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 195.221.165.249
```

- 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"
```

Configuration d'une interface réseau: Solaris

- /etc/hostname.nomIF: contient une entrée: le nom ou l'IP v4 de l'interface (notation CIDR acceptée)
- /etc/nodename: le nom du système local (non FQDN)
- /etc/defaultdomain: le nom de domaine complètement qualifié de la machine
- /etc/defaultrouter:contient les adresses des routeurs que l'hôte pourra utiliser.
- /etc/inet/hosts : hosts database (/etc/hosts est un lien symbolique vers ce fichier)

Références: System administration guide : IP services (voir sur http://docs.sun.com)

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: 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

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