# syntaxe des commandes utilisant les regexp

- Objet : boite à outils des expressions régulières
- Niveau requis :
  avisé
- Commentaires : *Contexte d'utilisation du sujet du tuto.*
- Débutant, à savoir : Utiliser GNU/Linux en ligne de commande, tout commence là !.

## Synthèse grep

```
grep [options] regexp [fichier...]
```

Voir : caractères utilisés dans les expressions régulières étendues

| | |
|---|---|
| -c | afficher le décompte des lignes correspondantes |
| -i | ignorer la case |
| -E | utiliser les regexp étendues<br>(correspond à egrep) |
| -o | afficher uniquement les parties (non vides) correspondantes des lignes sélectionnées, chaque partie étant affichée sur une ligne séparée. |

> Deux utilisations:
>
> Soit `grep [options] “expression” /chemin/fichier` (on applique grep sur un fichier)
> Soit `cmd | grep [options]` (on travaille à partir d'un flux d'entrée avec un filtre (pipe)

#### Sur un fichier

```
grep -E "(:[0-9]{4}:){1}" /etc/passwd
```

```
hypathie:x:1000:1000:Hypathie,,,:/home/hypathie:/bin/bash
```

#### Filtre un flux d'entrée

```
/sbin/ifconfig | grep -E "[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}"
```

```
          inet adr:192.168.0.22  Bcast:192.168.0.255  Masque:255.255.255.0
          inet adr:127.0.0.1  Masque:255.0.0.0
```

```
/sbin/ifconfig | grep -oE "[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}"
```

```
127.0.0.1
255.0.0.0
192.168.0.21
192.168.0.255
```

```
255.255.255.0
```

Attention de ne pas oublier `-o` pour afficher l'occurrence exacte d'un mot au lieu de la ligne entière où figure l'occurrence du mot !

```
echo "bfer aaa jhgao aaaaaa haug aaaaaaaa" | grep -E
"[[:blank:]][a]{3}[[:blank:]]"
```

```
bfer aaa jhgao aaaaaa haug aaaaaaaa
```

```
echo "bfer aaa jhgao aaaaaa haug aaaaaaaa" | grep -oE
"[[:blank:]][a]{3}[[:blank:]]"
```

```
 aaa
```

# Synthèse sed

### Options

| **options** | **significations** |
|---|---|
| `-e` | enchaîner plusieurs commandes |
| `-r` | utiliser les expressions régulières étendues dans un script |
| `-n` | mode silencieux : permet de ne rien modifier<br>associée au drapeau p (print)<br>(affichage sur la sortie standard) |
| `-f` | Les commandes sont lues à partir d'un fichier préalablement rédigé. |
| `-i` | Le fichier est édité sur place. |

### Commandes de sed

| | |
|---|---|
| d et D | supprimer |
| q | quitter |
| p et P | afficher avec `-n` |
| i\texte | insérer du texte |
| a\texte | ajouter du texte |
| c\texte | remplacer du texte |
| = | afficher |
| : | label |
| # | commentaire |
| {....} | Bloc |
| b | label (section) |
| g et G | obtenir |
| h et H | tenir |
| n et N | après |

| | |
|---|---|
| r *fichier* | lire *fichier* |
| s/…./…./ | substitution |
| t | test |
| w *fichier* | écrire *fichier* |
| x | eXchange |
| y/…./…./ | translittération |

#### Les drapeaux de la commande sed s/…/…/

| | |
|---|---|
| g | global : toutes les occurrences |
| 1, 2, etc. | un nombre : la nième occurrence |
| w | écrire les modifications effectuées dans un fichier |
| p | afficher la ligne modifiée |
| e | exécution d'une commande |

# awk

# Détail : ligne de commandes shell et awk

#### Utilisation du pipe

Comme toutes commandes, awk peut traiter la sortie d'un pipe.

- Soit la variable suivante :

```
ligne="mot1 mot2 mot3"
```

- Exemple 1 :

```
echo $ligne | awk -F " " '{print $1}'
```

```
mot1
```

- Exemple 2 :

```
echo $ligne|awk -F " " '{print $1 $2}'
```

```
mot1mot2
```

- Exemple 3 :

```
echo $ligne|awk -F " " '{print $0}'
```

```
mot1 mot2 mot3
```

#### Redirections d'entrée et de sortie

- On peut passer à awk le fichier à traiter avec **<** :

```
awk -F ':' '{ print $1 " est " $5 }' < /etc/passwd
```

- équivalent de :

```
awk -F ':' '{ print $1 " est " $5 }' /etc/passwd
```

- on peut rediriger la sortie de awk vers un fichier :

```
awk '{ print $1 ": moyenne math => " $4 }' fichier-awk.txt > Fichier1-sortie
```

# awk programmation

## Variables utilisateurs

Une variable utilisateur peut aussi bien avoir un type correspondant à une chaîne de caractères ou à un nombre, ou même correspondre au deux types.

- Exemple1 : Soit script “var-u.awk”

```
$1 == "Constance"{
        var=3
        $4=$4+var
        nom=$1 " -Correction note de math :"
        print nom, $4
        }
```

> Pour le fichier “fichier-awk.txt”,
> on donne comme condition la correspondance correcte entre $1 et la chaîne “Constance”, afin que le programme soit exécuté ;
> on crée la variable utilisateur var de valeur 3 ;
> pour modifier $4, on crée la variable $4 de valeur $4+var ;
> on crée la variable nom de valeur : $1 “ -Correction note de math :” (on voit là une variable dont la valeur est un mélange des deux types, numérique et caractère) ;
> on récupère la valeur des variables créées tout simplement avec la fonction print.

```
awk -f var-u.awk fichier-awk.txt
```

```
Constance -Correction note de math : 18
```

## Utilisation de test "if-else"

- Soit le fichier “var-u.txt” :

```
cat var-u.txt
```

```
$1 == "Constance"{
# la condition du programme
        var=3
# première variable crée
        $4=$4+var
# deuxième variable crée
        nom=$1 " -Correction note de math :"
# troisième crée
        print nom, $4
# action du programme
        }
```

- Soit le script “if-else.awk” :

```
BEGIN {
    print "Début du script:"
    variable=0
    commentaire=0
    print " nombre de variable = " variable
    print " nombre de commantaire = " commentaire
    }

/^# / {
commentaire++
}
/^[^#]/ && /[[:graph:]]=[[:graph:]]/{
variable++
}
{ if ( FNR > 1 ) {
  print "Le nombre de lignes commentées est: " commentaire ;
  print "Le nombre de variables est: " variable
  }
  else {
  print "Pas d'enregistrement à traiter; lecture du fichier ..."
  }
}
END {
   print "Fin de traitement du fichier '" FILENAME "'."
   print "Il y a eu " NR " enregistrements courants, " FNR " lignes
traitées."
   }
```

- Application du script “if-else.awk” sur le fichier “var-u.txt” :

```
awk -f if-else.awk var-u.txt
```

```
Début du script:
 nombre de variable = 0
 nombre de commantaire = 0
```

```
Pas d'enregistrement à traiter; lecture du fichier ...
Le nombre de lignes commentées est: 0
Le nombre de variables est: 0
Le nombre de lignes commentées est: 0
Le nombre de variables est: 1
Le nombre de lignes commentées est: 0
Le nombre de variables est: 1
Le nombre de lignes commentées est: 0
Le nombre de variables est: 2
Le nombre de lignes commentées est: 0
Le nombre de variables est: 2
Le nombre de lignes commentées est: 0
Le nombre de variables est: 3
Le nombre de lignes commentées est: 0
Le nombre de variables est: 3
Le nombre de lignes commentées est: 0
Le nombre de variables est: 3
Le nombre de lignes commentées est: 0
Le nombre de variables est: 3
Le nombre de lignes commentées est: 0
Le nombre de variables est: 3
Fin de traitement du fichier 'var-u.txt'.
Il y a eu 11 enregistrements courants, 11 lignes traitées.
hypathie@debian:~/Documents/AWK/Awk-exos$
hypathie@debian:~/Documents/AWK/Awk-exos$ awk -f if-else.awk var-u.txt
Début du script:
 nombre de variable = 0
 nombre de commantaire = 0
Pas d'enregistrement à traiter; lecture du fichier ...
Le nombre de lignes commentées est: 1
Le nombre de variables est: 0
Le nombre de lignes commentées est: 1
Le nombre de variables est: 1
Le nombre de lignes commentées est: 2
Le nombre de variables est: 1
Le nombre de lignes commentées est: 2
Le nombre de variables est: 2
Le nombre de lignes commentées est: 3
Le nombre de variables est: 2
Le nombre de lignes commentées est: 3
Le nombre de variables est: 3
Le nombre de lignes commentées est: 4
Le nombre de variables est: 3
Le nombre de lignes commentées est: 4
Le nombre de variables est: 3
Le nombre de lignes commentées est: 5
Le nombre de variables est: 3
Le nombre de lignes commentées est: 5
Le nombre de variables est: 3
Fin de traitement du fichier 'var-u.txt'.
```

```
Il y a eu 11 enregistrements courants, 11 lignes traitées.
```

# Iptables index

Les données transmises sur un réseau est seulement un flux d'électrons (CAT 6/5e) ou photons (fibre) et RF qui se modifient en amplitude en fréquence. Ces modulations sont régies par les médias sur lesquels ces particules voyagent.
Des exemples les électrons sont absorbés après une période de temps après laquelle ils sont restés dans les conducteurs qui les portent.
Il s'agit là de métrique.

Une certaine quantité du flux de particules devient toujours entropique dans un système fermé(par exemple du fait de la chaleur résiduelle). Le flux peut être altéré par des interférences électromagnétiques (EMI) de ligne électrique, par les ondes des téléphones portables et la connexion peut être endommagé ainsi que l'équipement.

Pour faire face à ces erreurs, chaque paquet détient une valeur CRC (contrôle de redondance cyclique) qui est calculé par un algorithme avant d'être envoyés. Lorsque les paquets arrivent à destination la somme de contrôle (checksum) est calculée. Si elle correspond aux données de la source, le paquet est bon. Si le paquet est mauvais il est renvoyé.

Lorsque plusieurs périphériques transmettent des messages sur un réseau en même temps, différents protocoles doivent être respectées et appliquées à ces paquets sur ce réseau. Par exemple, CSMA/CD régit “collisions”. C'est à la couche 3 du modèle OSI, que les routeurs acheminent des paquets à différents sous-réseaux en fonction de leurs données, par exemple selon les protocoles RIP et OSPF.

## Détail du vocabulaire

### Chaîne

Une chaîne est une suite de règles ordonnées.
Chaque chaîne peut être comparée à un ensemble de tests, chacun ayant pour résultat l’envoi du paquet vers la **cible** spécifiée si la condition est vérifiée. Si ce n’est pas le cas, on passe à la suivante. En arrivant à la fin d’une des chaînes, une cible par défaut est utilisée.

- **Les types de chaînes** :

| **chaîne** | **signification** |
|---|---|
| PREROUTING | Chaîne qui stoppe le paquet qui n'a subit aucune modification par rapport à ce que l'interface réseau a reçu. Il peut y être modifier pour de NAT. |
| INPUT | les paquets venant vers le PC : à ce stade, le paquet est prêt à être envoyé aux couches applicatives, c'est à dire aux serveurs et aux clients qui tournent sur la machine. |
| FORWARD | Chaîne qui permet aux paquets de passer d'une interface à une autres du PC (il devient un routeur), mais n'est pas livré à la couche applicative. |
| OUTPUT | Chaîne qui permet aux paquets de quitter le PC après que les couches applicatives ont été traitées. |

| chaîne | signification |
|---|---|
| POSTROUTING | La décision de routage a été prise. Les paquets entrent dans cette chaîne, juste avant qu'ils soient transmis vers le matériel. |

### Cible

Action effectuée en cas de validation d'une règle.
Voici les cibles prédéfinies les plus courantes :

- **Types de cibles** :

| Cible | Description |
|---|---|
| ACCEPT | Les paquets envoyés vers cette cible seront tout simplement acceptés et pourront poursuivre leur cheminement au travers des couches réseaux. |
| DROP | Cette cible permet de jeter des paquets qui seront donc ignorés. |
| REJECT | Permet d'envoyer une réponse à l'émetteur pour lui signaler que son paquet a été refusé. |
| LOG | Demande au noyau d'enregistrer des informations sur le paquet courant. Cela se fera généralement dans le fichier /var/log/messages (selon la configuration du programme syslogd). |
| MASQUERADE | Cible valable uniquement dans la chaîne POSTROUTING de la table NAT. Elle change l'adresse IP de l'émetteur par celle courante de la machine pour l'interface spécifiée. Cela permet de masquer des machines et de faire par exemple du partage de connexion. |
| SNAT | Egalement valable pour la chaîne POSTROUTING de la table NAT seulement. Elle modifie aussi la valeur de l'adresse IP de l'émetteur en la remplaçant par la valeur fixe spécifiée. |
| DNAT | Valable uniquement pour les chaînes PREROUTING et OUTPUT de la table NAT. Elle modifie la valeur de l'adresse IP du destinataire en la remplaçant par la valeur fixe spécifiée. |
| RETURN | Utile dans les chaînes utilisateurs. Cette cible permet de revenir à la chaîne appelante. Si RETURN est utilisé dans une des chaînes de base précédente, cela est équivalent à l'utilisation de sa cible par défaut. |

### Table

On peut se représenter une table comme ce qui gère les paquets grâce à un regroupement de chaînes, elles-mêmes composées de règles ; c'est ce qui permet de contrôler les paquets arrivant et sortant.
Il existe pour trois tables (Filter, NAT et Mangle).

En bref, une table applique aux paquets plusieurs chaînes

| TABLE | USAGE | CHAÎNES |
|---|---|---|
| FILTER | Filtrage des paquets réseaux ; mise en place des règles du pare-feu.<br>Tout ce qui n'est pas explicitement autorisé doit être strictement interdit.<br>En pratique, on commence par mettre à DROP les 3 chaînes FORWARD, INPUT et OUTPUT.<br>Puis on autorise que certains flux bien particuliers. | INPUT<br>contrôle les paquets à destination des applications<br>OUTPUT<br>analyse les paquets qui sortent des applications<br>FORWARD<br>filtrage des paquets qui passent d'une interface réseau à l'autre. Les paquets de ce type ne passent jamais par les chaînes INPUT et OUTPUT |
| NAT | Translation d'IP : elle permet de transformer un PC en routeur | PREROUTING<br>Les paquets vont être modifiés à l'entrée de la pile réseaux, qu'ils soient à destination des processus locaux où d'une autre interface<br>POSTROUTING<br>Les paquets sortant des processus locaux sont modifiés.<br>OUTPUT<br>les paquets qui sont près à être envoyés aux interfaces réseaux sont modifiés. |
| MANGLE | Modification des paquets : permet à Linux d'avoir un contrôle sur les débits des flux de données entrants et sortants de la machine, afin de rendre certains flux plus prioritaires que d'autres | PREROUTING<br>INPUT<br>FORWARD<br>OUTPUT<br>POSTROUTING |
| RAW | principalement utilisée pour placer des marques sur les paquets qui ne doivent pas être vérifiés par le système de traçage de connexion.<br>(nécessite le module iptable_raw) | PREOUTING<br>OUTPUT |

Voir http://www.admin-debian.com/securite/iptables-et-netfilter/

## script client simple

### commandes pour flush

```
iptables -F
iptables -X
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT
```

### liste de commandes testées et fonctionnelles pour futur script pare-feu user

```
iptables -F
iptables -X
```

```
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

iptables -t filter -A OUTPUT -p udp -m udp\
 --dport 53 -m conntrack --ctstate NEW,RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -A INPUT -p udp -m udp\
 --sport 53 -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -A OUTPUT -o lo -j ACCEPT
iptables -t filter -A INPUT -i lo -j ACCEPT

iptables -A OUTPUT -p icmp -m conntrack\
 --ctstate NEW,ESTABLISHED,RELATED -j ACCEPT

iptables -A INPUT -p icmp -m conntrack\
 --ctstate ESTABLISHED,RELATED -j ACCEPT

iptables -t filter -A OUTPUT -p tcp -m multiport\
 --dports 80,443,8000 -m conntrack --ctstate\
 NEW,RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -A INPUT -p tcp -m multiport\
 --sports 80,443,8000 -m conntrack --ctstate\
 RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -N OutGoingSSH
iptables -I INPUT -p tcp --dport 10015 -j OutGoingSSH
iptables -A OutGoingSSH -j LOG --log-prefix '[OUTGOING_SSH] : '

iptables -t filter -N InComingSSH
iptables -I OUTPUT -p tcp --sport 10015 -j InComingSSH
iptables -A InComingSSH -j LOG --log-prefix '[INCOMING_SSH] : '

iptables -t filter -A INPUT -i eth0 -s 192.168.0.0/24\
 -p tcp -m tcp --dport 10015 -m conntrack\
  --ctstate NEW,ESTABLISHED -j ACCEPT

iptables -t filter -A OUTPUT -o eth0\
 -p tcp -m tcp --sport 10015 -m conntrack\
 --ctstate ESTABLISHED -j ACCEPT

iptables -A OUTPUT -o eth0 -p tcp --dport 10015 -j ACCEPT

iptables -t filter -A OUTPUT -o eth0 -p tcp -m tcp --dport 10015 -m
conntrack\
  --ctstate NEW,ESTABLISHED -j ACCEPT

iptables -t filter -A INPUT -i eth0 -s 192.168.0.0/24 -p tcp --sport 10015 -
```

```
m conntrack\
 --ctstate ESTABLISHED -j ACCEPT
```

## commandes bash pour script passerelle (fin)

```
iptables -F
iptables -X
iptables -t nat -F
iptables -t nat -X
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT

iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
iptables -t nat -P INPUT ACCEPT
iptables -t nat -P OUTPUT ACCEPT
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE

#Maintenant que tout est à DROP il faut s'occuper de la boucle local
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# Et notre interface interne :
iptables -A INPUT -i eth1 -j ACCEPT
iptables -A OUTPUT -o eth1 -j ACCEPT

#On garde nos règles concernant le DROP sur FORWARD (FILTER)
#mais on oublie pas eth1 !
iptables -t filter -A FORWARD -i eth1 -o eth0 -s 192.168.1.0/24\
 -d 0.0.0.0/0 -p tcp -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

iptables -t filter -A FORWARD -i eth0 -o eth1 -s 0.0.0.0/0\
 -d 192.168.1.0/24 -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -t filter -A FORWARD -p icmp -j ACCEPT

iptables -t filter -A INPUT -p icmp -i eth0 -m conntrack\
 --ctstate ESTABLISHED,RELATED -j ACCEPT

iptables -t filter -A OUTPUT -p icmp -o eth0 -m conntrack\
 --ctstate ESTABLISHED,RELATED -j ACCEPT

iptables -t filter -A INPUT -p icmp -i eth1 -m conntrack\
 --ctstate ESTABLISHED,RELATED -j ACCEPT
```

```
iptables -t filter -A OUTPUT -p icmp -o eth1 -m conntrack\
 --ctstate ESTABLISHED,RELATED -j ACCEPT

# Et on prend soin de laisser entrer et sortir (INPUT, OUTPUT de FILTER)
# le flux nécessaire au DNS (53) et au web (80, 443..)
# et là encore on n'oublie pas eth1

iptables -t filter -A OUTPUT -o eth0 -p udp -m udp --dport 53\
 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -A INPUT -i eth0 -p udp -m udp --sport 53\
 -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -A OUTPUT -o eth1 -p udp -m udp --dport 53\
 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -A INPUT -i eth1 -p udp -m udp --sport 53\
 -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -A OUTPUT -o eth0 -p tcp -m multiport --dports\
 80,443,8000 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -A INPUT -i eth0 -p tcp -m multiport --sports\
 80,443,8000 -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -A OUTPUT -o eth1 -p tcp -m multiport --dports 80,443,8000 -j
ACCEPT
iptables -A INPUT -i eth1  -p tcp -m multiport --sports 80,443,8000 -j
ACCEPT



# Les règles imcp pour OUTPUT et INPUT
#(en y intégrant celles de testées pour FORWARD)

iptables -A OUTPUT -p icmp --icmp-type 0 -j ACCEPT
iptables -A INPUT -p icmp --icmp-type 0 -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type 0 -j ACCEPT

iptables -A INPUT -p icmp --icmp-type 3/4 -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 3/4 -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type 3/4 -j ACCEPT

iptables -A FORWARD -p icmp --icmp-type 3/3 -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 3/3 -j ACCEPT
iptables -A INPUT -p icmp --icmp-type 3/3 -j ACCEPT

iptables -A FORWARD -p icmp --icmp-type 3/1 -j ACCEPT
iptables -A INPUT -p icmp --icmp-type 3/1 -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 3/1 -j ACCEPT
```

```
iptables -A INPUT -p icmp --icmp-type 4 -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 4 -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type 4 -j ACCEPT

iptables -A INPUT -p icmp --icmp-type 8 -m limit --limit 2/s -j ACCEPT
iptables -A INPUT -p icmp --icmp-type 8 -j LOG --log-prefix "ICMP/in/8
Excessive: "
iptables -A INPUT -p icmp --icmp-type 8 -j DROP
iptables -A OUTPUT -p icmp --icmp-type 8 -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type 8 -j ACCEPT

iptables -A INPUT -p icmp --icmp-type 11 -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 11 -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type 11 -j ACCEPT

iptables -A INPUT -p icmp --icmp-type 12 -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 12 -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type 12 -j ACCEPT

iptables -A FORWARD -s 192.168.1.0/24 -d 192.168.0.0/24 -p icmp\
 --icmp-type echo-request -j ACCEPT
#pour le retour nous utilison la derniere regle
iptables -A FORWARD -s 192.168.0.0/24 -d 192.168.1.0/24 -p icmp\
 --icmp-type echo-reply -j DROP

iptables -A INPUT -p icmp -m limit -j LOG --log-prefix "ICMP/IN: "
iptables -A OUTPUT -p icmp -m limit -j LOG --log-prefix "ICMP/OUT: "


#TCP_BAD

iptables -N syn_flood
iptables -I INPUT -p tcp --syn -j syn_flood
iptables -A syn_flood -m limit --limit 1/s --limit-burst 3 -j RETURN
iptables -A syn_flood -j LOG --log-prefix '[SYN_FLOOD] : '
iptables -A syn_flood -j DROP

# SSH
iptables -t filter -N InComingSSH
iptables -I INPUT -i eth0 -s 192.168.0.0/24 -p tcp -m tcp\
 --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j InComingSSH
iptables -A InComingSSH -j LOG --log-prefix '[INCOMING_SSH] : '
iptables -A InComingSSH -j ACCEPT

iptables -t filter -A OUTPUT -o eth0 -p tcp -m tcp\
 --sport 22 -m conntrack --ctstate ESTABLISHED -j ACCEPT

iptables -t filter -A OUTPUT -o eth1 -p tcp -m tcp\
 --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

iptables -t filter -A INPUT -i eth1 -s 192.168.0.0/24 -p tcp\
```

```
 --sport 22 -m conntrack --ctstate ESTABLISHED -j ACCEPT

#FTP_IN

iptables -N ftp_in_accept
iptables -I INPUT -i eth0 -p tcp --sport 21 -m state\
 --state ESTABLISHED,RELATED -j ftp_in_accept

iptables -I INPUT -i eth0 -p tcp --sport 20 -m state\
 --state ESTABLISHED,RELATED -j ftp_in_accept

iptables -I INPUT -i eth0 -p tcp --sport 1024:65535 --dport\
 1024:65535 -m state --state ESTABLISHED -j ftp_in_accept

iptables -A ftp_in_accept -p tcp -j ACCEPT



iptables -A INPUT -i eth1 -p tcp --sport 21 -m state\
 --state ESTABLISHED,RELATED -j ACCEPT
iptables -A NPUT -i eth1 -p tcp --sport 20 -m state\
 --state ESTABLISHED,RELATED -j ACCEPT
iptables -I INPUT -i eth1 -p tcp --sport 1024:65535 --dport\
 1024:65535 -m state --state ESTABLISHED -j ACCEPT
```

# iptables : qqs compléments

## Masquerade

iptables [-t table] [Action] [Options Valeur] [Cible]

iptables -t nat -A POSTROUTING -p tcp –destination-port smtp -j MASQUERADE
iptables -t NAT -A POSTROUTING -d machin.distant -j MASQUERADE
iptables -t NAT -A POSTROUTING -d machin.distant -p udp –dport domain -j MASQUERADE

## DHCP

### Commandes pour récupérer les IP de la passerelle

```
INET_IP=`ifconfig $INET_IFACE | grep inet | \
cut -d : -f 2 | cut -d ' ' -f 1 | awk 'NR==3{print $1}'`
```

Puis

```
echo $INET_IP
192.168.1.1
```

DHCP fonctionne sur le protocole UDP. Donc, on n'autorise que les ports UDP utilisés par DHCP, qui sont les ports 67 et 68.
Ensuite, il faut autoriser l'ouverture de ces ports uniquement sur l'interface de la passerelle susceptible de recevoir les requêtes DHCP.
En d'autres termes, on autorisera le flux sur les 67 et 68 pour l'interface eth1 qui relie le réseau B à la passerelle. Par exemple, si votre interface eth0 est activée par DHCP, vous n'autoriserez pas les requêtes DHCP sur eth1. Pour rendre la règle un peu plus précise, Ce sont les critères que nous choisissons pour sélectionner les paquets, et que nous autorisons. Ce qui donne ceci :

- Dans une script

```
IPTABLES  -I INPUT -i ${LAN_IFACE} -p udp --dport 67:68 --sport \
     67:68 -j ACCEPT
```

- En ligne de commande :

```
IPTABLES  -I INPUT -i 192.168.1.1 -p udp --dport 67:68 --sport \
     67:68 -j ACCEPT
```

### Et le DHCP ?

Rassurez-vous nous n'avons pas oublié d'ouvrir les UDP pour que le serveur DHCP installé sur cette passerelle puisse continué d'attribuer des IP à notre réseau B.

- Ne pas ajouter de règle aveuglément !

Voici une autre maxime qu'il faut appliquer.

On lit dans de nombreux wiki l'ajout de cette règle :
`IPTABLES -I INPUT -i 192.168.1.1 -p udp –dport 67:68 –sport 67:68 -j ACCEPT`

Nous ne l'ajouterons pas !

- Dans notre script nous avons mis :

```
iptables -A INPUT -i eth1 -j ACCEPT
iptables -A OUTPUT -o eth1 -j ACCEPT
```

Et nous allons à voir si une règle est à ajouter avec l'outil tcpdump.

### Utiliser tcpdump

- On l'installe :

```
apt-get install tcpdump
```

- Puis on lance sur le serveur :

```
tcpdump -i eth1 port 67 or port 68
```

Rien ne se passe, c'est normal : pour voir quelque chose, il faut générer du flux depuis le client du réseau B.
On laisse le terminal en état d'attente, et on passe sur le client

- Côté client DHCP (ordinateur B) :

-On commence par annuler le bail DHCP.
Comme ceci :

```
dhclient -r eth0
```

-Il n'y a plus d'IP :

```
eth0      Link encap:Ethernet  HWaddr 00:1e:0b:67:9b:b7
          adr inet6: xxxxxxxxxxxx Scope:Lien
```

- On lance une requête DHCP

```
dhclient eth0
```

Observez ce que nous dit “tcpdump” au lancemant de la commande suivante sur l'ordinateur B (client DHCP) :

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
07:47:00.535348 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP,
Request from xx:xx:xx:xx:xx:xx (oui Unknown), length 300
07:47:00.535553 IP debian-serveur.mondomaine.hyp.bootps > debian-
hp.local.bootpc: BOOTP/DHCP, Reply, length 302
07:47:00.535745 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP,
Request from xx:xx:xx:xx:xx:xx (oui Unknown), length 300
07:47:00.626390 IP debian-serveur.mondomaine.hyp.bootps > debian-
hp.local.bootpc: BOOTP/DHCP, Reply, length 302
^C
4 packets captured
4 packets received by filter
```

- Et en plus l'IP de l'ordinateur B a changé !

```
INET_IP=`ifconfig $INET_IFACE | grep inet | \
cut -d : -f 2 | cut -d ' ' -f 1` && echo $INET_IP
```

```
192.168.1.4 127.0.0.1
```

# Sauvegarde des règles iptables avec systemd

Nous allons utiliser cette fois systemd pour utiliser “service” au lieu d'init.d

### Prérequis : sauvegarde des règles du pare-feu

Pour ce faire nous allons créer deux fichiers de sauvegarde iptables ;

- un pour “service start” et “service reload”
- un pour “service stop”

### Préparation de "service stop"

- Exécuter cette suite de commandes :

```
iptables -F
iptables -X
iptables -t nat -F
iptables -t nat -X
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

```
iptables-save > /etc/iptables-gateway-flush
```

### Préparation de "service start" et de "service reload"

- Exécuter ce script après l'avoir téléchargé :

Un script parce que ça fait beaucoup de commandes à copier/coller et coller le bloc entier dans une indigestion à bash.

Ne pas le lancé sur le serveur depuis un client en ssh.

script-iptables-passerelle

```
#!/bin/sh

/sbin/iptables -F
/sbin/iptables -X
/sbin/iptables -P INPUT DROP
/sbin/iptables -P OUTPUT DROP
/sbin/iptables -P FORWARD DROP
/sbin/iptables -t nat -P PREROUTING ACCEPT
/sbin/iptables -t nat -P POSTROUTING ACCEPT
/sbin/iptables -t nat -P INPUT ACCEPT
/sbin/iptables -t nat -P OUTPUT ACCEPT
/sbin/iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
##commenter / décommenter et adapter les quatre lignes suivantes pour
ne pas mettre en place / mettre en place
##un proxy transparent (squid)
/sbin/iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 -j DNAT -
```

```
-to 192.168.0.1:3129
/sbin/iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j
REDIRECT --to-port 3129
/sbin/iptables -t mangle -A PREROUTING -p tcp --dport 3128 -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --dport 3129 -j DROP
#accepter l'interface lo
/sbin/iptables -A INPUT -i lo -j ACCEPT
/sbin/iptables -A OUTPUT -o lo -j ACCEPT
#accepter le sous-réseau
/sbin/iptables -A INPUT -i eth1 -j ACCEPT
/sbin/iptables -A OUTPUT -o eth1 -j ACCEPT
#permettre le passage entre les deux interfaces eternet de la
passerelle
/sbin/iptables -t filter -A FORWARD -i eth1 -o eth0 -s 192.168.1.0/24 -
d 0.0.0.0/0 -p tcp -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -t filter -A FORWARD -i eth0 -o eth1 -s 0.0.0.0/0 -d
192.168.1.0/24 -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -t filter -A FORWARD -p icmp -j ACCEPT
#accepter le ping entre les réseaux locaux
/sbin/iptables -t filter -A INPUT -p icmp -i eth0 -m conntrack --
ctstate ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -t filter -A OUTPUT -p icmp -o eth0 -m conntrack --
ctstate ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -t filter -A INPUT -p icmp -i eth1 -m conntrack --
ctstate ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -t filter -A OUTPUT -p icmp -o eth1 -m conntrack --
ctstate ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -A OUTPUT -p icmp --icmp-type 0 -j ACCEPT
/sbin/iptables -A INPUT -p icmp --icmp-type 0 -j ACCEPT
/sbin/iptables -A FORWARD -p icmp --icmp-type 0 -j ACCEPT
/sbin/iptables -A INPUT -p icmp --icmp-type 3/4 -j ACCEPT
/sbin/iptables -A OUTPUT -p icmp --icmp-type 3/4 -j ACCEPT
/sbin/iptables -A FORWARD -p icmp --icmp-type 3/4 -j ACCEPT
/sbin/iptables -A FORWARD -p icmp --icmp-type 3/3 -j ACCEPT
/sbin/iptables -A OUTPUT -p icmp --icmp-type 3/3 -j ACCEPT
/sbin/iptables -A INPUT -p icmp --icmp-type 3/3 -j ACCEPT
/sbin/iptables -A FORWARD -p icmp --icmp-type 3/1 -j ACCEPT
/sbin/iptables -A INPUT -p icmp --icmp-type 3/1 -j ACCEPT
/sbin/iptables -A OUTPUT -p icmp --icmp-type 3/1 -j ACCEPT
/sbin/iptables -A INPUT -p icmp --icmp-type 4 -j ACCEPT
/sbin/iptables -A OUTPUT -p icmp --icmp-type 4 -j ACCEPT
/sbin/iptables -A FORWARD -p icmp --icmp-type 4 -j ACCEPT
/sbin/iptables -A INPUT -p icmp --icmp-type 8 -m limit --limit 2/s -j
ACCEPT
/sbin/iptables -A INPUT -p icmp --icmp-type 8 -j LOG --log-prefix
"ICMP/in/8 Excessive: "
/sbin/iptables -A INPUT -p icmp --icmp-type 8 -j DROP
/sbin/iptables -A OUTPUT -p icmp --icmp-type 8 -j ACCEPT
/sbin/iptables -A FORWARD -p icmp --icmp-type 8 -j ACCEPT
/sbin/iptables -A INPUT -p icmp --icmp-type 11 -j ACCEPT
```

```
/sbin/iptables -A OUTPUT -p icmp --icmp-type 11 -j ACCEPT
/sbin/iptables -A FORWARD -p icmp --icmp-type 11 -j ACCEPT
/sbin/iptables -A INPUT -p icmp --icmp-type 12 -j ACCEPT
/sbin/iptables -A OUTPUT -p icmp --icmp-type 12 -j ACCEPT
/sbin/iptables -A FORWARD -p icmp --icmp-type 12 -j ACCEPT
/sbin/iptables -A FORWARD -s 192.168.1.0/24 -d 192.168.0.0/24 -p icmp -
-icmp-type echo-request -j ACCEPT
/sbin/iptables -A FORWARD -s 192.168.0.0/24 -d 192.168.1.0/24 -p icmp -
-icmp-type echo-reply -j DROP
/sbin/iptables -A INPUT -p icmp -m limit -j LOG --log-prefix "ICMP/IN:
"
/sbin/iptables -A OUTPUT -p icmp -m limit -j LOG --log-prefix
"ICMP/OUT: "
/sbin/iptables -N syn_flood
/sbin/iptables -I INPUT -p tcp --syn -j syn_flood
/sbin/iptables -A syn_flood -m limit --limit 1/s --limit-burst 3 -j
RETURN
/sbin/iptables -A syn_flood -j LOG --log-prefix '[SYN_FLOOD] : '
/sbin/iptables -A syn_flood -j DROP
#autoriser la connexion avec les serveurs DNS
/sbin/iptables -t filter -A OUTPUT -o eth0 -p udp -m udp --dport 53 -m
state --state NEW,RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -t filter -A INPUT -i eth0 -p udp -m udp --sport 53 -m
state --state RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -t filter -A OUTPUT -o eth1 -p udp -m udp --dport 53 -m
state --state NEW,RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -t filter -A INPUT -i eth1 -p udp -m udp --sport 53 -m
state --state RELATED,ESTABLISHED -j ACCEPT
#autoriser la navigation web
/sbin/iptables -t filter -A OUTPUT -o eth0 -p tcp -m multiport --dports
80,443,8000 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -t filter -A INPUT -i eth0 -p tcp -m multiport --sports
80,443,8000 -m state --state RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -A OUTPUT -o eth1 -p tcp -m multiport --dports
80,443,8000 -j ACCEPT
/sbin/iptables -A INPUT -i eth1  -p tcp -m multiport --sports
80,443,8000 -j ACCEPT
#Si le serveur cups est branché sur un ordinateur du réseau
192.168.0.0/24, par exemple sur 192.168.0.22
# laisser décommenter les deux lignes suivantes :
/sbin/iptables -A INPUT -i eth0 -s 192.168.0.22 -d 192.168.0.1 -p tcp -
-sport 631 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -A OUTPUT -o eth0 -s 192.168.0.1 -d 192.168.0.22 -p tcp
--dport 631 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
#créer une chaîne utilisateur pour les connexion ssh, les loguer et les
accepter
/sbin/iptables -t filter -N InComingSSH
/sbin/iptables -I INPUT -i eth0 -s 192.168.0.0/24 -p tcp -m tcp --dport
22 -m conntrack --ctstate NEW,ESTABLISHED -j InComingSSH
/sbin/iptables -A InComingSSH -j LOG --log-prefix '[INCOMING_SSH] : '
/sbin/iptables -A InComingSSH -j ACCEPT
```

```
/sbin/iptables -t filter -A OUTPUT -o eth0 -p tcp -m tcp --sport 22 -m
conntrack --ctstate ESTABLISHED -j ACCEPT
/sbin/iptables -t filter -A OUTPUT -o eth1 -p tcp -m tcp --dport 22 -m
conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
/sbin/iptables -t filter -A INPUT -i eth1 -s 192.168.0.0/24 -p tcp --
sport 22 -m conntrack --ctstate ESTABLISHED -j ACCEPT
#créer une chaîne utilisateur pour les connexions ftp, et les accepter
/sbin/iptables -N ftp_in_accept
/sbin/iptables -I INPUT -i eth0 -p tcp --sport 21 -m state --state
ESTABLISHED,RELATED -j ftp_in_accept
/sbin/iptables -I INPUT -i eth0 -p tcp --sport 20 -m state --state
ESTABLISHED,RELATED -j ftp_in_accept
/sbin/iptables -I INPUT -i eth0 -p tcp --sport 1024:65535 --dport
1024:65535 -m state --state ESTABLISHED -j ftp_in_accept
/sbin/iptables -A ftp_in_accept -p tcp -j ACCEPT
/sbin/iptables -A INPUT -i eth1 -p tcp --sport 21 -m state --state
ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -A INPUT -i eth1 -p tcp --sport 20 -m state --state
ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -I INPUT -i eth1 -p tcp --sport 1024:65535 --dport
1024:65535 -m state --state ESTABLISHED -j ACCEPT

exit 0
```

- Lui ajouter droits d'exécution et compagnie ...

```
chmod +x /etc/init.d/firewall-client.sh
chmod 755 /etc/init.d/firewall-client.sh
```

- L'exécuter : adapter la commande suivante en fonction de chemin du fichier téléchargé

Par exemple :

```
bash /home/hypathie/script-iptables-passerelle
```

```
iptables-save > /etc/iptables-gateway
```

#### Mise en place de service systemd

- Il faut créer un fichier dans le répertoire “cd /etc/systemd/system/”:

```
vim /etc/systemd/system/iptables.service
```

```
[Unit]
Description=Firewall et NAT
After=network.target

[Service]
```

```
Type=oneshot
RemainAfterExit=yes
ExecStart=/bin/sh -c "/sbin/iptables-restore < /etc/iptables-gateway"
ExecReload=/bin/sh -c "/sbin/iptables-restore < /etc/iptables-gateway"
ExecStop=/bin/sh -c "/sbin/iptables-restore < /etc/iptables-gateway-flush"

[Install]
WantedBy=multi-user.target
```

**cp /etc/systemd/system/iptables.service /lib/systemd/system/**

**Prise en compte par systemD**

- On installe

```
apt-get install systemd
```

- Puis on charge son fichier

```
systemctl -f enable iptables.service
```

- Et maintenant :

```
systemctl start iptables.service
```

# Example squid conf

```
acl to_localhost dst 127.0.0.0/8 0.0.0.0/32 ::1

#acl localnet src 10.0.0.0/8     # RFC1918 possible internal network
#acl localnet src 172.16.0.0/12  # RFC1918 possible internal network
#acl localnet src 192.168.0.0/16 # RFC1918 possible internal network
acl LAN src 132.5.210.0/24 # network local
acl SSL_ports port 443
acl SSL_ports port 10000
acl SSL_ports port 631
acl SSL_ports port 563
acl SSL_ports port 873
# acl badsite url_regex "/etc/squid/squid-deny.acl"

acl Safe_ports port 80 # https
acl Safe_ports port 21 # webmin
acl Safe_ports port 443 # Cups
acl Safe_ports port 70 # snews
acl Safe_ports port 210 # rsync
acl Safe_ports port 1025-65535 # http
acl Safe_ports port 280 # ftp
acl Safe_ports port 488 # https
acl Safe_ports port 591 # gopher
```

```
acl Safe_ports port 777 # wais
acl Safe_ports port 631 # unregistered ports
acl Safe_ports port 873 # http-mgmt
acl Safe_ports port 901 # gss-http
acl purge method PURGE # filemaker
acl CONNECT method CONNECT # multiling http
http_access allow manager localhost
http_access deny manager
http_access allow purge localhost
http_access deny purge
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
icp_access allow localhost

http_access allow LAN
http_access deny all

# http_access deny badsite
icp_access deny all

http_port 132.5.210.16:3128 transparent
hierarchy_stoplist cgi-bin ?
cache_mem 512 MB
maximum_object_size_in_memory 1024 KB
minimum_object_size 0 KB
maximum_object_size     2048 KB
cache_dir ufs /var/spool/squid3 4096 16 256
access_log /var/log/squid3/access.log squid
refresh_pattern ^ftp:           1440    20%     10080
refresh_pattern ^gopher:        1440    0%      1440
refresh_pattern -i (/cgi-bin/|\?) 0     0%      0
refresh_pattern (Release|Package(.gz)*)$        0       20%     2880
refresh_pattern .               0       20%     4320

hosts_file /etc/hosts
pipeline_prefetch on
shutdown_lifetime 3 second
```

TCP_REFRESH_UNMODIFIED/304

- cache directories

new cache directory :

```
/home/hypathie/cache/spool/squid3/
```

→

```
/mnt/proxy/cache/spool/squid3
```

- logs directory :

```
/var/log/squid3/access.log
```

→

```
/mnt/proxy/log/squid3/access.log
```

```
tail -f /mnt/proxy/log/squid3/access.log
```

ou

```
tail -f /var/log/squid3/access.log
```

owner : proxy not root

- cache_store_log

```
/mnt/proxy/cache_store_log/store.log
```

→

```
/home/hypathie/cache/spool/cache_store_log/store.log
```

- cache.log

```
/var/log/squid3/cache.log
```

→

```
/mnt/proxy/log/squid3/cache.log
```

only “TCP_MISS/200” never “TCP_HIT”

**Cache hit.** Un Cache hit a lieu chaque fois que Squid est en mesure de satisfaire une requête Web depuis son cache. Les paramètres tels que cache hit ratio et le cache hit rate donne le pourcentage de requêtes ayant satisfaits au hit. On considère généralement qu'un ratio compris dans l'intervalle [30%, 60%] est un bon ratio. Un autre paramètre, le byte hit ratio donne le volume de données délivré depuis le cache, en octets [byte].

**Cache miss.** Un Cache miss a lieu chaque fois que Squid n'est pas en mesure de satisfaire une requête Web depuis son cache. Cela peut survenir pour nombre de raisons :

-l'objet sollicité l'est pour la première fois !
-l'objet sollicité n'est pas cachable ;
-l'objet sollicité est périmé ;
-l'objet sollicité a été purgé du cache pour faire de la place à d'autres objets.

- Voici les règles qui sont suivies pour décider de la mise en cache :

Si des instructions interdisant la mise en cache sont rencontrées (Cache-Control: private), la ressource n'est pas mise en cache.
Si la connexion est sécurisée ou authentifiée, la ressource n'est pas mise en cache. Si aucune directive de validation n'est présente (Last-Modified ou Etag), la ressource n'est pas mise en cache. Autrement, la ressource est considérée comme éligible au cache et Squid décide de la stocker.

# Test config

```
#Définition des ACL
acl all src all
acl manager proto cache_object
acl localhost src 127.0.0.1/32 ::1
acl to_localhost dst 127.0.0.0/8 0.0.0.0/32 ::1

#acl localnet src 10.0.0.0/8 # RFC1918 possible internal network
#acl localnet src 172.16.0.0/12  # RFC1918 possible internal network
#acl localnet src 192.168.0.0/16 # RFC1918 possible internal network

#acl localnet src 192.168.1.0/24 # RFC1918 possible internal network
#acl lan src 192.168.0.1 192.168.1.0/24

acl SSL_ports port 443      # https
acl SSL_ports port 563      # snews
acl SSL_ports port 873      # rsync
acl Safe_ports port 80      # http
acl Safe_ports port 21      # ftp
acl Safe_ports port 443     # https
acl Safe_ports port 70      # gopher
acl Safe_ports port 210     # wais
acl Safe_ports port 1025-65535  # unregistered ports
acl Safe_ports port 280     # http-mgmt
acl Safe_ports port 488     # gss-http
acl Safe_ports port 591     # filemaker
acl Safe_ports port 777     # multiling http
acl Safe_ports port 631     # cups
acl Safe_ports port 873     # rsync
acl Safe_ports port 901     # SWAT
acl purge method PURGE
acl CONNECT method CONNECT

###Nos ACL
#Notre réseau
acl lanhome src 192.168.1.0/255.255.255.0

#Les domaines qui doivent passer par Tor
#google
acl domain_tor dstdomain .google.fr
acl domain_tor dstdomain .google.com
```

```
acl domain_tor dstdomain .googleapis.com
acl domain_tor dstdomain .googleusercontent.com
acl domain_tor dstdomain .recaptcha.net

#facebook
acl domain_tor dstdomain .fbcdn.net
acl domain_tor dstdomain .facebook.com

#yahoo
acl domain_tor dstdomain .yahoo.com
acl domain_tor dstdomain .yimg.com
acl domain_tor dstdomain .yahoo.fr

#torrent
acl domain_tor dstdomain .openbittorrent.com

#On définie le contrôle si c est un post ou pas
acl method_post method POST

#Le contrôle des extentions qui ne doivent pas être mises en cache
acl extention_no_cache url_regex \.iso$ \.mdf$ \.mkv$ \.mp4$ \.wma$ \.mp3$
\.wav$ \.flac$ \.torrent$ \.mpeg$ \.mpg$ \.exe$ \.vbs$ \.msi$ \.avi$ \.php$
\.php5$ \.php4$ \.php3$ \.html$ \.htm$
#Si on veut que certaines extentions passent par tor
acl files_to_tor  url_regex \.js$ \.css$
#Si on veut que certaines extentions NE passent PAS par tor
acl files_NO_tor  url_regex \.flv$ \.avi$ \.mpg$ \.mpeg$ \.wmv$

###L acces HTTP... Utilise donc les ACL définis plus haut. Je m'étend pas
dessus
http_access allow manager localhost
http_access deny manager
# Only allow purge requests from localhost
http_access allow purge localhost
http_access deny purge
# Deny requests to unknown ports
http_access deny !Safe_ports
# Deny CONNECT to other than SSL ports
http_access deny CONNECT !SSL_ports
http_access allow lanhome
http_access allow localhost

# And finally deny all other access to this proxy
http_access deny all

###Les ICP, protocole d'échange entre serveurs de cache
#Allow ICP queries from local networks only
icp_access allow localnet
icp_access deny all
```

```
###Le port d'écoute ! Ici on dit qu'on prend le port 3128, qui écoute notre
adresse.
#Transparent, signifie que le proxy accepte une redirection de port, ainsi
qu'on ne soit pas obligé de spécifier de proxy dans le navigateur
http_port 192.168.2.1:3128 transparent

#Heu...
hierarchy_stoplist cgi-bin ?

#La mémoire utilisée
cache_mem 128 MB
maximum_object_size_in_memory 1 MB

##Vous pouvez modifier l'emplacement des repertoires de log
access_log /var/log/squid/access.log squid
#access_log /home/hypathie/squid/squid_access.log squid

cache_log /var/log/squid/cache.log
#cache_log /home/[user]/squid/cache.log



cache_store_log /var/log/squid/store.log
cache_store_log /home/[user]/squid/store.log

###Durée de cache en seconde.
refresh_pattern -i \.gif$ 10080 150% 43200 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.flv$ 10080 150% 43200 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.js$ 10080 150% 43200 ignore-no-store override-expire
override-lastmod  ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.pdf$ 10080 90% 43200 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.art$ 10080 150% 43200 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache  ignore-must-revalidate
refresh_pattern -i \.avi$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.mov$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.wav$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.mp3$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.qtm$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.mid$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.viv$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.mpg$ 10080 150% 40320 ignore-no-store override-expire
```

```
override-lastmod  ignore-reload ignore-no-cache  ignore-must-revalidate
refresh_pattern -i \.jpg$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod  ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.jpeg$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.png$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.rar$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.ram$ 10080 150% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache  ignore-must-revalidate
refresh_pattern -i \.gif$ 10080 300% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.txt$ 1440 100% 20160 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.zip$ 2880 200% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.arj$ 2880 200% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.exe$ 2880 200% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.tgz$ 10080 200% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.gz$ 10080 200% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate
refresh_pattern -i \.tgz$ 10080 200% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache  ignore-must-revalidate
refresh_pattern -i \.tar$ 10080 200% 40320 ignore-no-store override-expire
override-lastmod ignore-reload ignore-no-cache ignore-must-revalidate

#Suggested default:
refresh_pattern ^ftp:       1440    20% 10080
refresh_pattern ^gopher:    1440    0%  1440
refresh_pattern -i (/cgi-bin/|\?) 0 0%  0
refresh_pattern (Release|Package(.gz)*)$    0   20% 2880

#refresh_pattern (\.deb|\.udeb)$   129600 100% 129600
refresh_pattern .       0   20% 4320

###ICI on peut spécifier un autre serveur DNS si on le souhaite
#dns_nameservers 89.233.43.71 89.104.194.142

##Changer la durée du cache des noms de domaine.
#positive_dns_ttl 48 hours
#negative_dns_ttl 1 minutes

# Don't upgrade ShoutCast responses to HTTP=>Heuuu
acl shoutcast rep_header X-HTTP09-First-Line ^ICY.[0-9]
upgrade_http0.9 deny shoutcast

# Apache to signal ETag correctly on such responses=>Heeuuu
```

```
acl apache rep_header Server ^Apache
broken_vary_encoding allow apache

#   You can add up to 20 additional "extension" methods here.
extension_methods REPORT MERGE MKACTIVITY CHECKOUT

###Les repertoires
hosts_file /etc/hosts
coredump_dir /var/spool/squid
##le nom du proxy...
#visible_hostname not_your_business

#cache_dir ufs /home/[user]/squid/cache 1000 16 256
cache_dir ufs /home/hypathie/cache/spool/squid3/ 100 16 256

#La mémoire, demandez moi pas la différence
memory_pools_limit 256 MB

### Cache
#On interdit de mettre en cache les extensions
#sans caches définis dans les ACL plus haut
#cache deny  extention_no_cache
#cache allow src all
### MULTIPLE CACHE
#Et oui, vous avez vu squid redirige soit vers privoxy un soit vers
privoxy2. Il a donc #deux caches différents.
#On indique qu'il a quelqu'un derrière, il doit pas renvoyer directement le
paquet sur #le net
#prefer_direct off

#never_direct deny SSL_ports
#never_direct allow all

#
```

# Inverser les mots d'une phrase

```
PROCEDURE INVERSER L'ORDRE DES MOTS D'UNE PHRASE TERMINEE PAR UN POINT
_____________________________________________________________________________
_________________


1) Définition du problème :


une phrase      |    |    Une phrase nouvelle phrase dont les
avec point   --------->  |    | --->  mots sont séparés d'un seul espace
écrit à l'envers
```

```
obigatoire      |    |       dont chacun des mots sont écrits à l'envers.



2 ) Jeu d'essai


Phrase                       phrase inversée
-------------------------------------------------------------------
' '.                        ' '
 .                        '.'
"Il fait beau."                          "lI tiaf uaeb."
"Li tiaf uaeb."                    "iL fait beau."
"il    fait  beau."                               "li tiaf uaeb."
"il fait       beau."                             "li tiaf uaeb."
"il fait. beau"                                    "li tiaf."


3)Algo de principe:

Parcourir une phrase terminée par un point pour lui enlever les espaces
inutiles.
Saisir chaque mot avec sa taille sur cette phrase.
Inverser et copier chaque mot de cette phrase dans une nouvelle chaîne.



4) Définition des données



5) Programme de test


6) Programme de la procédure inverser l'ordre des lettres des mots dans une
phrase
   _____________________________
   réflexion préalable sur l'affichage d'un tableau
    ---------------------
    = > comment afficher un tableau dans le bon sens ?

    Enregistrer un tableau de 1 à n (sens du début à la fin)


                               n
     | b | o | n | j | o | u | r |
       1   2   3   4   5   6   7

     Pour i de 1 à n

     cible := Mot[1], cible := 'b' ; pour cible := Mot[2], cible := 'o' ...
```

```
                    // donc pour afficher tous les éléments de tab dans
l'ordre' on fait
                    //  cible := Mot[n], cible := n
                    //  lire (Mot[cible]) de 1 à n
                    // Pour avancer on sait que n suit l'indice i du
tableau, càd, quand n =1 ; i = 1'
                    // Donc ci-dessus revient à : lire (Mot[i]) de 1 à n
                    // Pour avancer : Pour i de 1 à n
                    //                lire (Mot[i]) ; i := i + 1
                  // Ou encore :
           //  TANTQUE i <= n FAIRE
                    //       i := i + 1
                    //       lire(Mot[i])
           //  FINTANTQUE

        Pour lire un tableau en sens inverse de longueur à 1  (par exemple
de 7 à 1)
        Pour i de longueur à 1 (par exemple, longueur =  7 ; on a 'r' )

        i commence à 7 et termine à 1 ; à chaque tour on enleve 1 à i
(i=i-1)

        cible := mot[i] ; par exemple quand i = 4 ; mot[4]='n' ; cible prend
la valeur 'n'

        mot-inverse[longueur - i ] := cible ;ainsi 7-4 =3 donc Mot-
Inverse[3]='n'
                                    n
       | b | o | n | j | o | u | r |     => lire(MotInverse[longueur - i ])
         1   2   3   4   5   6   7
```

---

```
fonction inversMot(entrée chaine : Mot
          entrée long   : entier) : Mot
Type
Mot = tableau[long] de caractères

Variables

long : entier // taille réelle de mot
i : entier // indice de parcours en sens inverse de Mot[long]
MotInverse : Mot // c est le mot inversé

Début
      écrire('saisissez un mot')
      lire(mot, tailleMot)
      i := tailleMot
      long := long(mot)
      Pour i de (long) à 1
         cible := mot[i]
```

```
         motInverse[long - i]
      FinPour
      copier(MotInverse)
Fin

_____________________________
```