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 generes 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

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

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)

SSH

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

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

SSH

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

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 »

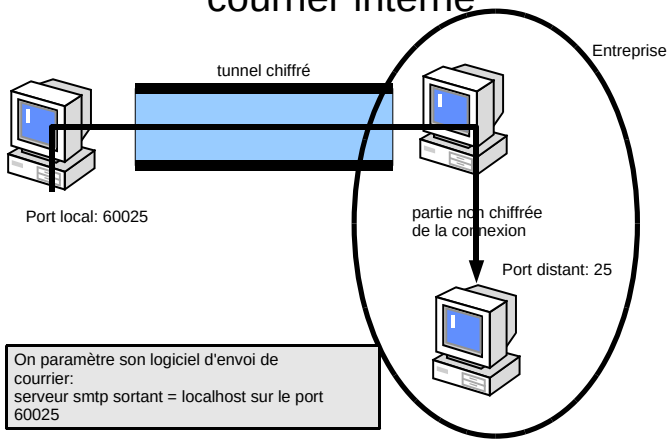
agents d'authentification: ssh-add

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

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

tunnel SSH: accès à un serveur de courrier interne



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

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

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

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"
```

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
  -     -
```

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

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)

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

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

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

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)

netstat -s: stat. par protocole tcp/ip

services réseau, super-serveurs

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

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
```

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

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)

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.

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

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.

NIS

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

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 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)

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
-

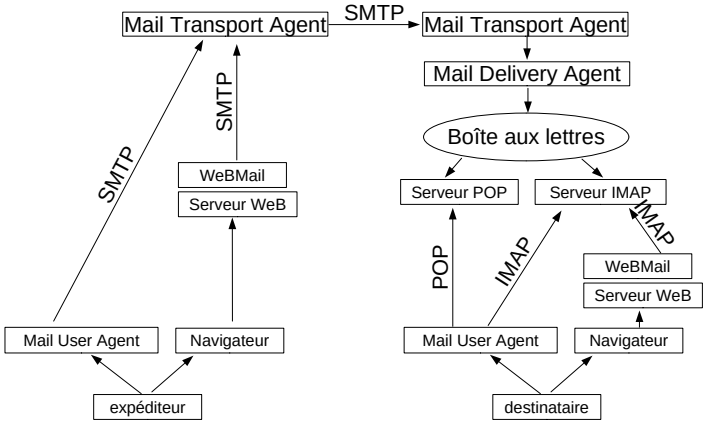
Bibliographie sur NFS

- 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

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

SMPT: architecture



SMTP:

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

SMTP: commandes

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

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)

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

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

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

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.org
  - Pascal.Petit@shayol.org
  - 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, ...

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)

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 ?

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

Domaines virtuels

- voir http://x.guimard.free.fr/postfix/index.php?page=VI (VIRTUAL README)

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

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

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)