """ <qtext> """

<qtext> ::=  "\" <x> | "\" <x> <qtext> | <q> | <q>
  <qtext>

<char> ::= <c> | "\" <x>

<a> ::= any one of the 52 alphabetic characters A
  through Z in upper    case and a through z in
  lower case

<c> ::= any one of the 128 ASCII characters except
  <s> or <SP>
```


format des adresses (2)

```
<d> ::= any one of the ten digits 0 through 9

<q> ::= any one of the 128 ASCII characters except
  CR, LF, quote ("),   or backslash (\)

<x> ::= any one of the 128 ASCII characters (no
  exceptions)

<s> ::= "<", ">", "(", ")", "[", "]", "\", ".",
  ",", ";", ":", "@", """, and the control
  characters (ASCII codes 0 through 31 inclusive
  and 127)

Le nom de domaine peut être écrit en majuscules ou
  minuscules mais il n'est pas sensible à la casse
  (ShayoL.org et shayol.org sont le même domaine).

La « local-part » peut être sensible à la casse
  mais c'est déconseillé.
```

format des adresses (3): exemples

- Exemples corrects:
  - pascal.petit@shayol.org
  - pascal.petit@ShaYol.or
  - Pascal.Petit@shayol.or
  - petit+adieve@shayol.org
  - "Pascal Petit"@foo.fr
  - Pascal\, Petit@shayol.org
  - petit@[81.56.171.187]
- Exemples incorrects:
  - Pascal Petit@foo.fr
  - petit@toto$.fr
  - Cécile@toto.fr

SMTP: DEMO

- On prend les courriers envoyés dans la première demo.
- On met en évidence les champs de traçage (Received) et les informations récupérées de l'enveloppe (Return-path)
- On commente les autres champs et notamment le fait que le champ From n'a rien à voir avec MAIL FROM:, et le champ To: avec le RCPT TO:

MIME (RFC2045 à 2049)

- Multipurpose Internet Mail Extension
- permet de transporter des données de types variés dans des courriels
- Principe:
  - l'entête contient une description du type de données
  - les champs d'entête peuvent utiliser un codage indépendant de cette description (car rien ne garantit qu'elle sera lue avant)

MIME champs d'entête

- MIME-Version: numéro de version (2.0)
- Content-Type: indique le type d'informations sous la forme : *type/subalterne; attribut=chaîne [; attribut=chaîne]*
- Exemples de types:
  - text/plain; text/html; text/enriched
  - image
  - multipart: le courrier composé de plusieurs partie avec son champ Content-Type et, éventuellement, Content-Transfert-encoding. ex: multipart/alternative.
  - application/pdf; application/postscript; application/zip;

MIME: champ d'entête (2)

- les données sont codées (image, texte avec des caractères accentués, ...) pour palier les limitations des passerelles
- champ Content-Transfert-Encoding: précise le codage utilisé (rfc 2045):
  - 7bit
  - 8bit
  - binary (site supportant 8BITMIME)
  - quoted-printable
  - base64

MIME

- codage des entêtes
- RFC 1524: problème posé par MIME aux MUA
- RFC3030: répartition des attachements sur plusieurs courriers

MIME: demo

- on analyse les entêtes de divers courriers contenant des attachements, en multipart/alternative, ...

208

Accusé de réception

- par défaut: accusé de non réception (temporaire, définitif)
- accusés de réceptions supportés par les MUA : impose que le MUA du destinataire le supporte
- DSN (RFC 3461), type MIME correspondant (RFC 3462), format (RFC3464)

209

Configuration de base d'un MTA: postfix

- Les questions auxquelles répondre :
  - domaine indiqué sur les courriers sortant
  - domaines gérés par le MTA (les courriers adressés à ces domaines sont gérés localement)
  - pour quels clients accepte-t-on de relayer le courrier ?
  - de quelles destinations relayer le courrier ?
  - méthode de livraison du courrier: directe ou indirecte ?

210

Domaines gérés par le MTA: courrier entrant

- paramètre *mydestination* de postfix
- mydestination=$myhostname localhost. $mydomain exemple1.com exemple2.com
- liste des domaines dont le courrier reçu sera délivré dans les boîtes aux lettres locales
- toto@exemple1.com et toto@exemple2.com auront la même boîte aux lettres

211

de quels clients relayer le courrier ?

- paramètres *mynetwork* et m*ynetworkstyle* (ignoré si mynetwork est défini)
- mynetwork=127.0.0.0/8 192.168.196.0/24 81.56.171.187/32
- mynetworkstyle=host|subnet|class : autorise les machines: machine locale, machine du même sous-réseau, machine de la même classe réseau
- mynetwork= liste de réseaux dont les machines seront autorisées

213

De quelles destinations relayer le courrier

- paramètre *relay_domains* de postfix
- par défaut: mydestination
- sert pour définir les domaines dont on est MX de secours

214

Méthode de livraison du courrier: directe ou indirecte

- paramètre *relayhost* de postfix
- indique la machine à laquelle envoyer le courrier sortant
- s'il est vide, le serveur de courrier gère lui-même l'envoi à la destination (détermination des MX puis envoi à l'un des MX)

215

bibliographie:

- « unix, guide de l'administrateur », Nemeth, Snyder, Seebass et Hein; 2001, Campus Press: un peu daté mais excellent tant sur les aspects techniques que méthodologiques
- « Unix Administration », Moreno; Dunod; 2003: un guide complet traitant tant de Sys V que de BSD. rigoureux mais abord un peu raide.

216

Bibliographie

- « linux administration système », Pons, ENI, 2003: un ouvrage sans prétention qui traite correctement les points clefs de l'administration système sans passer sur les notions de fond.
- les cours réseau sur linux-france.org: http://www.linux-france.org/prj/inetdoc/cours
- single unix v3 specification : http://www.unix.org/online.html
- linux standard base : http://refspecs.freestandards.org/lsb.shtml
- Documentation de postfix en vf: http://x.guimard.free.fr/postfix/

217

Sujet optionnels

- Les transparents qui suivent traitent de sujets optionnels
- Ces sujets ne seront pas forcément traités en cours

218

Lutte contre le spam

- Là, on met des généralités
- Architecture classique avec serveur entrant, serveur interne de mbox
- Indiquer les effets du refus du spam à chaque étape (conclusion: il vaut mieux refuser sur le serveur entrant lors de la soumissions du spam)
- Citer quelques techniques classiques de lutte contre le spam en s'appuyant sur : http://2005.jres.org/tutoriel/expose-tutoriel-spam.pdf

219

Listes grises

220

RBL

221

Filtre baylesiens

222

Open BSD spamd

223

Étude de cas: installation d'outils sur postfix/debian etch

224

SPF

- MX: enregistrement du DNS identifiant les serveurs de courriers ENTRANTs d'un domaine
- SPF: méthode pour identifier les serveurs de courrier SORTANT d'un domaine (RFC 4408)
- SPF teste le MAIL FROM: (mfrom) de l'enveloppe du mail
- But: pas d'authentification de l'émetteur avec SMTP. Ainsi, tout le monde peut envoyer un courrier ayant comme émetteur pascal.petit@shayol.org.

225

SPF

- Exemples (2008-04) :

```
$ dig +short TXT cru.fr
"v=spf1 mx a:home.cru.fr/24 a:smtp1.cru.fr/24
  ?all"
$ dig +short TXT freebsd.org
"v=spf1 ip4:69.147.83.53 ip4:69.147.83.54
  ip6:2001:4f8:fff6::35 ip6:2001:4f8:fff6::36
  ~all"
$ dig +short TXT gmail.com
"v=spf1 redirect=_spf.google.com"
$ dig +short TXT _spf.google.com
```

226

SPF:

- Dans un champ TXT du domaine :
  - v=spf1
  - une liste de filtres appliqués les uns après les autres (+filtre: OK si filtre validé, -filtre: pas OK si filtre validé)
  - Fini traditionnellement par un filtre all (valide tout)
  - Qualifiants :
    - « + » : succès
    - « - »: échec
    - « ~ »: échec soft
    - « ? » : résultat neutre
    - Rien: équivaut = « + »

227

SPF: problèmes

- Utilisateurs itinérants
  - Ne peuvent émettre des courriers d'un domaine que des serveurs identifiés
  - Impossibilité d'émettre des courriers de ce domaine en utilisant son fournisseur d'accès ou d'autres serveurs
  - Solution: proposé un service de soumission authentifié de courrier (port 587 en général, utilisant SASL: Simple Authentification and Security Layer)
  - Gare aux filtres

228

SPF: problèmes

- Redirections (/etc/aliases, .forward)
- Dans le cas où une redirection a été placée (par ex. pour se faire transmettre son courrier prof. sur son adresse personnelle), le MAIL FROM: est inchangé

229

SPF, PB redirections : exemple

- Albert écrit depuis foobar.org à Bernard@grosbiz.com.
  - Le serveur de courrier sortant S1 de foobar.org envoit un courrier au serveur entrant de grosbiz.com
  - Le MAIL FROM est de la forme qqc@foobar.org
  - L'enregistrement SPF de foobar.org comprend le serveur S1
- Bernard a redirigé son courrier vers son adresse perso bernard645@fai.fr)
  - Le serveur de courrier sortant de grosbiz.com envoit le courrier reçu au serveur entrant de fai.fr
  - Le MAIL FROM: est inchangé et vaut qqc@foobar.org
  - PB: L'enregistrement SPF de foobar.org ne comprend pas le serveur smtp sortant de grosbiz.com.

230

Solution (?) :SRS (Sender rewriting Scheme) :

- http://www.openspf.org/SRS
- Le serveur de courrier intermédiaire réécrit le « Mail FROM: de façon :
  - À garder trace de sa valeur initiale
  - À y mettre une valeur permettant d'envoyer les rebonds éventuels au serveur intermédiaires
  - Qui les retransmettra au serveur d'origine

231

SPF: bilan en avril 2008

- Peu utilisé mais Utilisé par des poids lourds (hotmail, ...)
- Incompatiblité entre RFC 4408 de spf (mfrom) et RFC 4406 de sender id (pra)
- Impose de changer les pratiques et les logiciels
  - Utilisateurs itinérants
  - Redirections (forward)
- Efficacité insuffisante contre le spam
- À utiliser :
  - Comme un élément d'évaluation parmi d'autres
  - enregistrement laxiste sur son domaine pour faciliter l'acceptation par les domaines qui l'utilisent

232

RFC 4405 (SUBMITER) et RFC 4407 (PRA): déterminer l'expéditeur d'un courrier

- Voir http://www.bortzmeyer.org/4407.html et http://www.bortzmeyer.org/4405.html
- Prendre un exemple de courrier et indiqué d'où sort chaque entête (qui la positionne et quelle intérêt à sa valeur)
- PRA:
  - Sender-id s'appuie dessus
  - Algo très simple mais breveté par microsoft

RFC 4406: Sender ID: Authenticating E-Mail

- Voir http://www.bortzmeyer.org/4406.html
- Amusant: m$ utilise spf et pas sender id. aol utilise sender-id
- Incompatible avec la RFC 4408
- Encore moins utilisé que spf
- Avis perso: à fuir.
-

Bibliographie

- Unix Administration, Jean-Michel Moreno, Dunod
- BSD, Emmanuel Dreyfus, Eyrolles
- TCP/IP Administration de réseau, Craig hunt, O'Reilly
- Documentation de postfix en vf: http://x.guimard.free.fr/postfix/

Administration système unix cours 2

surveillance système

- sar est dans le package sysstat

impressions:

- sysV
- BSD
- IPP: cups