



*Architecture de filtrage réseau :
Filtrage IP
Windows, MacOS X, Unix et Cisco*

Denis Pugnère

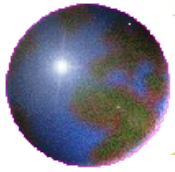
CNRS / IN2P3 / IPNL

d.pugnere @ ipnl.in2p3.fr



Besoin des individus

- **Utilisateurs :**
 - Transparence,
 - convivialité, facilité d'utilisation
- **Exploitants du réseau :**
 - Simplification de l'exploitation
 - Vision globale et cohérente de l'ensemble
 - Information temps réel et facilité d'intervention
- **Partenaires : Confiance, respect des engagements, réactivité**
- **Responsable sécurité :**
 - Vision globale de l'application de la politique de sécurité dans le système d'information
 - Tableaux de bord



Besoins d'un réseau

- Gestion du contrôle d'accès,
- Analyse de contenu, génération de journaux
- Souplesse et évolutivité dans la gestion, facilité d'exploitation,
- Redondance, résistance, qualité de service, performance



État des lieux des réseaux

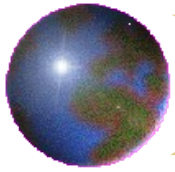
- Périimètre parfois poreux :
 - Ordinateurs nomades (sortent et entrent dans le périmètre)
 - Ré-encapsulation d'IP sur un protocole autorisé
 - VPN (IPSEC ou SSL)
 - Tunnels SSH, ICMP, HTTP
 - Accès modems
 - réseaux sans-fil
 - Extranets, applicatifs mal conçus, etc...
 - Échanges avec des mémoires et disques miniatures (disques et clés USB...)
 - Connexions des appareils nomades : PDA, téléphones, lecteurs MP3...
- Multiplicité des projets, services, partenaires, collaborations





Panorama des menaces

- Tous les systèmes ont des bogues de sécurité
 - Qu'ils soient matériels ou logiciels
 - Voir les avis des différents CERT (dont CERT-A et CERT Renater)
- Certains systèmes sont encore livrés ouverts par défaut (le filtre intégré n'est pas activé)
- De nombreux services réseau inutiles sont activés par défaut



Outils classiques d'attaque

- Scanneurs de réseau (nmap...) : découverte de services réseau accessibles
- Scanneurs de vulnérabilités (metasploit, nessus...) : découverte de failles dans les services/applications accessibles :
 - XSS, injection html, ldap,
- Tests en force brute : connexions (ssh, ftp, http(s), pop(s), imap(s)...) avec logins+mot de passe à la chaîne jusqu'au succès



Utilisation du filtrage

- Pourquoi filtrer ?
 - Se protéger contre les attaques
 - Protéger quoi ?
 - Les services réseaux applicatifs, les services réseaux
 - Les postes individuels, les serveurs
 - L'infrastructure réseau
 - Protéger contre quoi ?
 - L'accès non autorisé
 - L'usurpation (adresse, identité...)
 - Le deni de service
- C'est une des composantes de la politique de sécurité
- Contribue à « Disponibilité, Intégrité, Confidentialité »

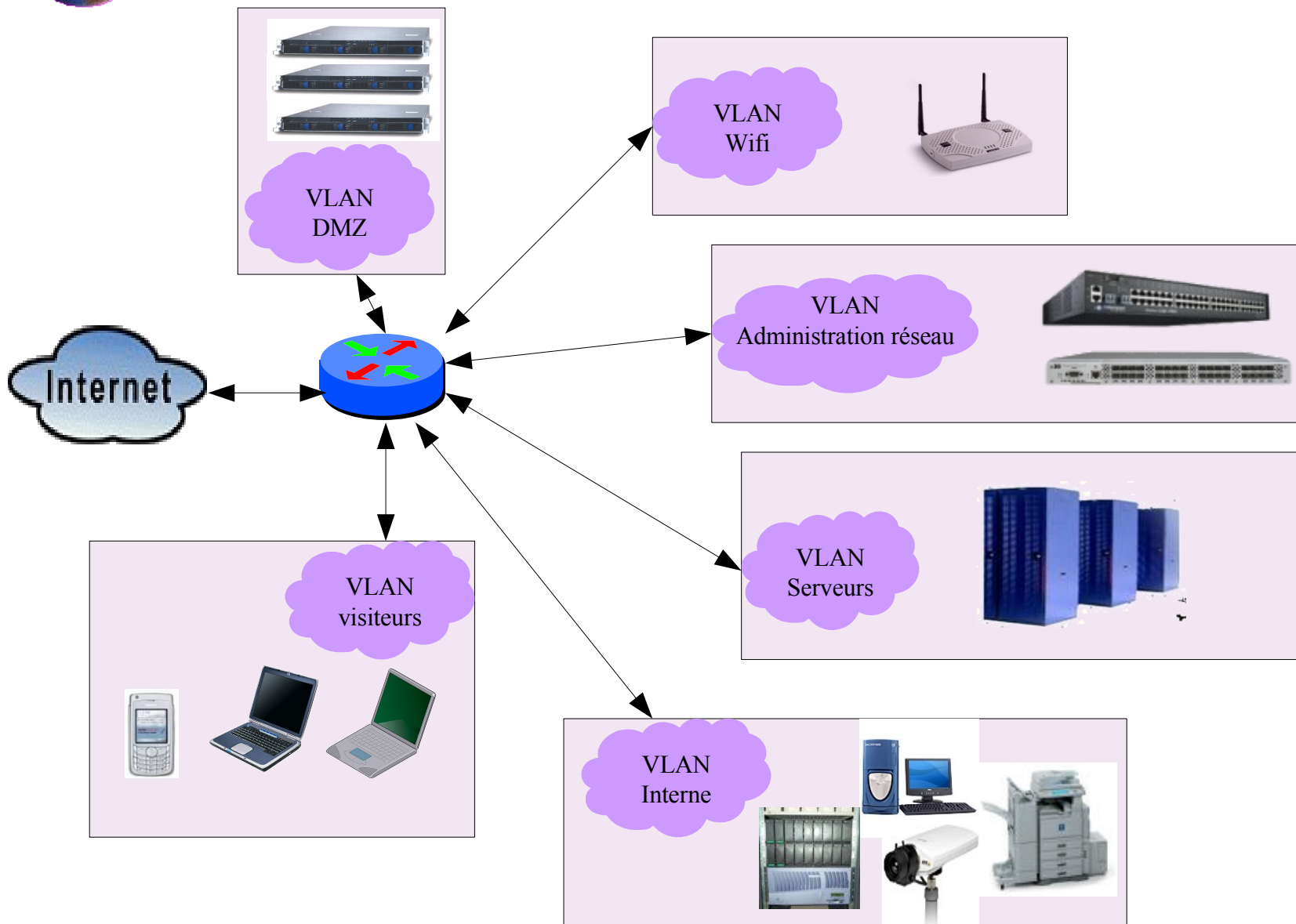


Architecture d'un réseau

- Principes :
 - Adopter une politique de filtrage de type tout interdit sauf... :
 - Permettre à un ensemble de personnes d'avoir accès au strict nécessaire
 - Établir une matrice de flux
 - Séparer les flux, isoler les services, les serveurs, les postes de travail, les classes d'utilisateurs
 - Vérifier que le filtrage actif est bien celui adopté
 - Garder des traces
 - Surveiller les accès (consultation de journaux de bord)
=> détection des évènements anormaux grâce aux traces



Architecture dite « cloisonnée »



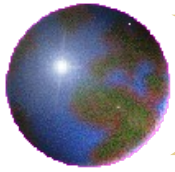


Exemple de matrice de filtrage

\ vers	Internet	VLAN Visiteurs	VLAN DMZ	VLAN Wifi	VLAN Admin réseau	VLAN Serveurs	VLAN Interne
Internet	N/A	NON	FP	NON	NON	NON	NON
Visiteurs	OK	N/A	FP	NON	NON	FP	NON
DMZ	FP	NON	N/A	NON	NON	FP	NON
Wifi	OK	NON	FP	N/A	NON	FP	NON
Admin réseau	NON	NON	NON	NON	N/A	NON	NON
Serveurs	FP	NON	FP	NON	NON	N/A	NON
Interne	FP	FP	OK	FP	FP	FP	N/A

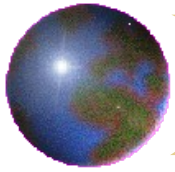
FP : Filtrage partiel

Ou utiliser un diagramme de flux



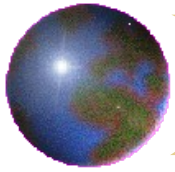
Plusieurs Niveaux de filtrage

- Niveau 2 : adresses adresse M.A.C, par vlan
- Niveau 3 : adresses IP, protocoles, tunnels...
- Niveau 4 : TCP, UDP, ICMP...
- Niveaux applicatifs :
 - utilisation de proxys
 - Filtrage des utilisateurs, groupes : contrôle d'accès



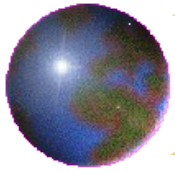
Filtrage niveau 2

- Sont généralement réalisés par éléments actifs du réseau (Commutateurs, Commutateurs routeurs)
 - VLAN (IEEE 802.1Q) :
 - Par port physique
 - Par adresse MAC
 - Par adresse IP
 - Par utilisateur (802.1x)
 - Trames Ethernet broadcast



Filtrage niveau 3

- Réalisé sur les routeurs
- Sur les adresses IPv4, IPv6, multicast
 - Anti-spoofing en entrée et en sortie
 - Adresses broadcast .255 .0 (fonction du subnet-mask)
 - RFC 1918 (adresses privées et non routées)
 - Adresses réservées
- Tunnels IP (GRE, protocole 47)
- VPN, IPsec
- Référence sur les réseaux réservés :
 - <http://www.cymru.com/Bogons/index.html>



Filtrage niveau 4

- Réalisé sur les routeurs, firewall et serveurs
- Protocoles
 - ICMP (type et code),
 - UDP (port source, destination)
 - TCP
 - Ports (source et destination)
 - Flags (SYN, ASK, RST, PSH, URG, FIN)
 - Routage BGP, RIP, OSPF...
- Filtres plus ou moins complexes :
 - Filtrage sans état (analyse des paquets indépendamment les uns des autres)
 - Utilisation de firewall « statefull inspection » (suivi de l'état)
 - Suivi des connexions déjà établies
 - Diminution complexité des règles de filtrage



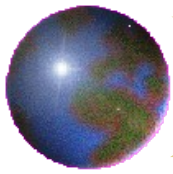
Principe du filtrage dynamique

- On ne laisse rien rentrer (sauf vers nos serveurs)
- On contrôle le trafic sortant pour en déduire ce qu'il faut laisser entrer
- On garde dans une table toutes les nouvelles connexions (valides) sortantes jusqu'à une fin de session ou bien un timeout
 - Pour TCP : la connexion établie après le 3way handshake
 - Pour UDP s'il est le premier paquet sortant contenant une adresse et un port non récent
- On autorise un paquet à rentrer que si la connexion est déjà dans la table
- L'entrée est retirée de la table à la fin de session pour TCP ou bien sur timeout pour les autres protocoles (UDP, ICMP)
- Support des protocoles appliquant des créations de canaux multiple résultants d'une négociation dans un canal de contrôle (ftp , h323...)



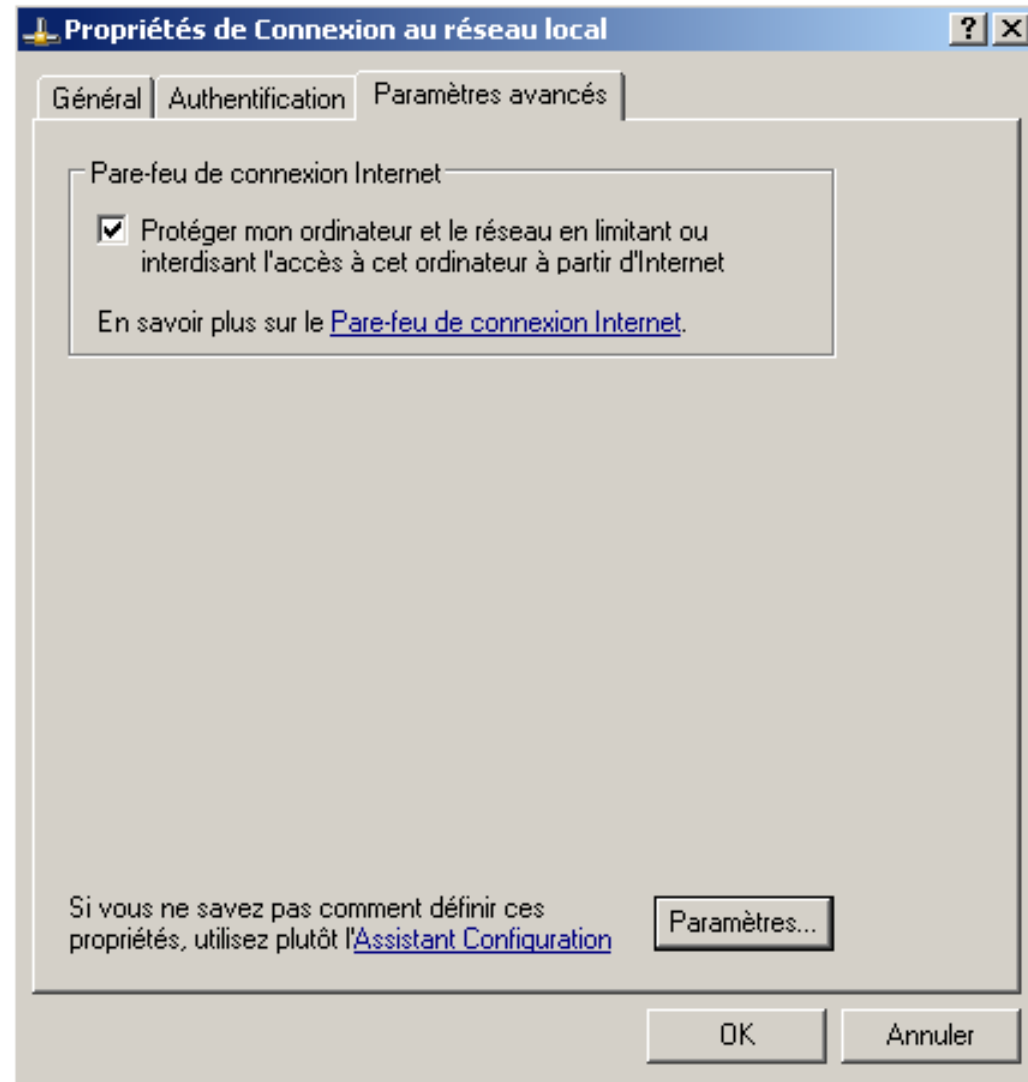
Filtrage niveau applicatif

- En fonction des spécifications des protocoles
- Proxies (mandataires) applicatifs
- Proxies « circuits » : SOCKS
- Exemples de proxies applicatifs :
 - FTP,
 - H323,
 - Proxy http (filtrage d'URL, javascript, cookies)



Filtrage IP sous Windows XP SP1

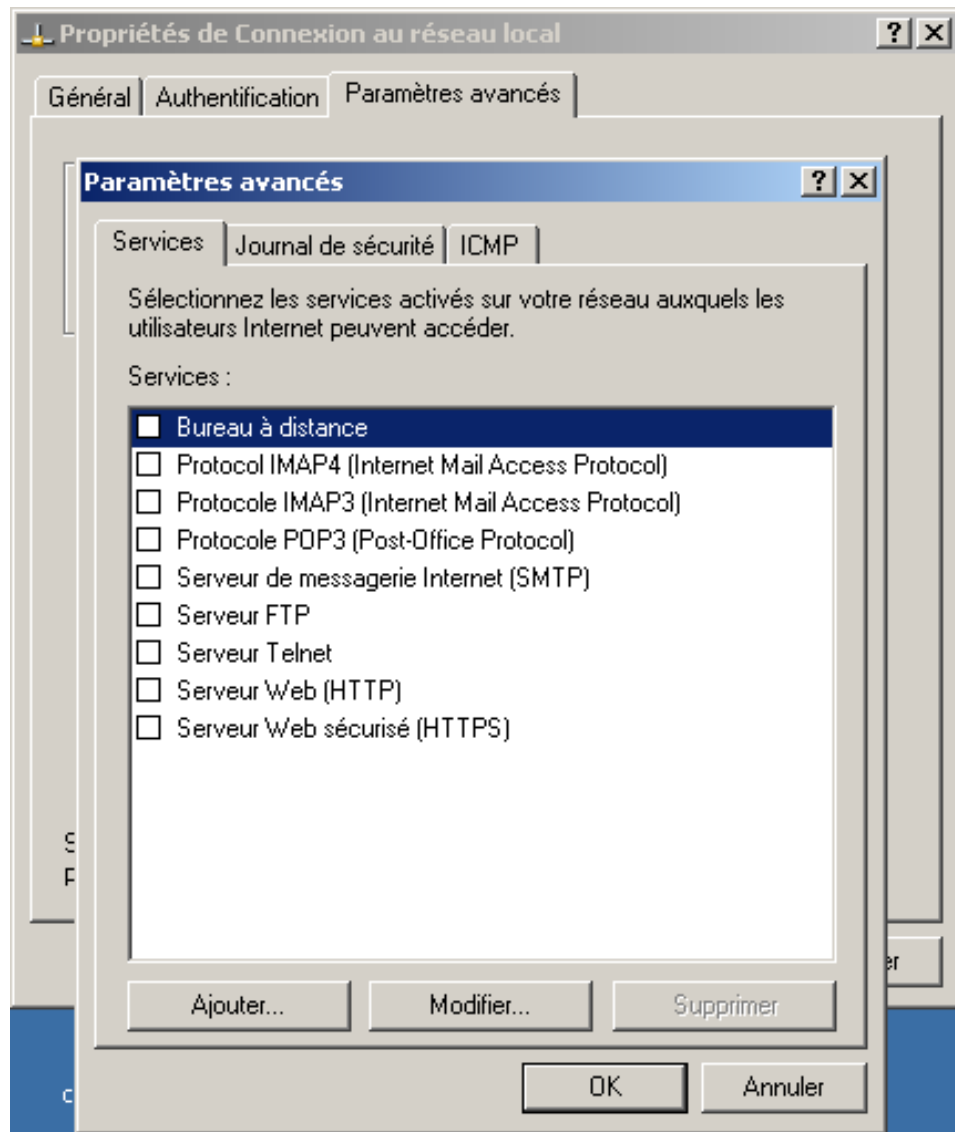
- 2 systèmes de filtrage
 - « à la W2000 »
 - « à la WXP » : Internet Connection Firewall
- ICF : Apparu avec WXP :
 - Propriété de connexion au réseau local
 - Onglet « Paramètres avancés »
 - Paramètres...





ICF de Windows XP (et XP SP1)

- Selon Microsoft : à mémoire d'état : « stateful inspection » : laisse passer le trafic entrant issu d'une connexion initiée de l'ordinateur
- Services prédéfinis
- Possibilité d'ajouter des filtres sur de nouveaux services
- Possibilité d'enregistrer dans un journal les paquets rejetés (journal de sécurité)...
- Possibilité de filtrer les paquets TCP, UDP et ICMP
- Pas de filtrage applicatif (ports ouverts dynamiquement)
- Pas de filtrage par processus





ICF suite...

- ICF et partage de connexion Internet (ICS)
 - ICF sert de firewall pour tous les ordinateurs du réseau local
 - MS Exchange et Office2000/Outlook : problème car utilise les RPC (c'est le serveur qui avertit les clients d'un nouveau message)
- ICF et journalisation
 - Peut enregistrer traces sur trafic autorisé et trafic stoppé
 - Taille maximale du journal paramétrable
 - Format (Extended Log File Format) défini par le W3C



Windows XP SP2

- Remplacement d'ICF par « Windows Firewall »
- Rappel : Sous XP SP1 : ICF désactivé par défaut
- Sous XP SP2 :
 - Windows Firewall activé par défaut sur toutes les interfaces
 - Possibilité d'appliquer des filtres globaux sur toutes les connexions
 - Exceptions :
 - basées sur les applications (nom de l'exécutable)
 - basées sur les adresses réseau (IPv4, IPv6, réseau IP local)
 - Possibilité de bloquer temporairement toutes les exceptions
 - Intégration au « group policy »
- Ajout d'une option (-b) à netstat : processus écoutant sur les ports
`netstat -b <=> netstat -o + tasklist /svc`
- Commande netsh : affichage de l'état du firewall (règles...)
`netsh firewall show state enable`



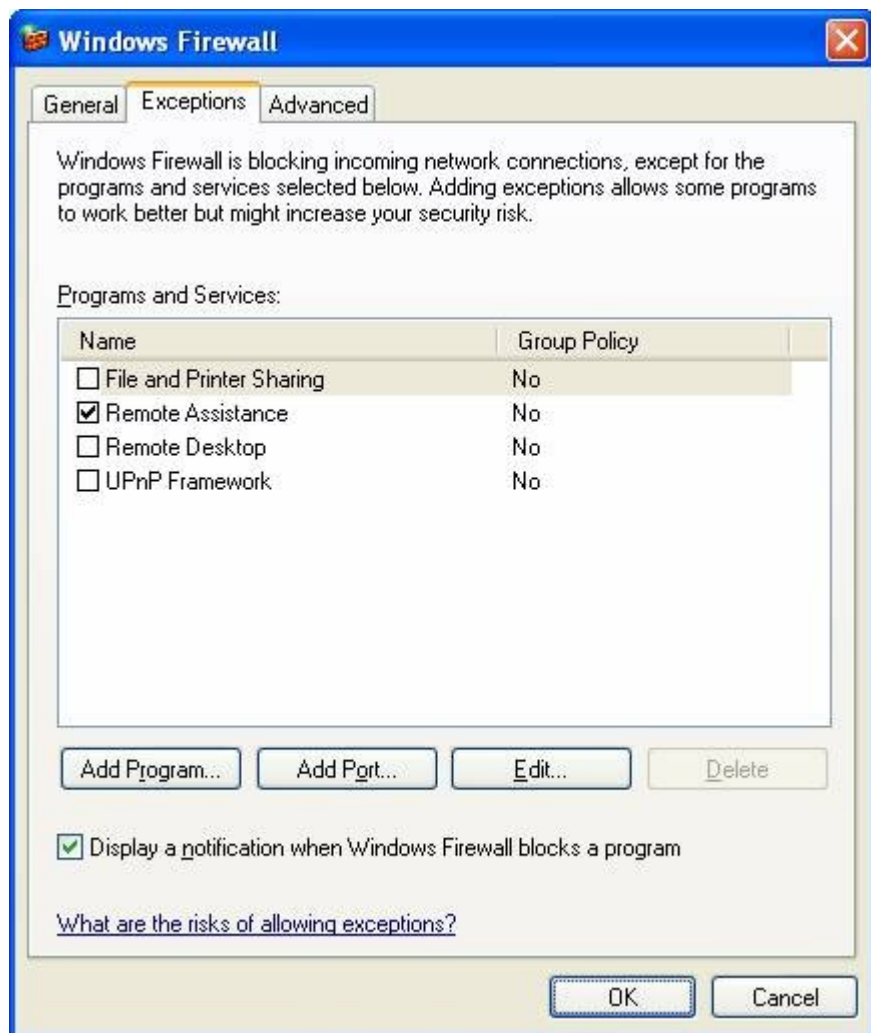
Windows XP SP2 : onglet général



- **“On”** : tout le trafic entrant est bloqué, sauf pour les exceptions
- **“Don't allow exceptions”** : tout le trafic entrant est bloqué
- **“Off”** : « Windows firewall » est désactivé



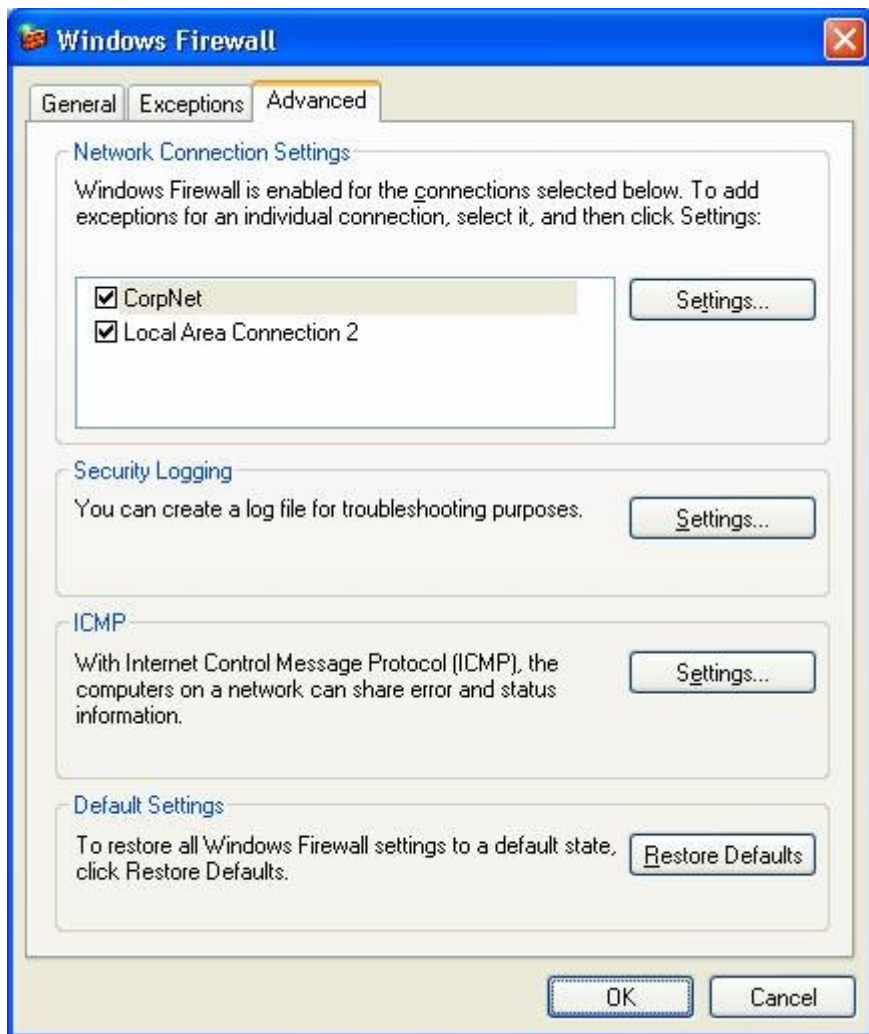
Windows XP SP2 : onglet exceptions



- On peut activer/désactiver
 - Un programme
 - Un port
 - Une combinaison des 2



Windows XP SP2 : onglet Advanced



- « Network connection settings » :
 - On peut activer/désactiver la protection sur une interface particulière
 - Dans settings : filtrage ICMP
- « Security logging » : traçage des connexions : refusées et/ou acceptées
- Pas de filtrage des connexions sortantes !



Filtrage Unix

- Linux : plusieurs implémentations
 - Netfilter (iptables)
 - Ipfiler (ipfw)
- *BSD via ipfilter (ipfw)
- Solaris , AIX , HP-UX, IRIX ... via ipfilter ou bien netfilter
- IRIX, fourni avec le système : ipfilterd



Filtrage Linux

- Linux : plusieurs implémentations en fonction du noyau
 - Ipfwadm : noyau 2.0
 - 3 destinations pour un paquet : accept, deny, reject
 - Directions : in, out, both
 - Règles : input, output, forwarding, masquerading
 - Filtrage sur @IP/port/interface source ou destination
 - Masquage (masquerading) des connexions sur @IP/port/interface source ou destination
 - Ipchains : noyau 2.2
 - 6 destinations par défaut (accept, deny, reject, masq, redirect, return), création d'autres destinations
 - Support port forwarding
 - Création de chaînes (maintenance plus facile)
 - Support QOS, support négation (!)

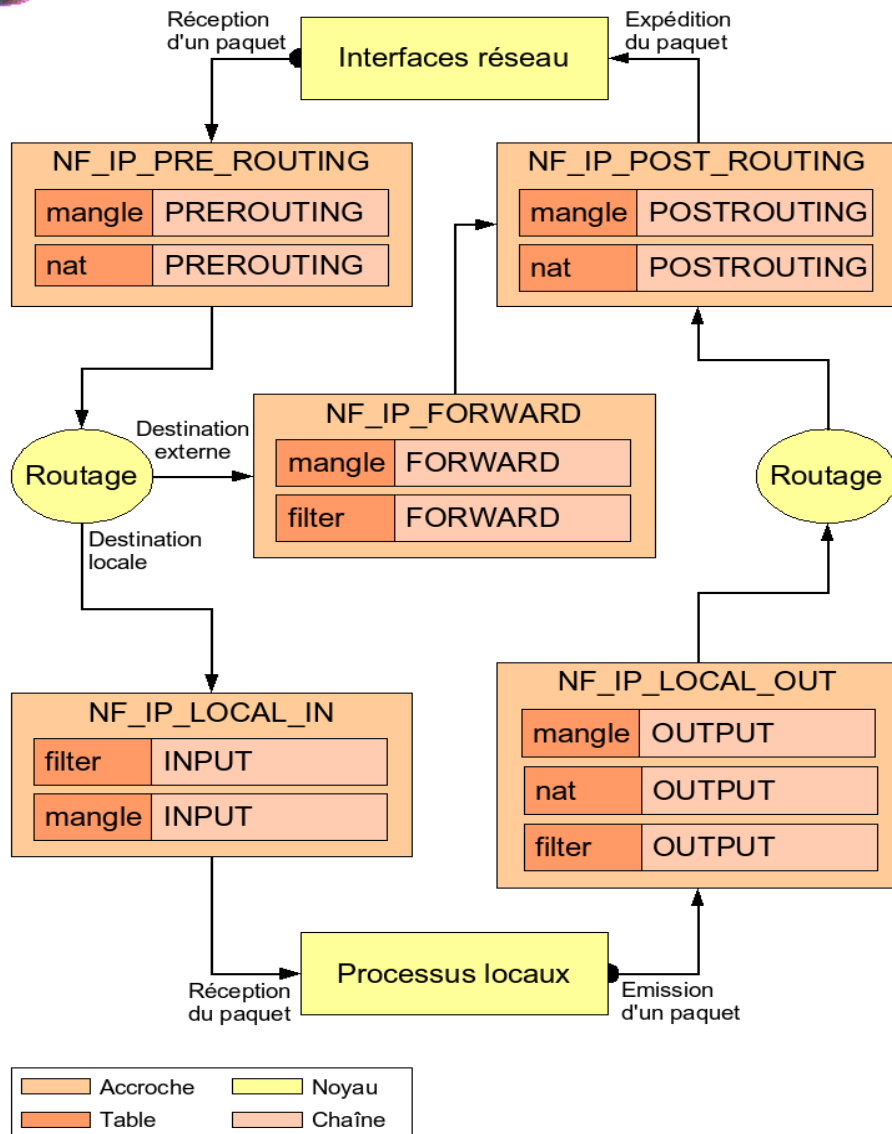


Filtrage Linux (suite)

- Iptables/netfilter : noyau 2.4/2.6
 - destinations : ACCEPT, DROP, QUEUE, RETURN, REJECT, TOS, MIRROR, MASQ, MARK, DNAT, SNAT, REDIRECT), LOG
 - Tables :
 - Chaines pré-définies : INPUT, FORWARD, OUTPUT
 - Nat : PREROUTING, OUTPUT, POSTROUTING
 - mangle
 - Support des 6 drapeaux TCP : SYN, ACK, FIN, URG, RST, PSH
 - Support filtrage par adresse MAC
 - Support de plusieurs ports (source ou destination) dans une seule règle
 - Stateful inspection (suivi de connexion). Les règles peuvent préciser le type de connexion : NEW, RELATED, INVALID, ESTABLISHED
 - Limitation du nombre (rate limiting) contre les deni-de-service
 - Modules spécifiques pour le support applicatif (ftp, h323, SIP, tftp...)
- Interfaces graphiques ou générateurs de règles :
firestarter, shorewall, fwbuilder, guarddog, narc, smoothwall, gShield...



Processus de filtrage dans le kernel



Lorsqu'un paquet arrive sur la machine où fonctionne un noyau Linux, une décision de routage est prise :

- si le paquet n'est pas destiné à un processus local, il passe dans la chaîne FORWARD si le routage est activé, il est détruit sinon ;
- si le paquet est destiné à un processus local, il passe dans la chaîne INPUT ;
- Lorsqu'un processus local émet un paquet, celui-ci passe dans la chaîne OUTPUT.



Construction d'une règle Iptables

- Commande : -A --add, -D --delete, -I --insert, -R --replace, -L --list
- Chaîne d'application : -A INPUT,OUTPUT,FORWARD,chaîne
- État : -m state --state NEW
- Sélection de ce qu'il faut filtrer : UDP, TCP, ICMP, ports, codes, adresse(s) IP, source, destination
- Décision : -j ACCEPT

```
iptables -A [INPUT,OUTPUT,FORWARD,chaîne]
[-p protocol] [-s source] [-d destination] [-i interface] [-o interface]
[--sport port] [--dport port] [-m multiports --sports port[,port[,port...]] --dports port[,port[,port...]]
[--tcpflags SYN ACK FIN RST URG PSH ALL NONE] [-m state --state NEW, INVALID, ESTABLISHED, RELATED]
[--icmp-type number]
[-m limit --limit n/s]
-j (ACCEPT,REJECT,DROP,LOG,chaîne)
```

protocol : tcp, udp, icmp, or all

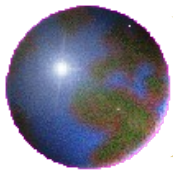
source : address[/mask] : mask numérique ou masque réseau

destination : address[/mask] : mask numérique ou masque réseau



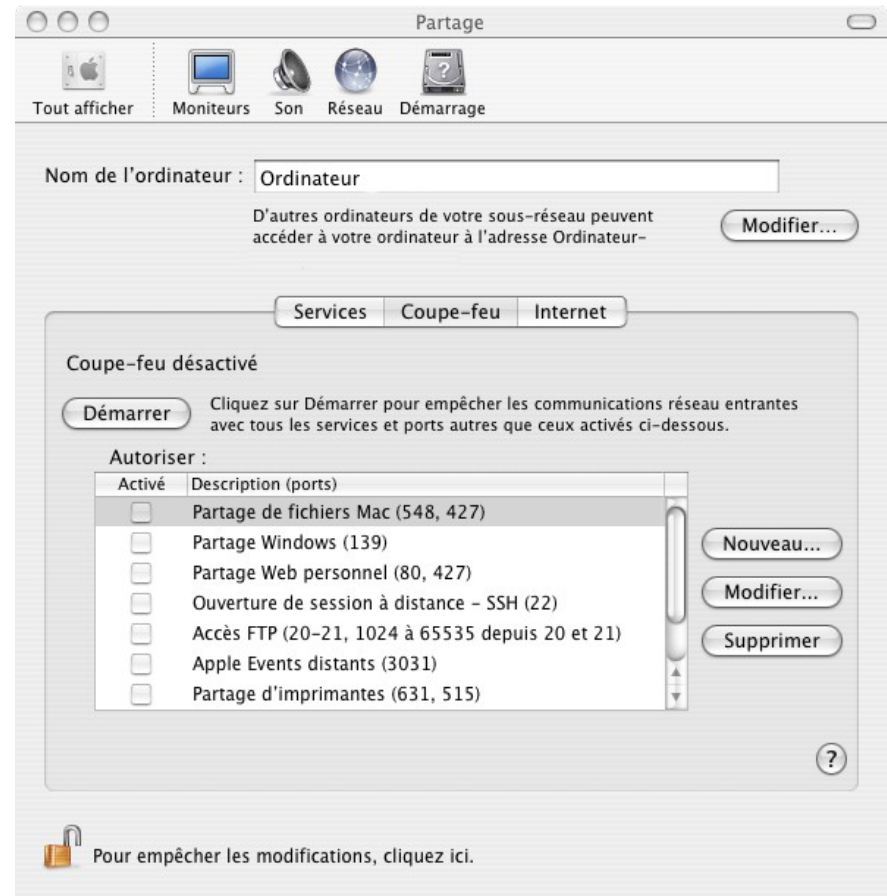
Exemple de filtrage Iptables sous Linux

```
#!/bin/sh
# INPUT
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -p icmp -m limit --limit 1/s --icmp-type echo-request -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -m state --state NEW -m udp -p udp --dport 161 -s 192.168.2.1 -j ACCEPT
iptables -A INPUT -m state --state INVALID,NEW -j LOG
iptables -A INPUT -m state --state INVALID,NEW -j DROP
# OUTPUT
iptables -A OUTPUT -o lo -j ACCEPT
# pour les connexions deja etablies
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -m state --state NEW -m udp -p udp --dport 53 -d 192.168.21.12 -j ACCEPT
# icmp
iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT
# on reject le reste
iptables -A OUTPUT -j REJECT --reject-with icmp-host-prohibited
```



Filtrage IP sous MacOS

- MacOS .../7/8/9 : pas de filtrage IP au niveau système
- MacOS X : firewall intégré au système, application graphique :
- Services pré-déterminés
 - Pas de choix sur les protocoles : TCP, UDP, ICMP
 - Pas de filtrage des connexions sortantes
- Possibilité d'ajouter des autorisations (connexions entrantes) sur les applications
- Possibilité d'enregistrer des traces





Ipfw (MacOS X): fonctionnement

- MacOS X : ipf (ipfilter) issu des *BSD

- Ouvrir un shell
- Nécessité d'être root (commande su ou sudo) ou administrateur
- Commandes Unix : exemple : bloquer « Appleshare IP » (référence article Apple TIL 'Ports Numbers used by AppleShare IP')

```
# ipfw -f flush
# ipfw add 200 reject tcp from any to any 548
# ipfw add 201 reject tcp from any to any 427
```

- Liste des règles :

```
# ipfw list
```

- Effacer les règles :

```
# ipfw -f flush
```

- Ajouter les règles :

```
# ipfw add ...
```

Syntaxe : <rule-number> <allow or deny> <protocol> from <source> to <destination> <options>

- Effacer les règles :

```
# ipfw delete ...
```

- Exemple de règle par défaut :

```
# ipfw add 65535 deny ip from any to any
```

- Règles prises en compte de 0 à 65535 (par ordre croissant).
- Le traitement de la chaîne des règles s'arrête quand la règle courante correspond au paquet courant



Ipfw (MacOS X) : exemples

- Accès au serveur web local depuis localhost, tout autre trafic vers le serveur web est stoppé :

```
# ipfw -f flush
# ipfw add 100 allow tcp from 127.0.0.1 to any 80
# ipfw add 101 reject tcp from any to any 80 in via en0
```

- To any : fonctionne quelle que soit l'adresse IP (destination) de la machine (utile quand ip dynamique)
- Autoriser accès Appleshare IP depuis un seul client (192.168.0.140) :

```
# ipfw -f flush
# ipfw add 301 allow tcp from 192.168.0.140 to any 548
# ipfw add 302 allow tcp from 192.168.0.140 to any 427
# ipfw add 303 reject tcp from any to any 548
# ipfw add 304 reject tcp from any to any 427
```

- Accès au service telnet depuis un ensemble d'adresses IP :

```
# ipfw -f flush
# ipfw add 400 allow tcp from 192.168.0.0:255.255.255.0 to any 23
# ipfw add 401 reject tcp from any to any 23
```

- NB : Trafic vers autres services autorisé



Ipfw (MacOS X) : exemple de script complet restrictif

```
#!/bin/sh

IPFW=/sbin/ipfw
# interface ethernet : en0, ppp : ppp0
INTERFACE=en0

# efface toutes les regles
${IPFW} -f flush
# interface Local loopback est ouverte
${IPFW} add 1000 allow ip from any to any via lo0
${IPFW} add 1001 deny all from any to 127.0.0.0/8
# autorise le trafic sortant
${IPFW} add 2000 pass tcp from any to any out via ${INTERFACE}
# autorise connexions TCP deja ouvertes
${IPFW} add 3000 pass tcp from any to any established
# Autorise les fragments IP de passer
${IPFW} add 4000 pass all from any to any frag
# Autorise les requêtes et réponses DNS
${IPFW} add 5000 allow udp from any to any 53 out via ${INTERFACE}
${IPFW} add 5001 allow udp from any 53 to any in via ${INTERFACE}
# Arrete tout trafic hors ICMP
${IPFW} add 8000 deny tcp from any to any via ${INTERFACE}
${IPFW} add 8000 deny udp from any to any via ${INTERFACE}
```

- Très restrictif :
 - Gène les flux FTP en mode actif
 - Gène MS Netmeeting et Quicktime/RealPlayer (mode RTP)



Ipfw (MacOS X) : exemple de script complet moins restrictif

```
#!/bin/sh

IPFW=/sbin/ipfw
INTERFACE=en0
# efface toutes les règles
${IPFW} -f flush

# Autorise le trafic sortant
${IPFW} add 2000 pass tcp from any to any out via ${INTERFACE}

# Autorise les connexions TCP déjà ouvertes
${IPFW} add 3000 pass tcp from any to any established

# Autorise les fragments IP à passer
${IPFW} add 4000 pass all from any to any frag

# Stoppe le trafic vers les services sur les ports réservés
${IPFW} add 8000 deny tcp from any to any 1-1023 in via ${INTERFACE}
${IPFW} add 8000 deny udp from any to any 1-1023 in via ${INTERFACE}
```

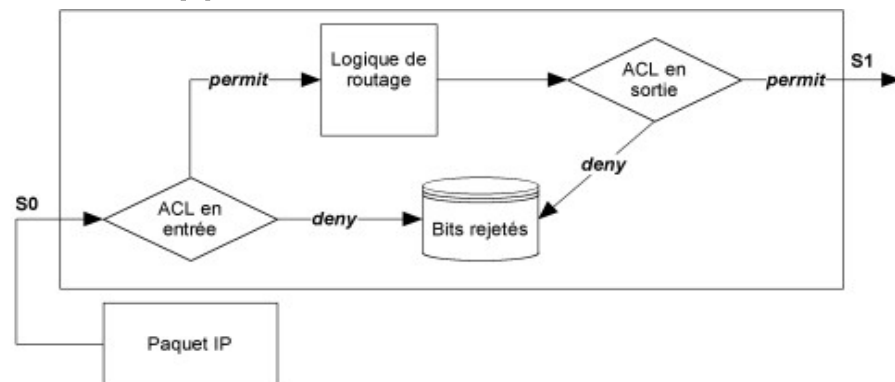
- Autorise trafic :
 - ftp actif
 - Udp
 - Icmp



Les ACL de Cisco

- Les ACL (Access Control List ou access-list) sont définies par un numéro (ou un nom) et peuvent être de plusieurs types :

```
routeur(config)#access-list ?  
<1-99>      IP standard access list  
<100-199>   IP extended access list  
<1100-1199> Extended 48-bit MAC address access list  
<200-299>   Protocol type-code access list  
<700-799>   48-bit MAC address access list
```



- IP standard access list : Ne permet d'utiliser que les adresses source pour identifier les paquets
- IP extended access list : Permet d'identifier un paquet par les adresses IP, protocoles et ports source et destination
- Protocol type-code access list : Filtrage sur le protocole
- 48-bit MAC address access list : Filtrage en fonction de l'adresse MAC



Syntaxe des ACL étendues de Cisco

```
access-list access-list-number {deny | permit} protocol source source-wildcard destination destination-wildcard  
[precedence precedence] [tos tos] [fragments] [log] [log-input] [time-range time-range-name] [dscp dscp]
```

```
access-list access-list-number {deny | permit} tcp source source-wildcard [operator port]  
destination destination-wildcard [operator port] [established] [precedence precedence] [tos tos]  
[fragments] [log] [log-input] [time-range time-range-name] [dscp dscp] [flag]
```

```
access-list access-list-number {deny | permit} icmp source source-wildcard destination destination-wildcard [icmp-type |  
[[icmp-type icmp-code] | [icmp-message]] [precedence precedence] [tos tos] [fragments] [log] [log-input] [time-range  
time-range-name] [dscp dscp]
```

- *access-list-number* : 100 to 199 or 2000 to 2699

- *protocol*, enter the name or number of an IP protocol: **ahp**, **eigrp**, **esp**, **gre**, **icmp**, **igmp**, **igrp**, **ip**, **ipinip**, **nos**, **ospf**, **pcp**, **pim**, **tcp**, or **udp**, or an integer in the range 0 to 255 representing an IP protocol number. To match any Internet protocol (including ICMP, TCP, and UDP) use the keyword **ip**.

- *source*, *source-wildcard*, *destination*, and *destination-wildcard* :

any for 0.0.0.0 255.255.255.255 (**any host**).

The keyword **host** for a single host 0.0.0.0

- operators include **eq** (equal), **gt** (greater than), **lt** (less than), **neq** (not equal), and **range** (inclusive range).

Operators require a port number (**range** requires two port numbers separated by a space).

- *flag* : Enter one of these flags to match by the specified TCP header bits: **ack** (acknowledge), **fin** (finish), **psh** (push), **rst** (reset), **syn** (synchronize), or **urg** (urgent).

- *icmp-type* : Enter to filter by ICMP message type, a number from 0 to 255.

- *icmp-code* : Enter to filter ICMP packets that are filtered by the ICMP message code type, a number from 0 to 255.

- *icmp-message* : Enter to filter ICMP packets by the ICMP message type name or the ICMP message type and code name



Caractéristiques des access-list Cisco

- Les masques de sous-réseau se notent « à l'envers » chez Cisco.
- Il est possible d'utiliser :
 - log : qui permet de journaliser le paquet matché
 - Log-input : Journalise en plus l'interface et l'adresse MAC.
- Il est possible d'appliquer l'access-list à différentes entités, par exemple:
 - Une interface (physique ou vlan), en précisant le sens d'application : entrée et/ou sortie (par rapport au routeur)
 - Communauté SNMP : Par communauté (RO, RW), une telle ACL permet de limiter aux machines d'administration et de supervision l'accès à SNMP.
- established : qui correspond aux drapeaux ACK ou RST : Il s'agit d'un simple filtrage du sens de l'établissement de session, established ne vérifie en rien l'appartenance d'un paquet à une session TCP, il ne fait que contrôler que les flags ASK ou RST sont bien positionnée.



Exemple de filtrage avec L'IOS Cisco

```
interface Vlan1
description reseau test1 (default)
ip address 192.168.1.1 255.255.255.0
ip access-group 101 out
!
interface Vlan2
description reseau test2
ip address 192.168.2.1 255.255.255.0
ip access-group 102 out
!
! permet le ping sur tout le réseau
! permet NTP et ssh
! bloque et journalise (sauf le HTTP) tout le reste
access-list 102 remark ACL d'exemple
access-list 102 permit icmp any 192.168.2.0 0.0.0.255 echo
access-list 102 permit udp any eq ntp host 192.168.2.4 eq ntp
access-list 102 permit tcp any host 192.168.2.4 eq 22
access-list 102 remark Do not log HTTP request (codered, nimda, whatever)
access-list 102 deny tcp any any eq 80
access-list 102 deny ip any any log
```



Bibliographie

- Note d'information CERTA : Filtrage et pare-feux
<http://www.certa.ssi.gouv.fr/site/CERTA-2006-INF-001.pdf>
- Netfilter : www.netfilter.org
- Ipfiler : <http://coombs.anu.edu.au/~avalon>
- Iptables tutorial :
<http://iptables-tutorial.frozentux.net/iptables-tutorial.html>
- Windows XP SP2 : « The cable guy 01 et 02/2004 » :
<http://www.microsoft.com/technet/community/columns/cableguy/cg0104.mspix>
<http://www.microsoft.com/technet/community/columns/cableguy/cg0204.mspix>
- Configuring IP access-lists :
<http://www.cisco.com/application/pdf/paws/23602/confaccesslists.pdf>