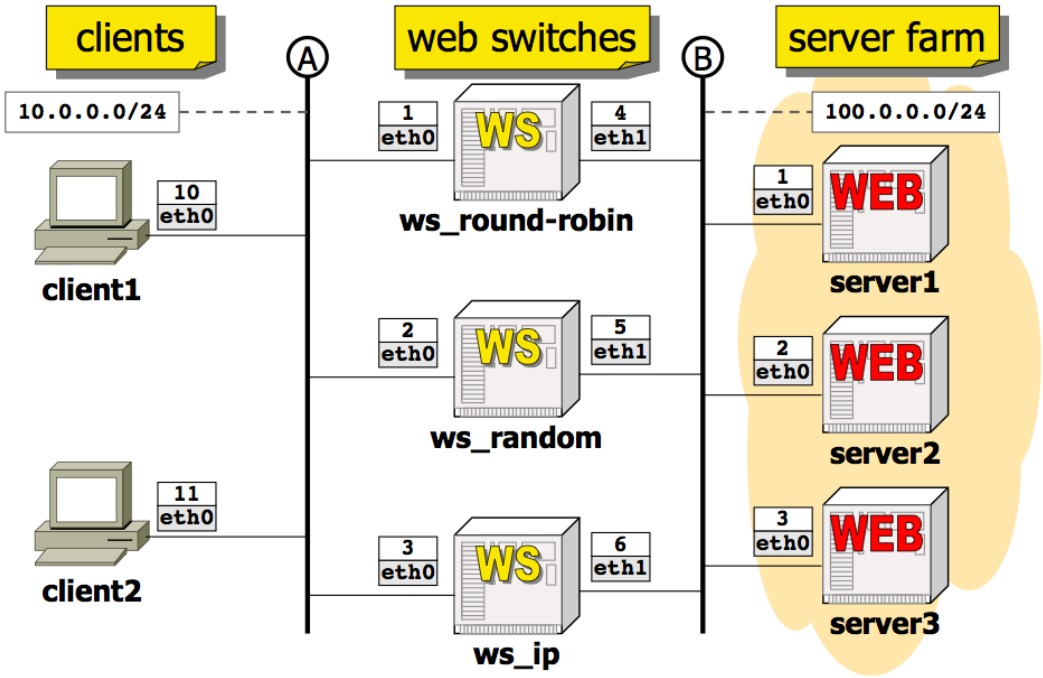
Load Balancing (server farm)

Ecrivez un compte rendu des résultats de votre expérimentation, et répondez aux questions.

http://wiki.netkit.org/netkit-labs/netkit-labs_application-level/netkit-lab_loadbalancer-ws/netkit-lab_loadbalancer-ws.pdf

http://wiki.netkit.org/netkit-labs/netkit-labs_application-level/netkit-lab_loadbalancer-ws/netkit-lab_loadbalancer-ws.tar.gz



servers
- offrent une page HTML (tout à fait simple) par default
- chacun des 3 serveurs a une page différente, pour qu'il puisse être facilement individué

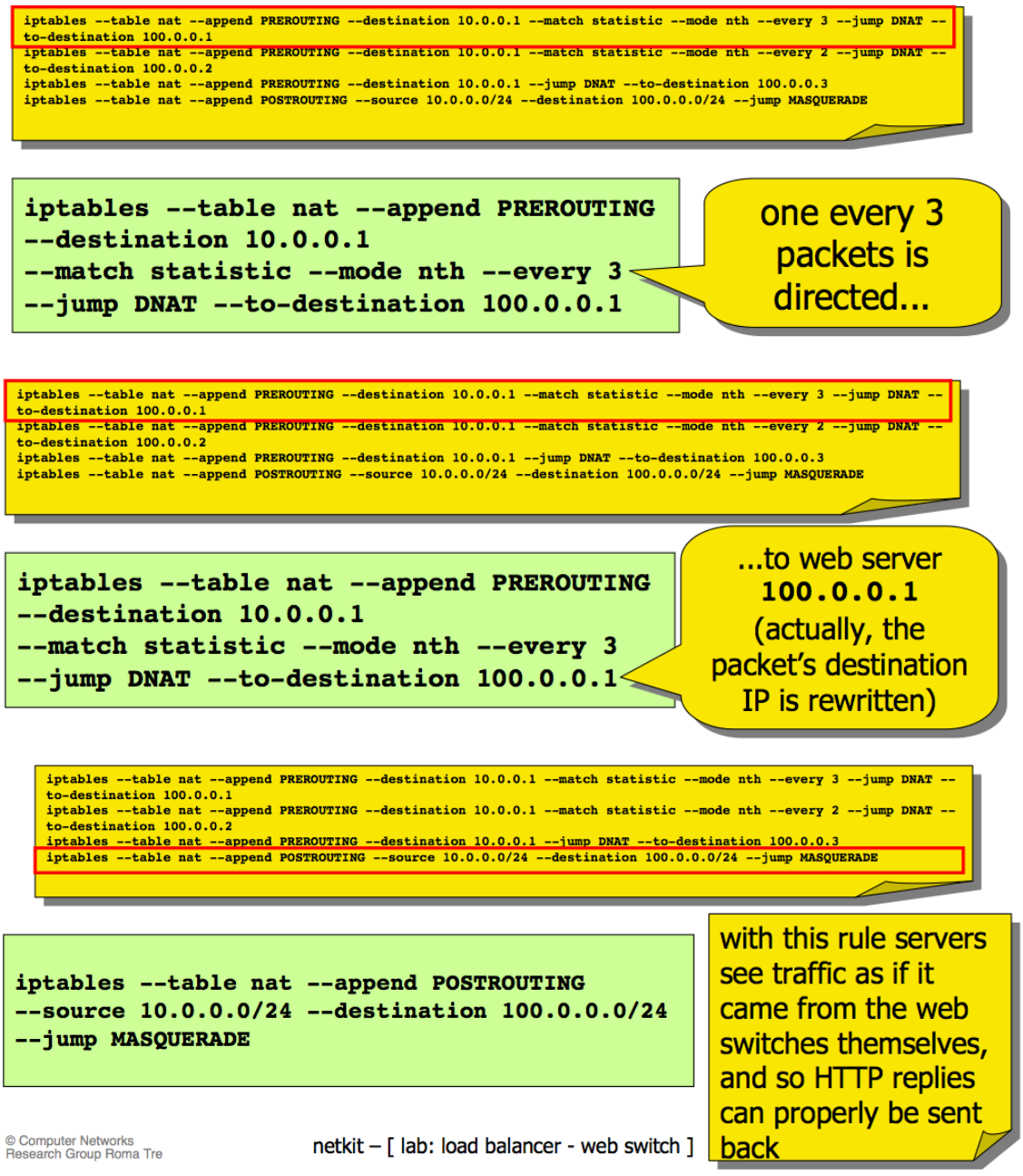
web switches
- chaque *web switch* implémente une politique différente pour diriger les requetes aux serveurs
- 1) ws_round-robin : envoyer la requete « i » au serveur ((i-1) mod 3)+1
- 2) ws_random : envoyer chaque requete à un serveur « random »
- 3) ws_ip : envoyer les requetes en se basant sur l'adresse IP (source) du client

clients
- avec un simple web browser (*links*)

- Chaque **serveur** a une adresse IP différente dans le sous-réseau **100.0.0.0/24**
- Pas de configuration spéciale, seulement une simple page HTMP par default **/var/www/index.html**

Chaque **web switch** a 2 interfaces :
- une vers le réseau « interne », avec une adresse IP dans le même sous-réseau que les serveurs
- une vers le réseau externe, qui expose donc un seul et unique adresse IP « virtuel » (*virtual IP address (VIP)*) aux clients
- Les clients voient seulement l'adresse VIP des web switches :
ils ne savent donc pas combien de serveurs il y a dans le *web farm*

Les **web switches** sont implémentés utilisant *Linux firewall iptables*
- 1) En particulier, pour la politique **« round robin »** :



-->A noter qu'il s'agit, en effet, d'une technique qui permet aux 3 *web switches* d'opérer dans la même topologie. Sans cela, les *web servers* seraient capable d'envoyer leur trafic seulement à travers une seule et unique *web switch* (via une route par default)

2)Pour implémenter la politique **« random »** :

```
iptables --table nat --append PREROUTING --destination 10.0.0.2 --match statistic --mode random --probability 0.33 --
jump DNAT --to-destination 100.0.0.1
iptables --table nat --append PREROUTING --destination 10.0.0.2 --match statistic --mode random --probability 0.5 --
jump DNAT --to-destination 100.0.0.2
iptables --table nat --append PREROUTING --destination 10.0.0.2 --jump DNAT --to-destination 100.0.0.3
iptables --table nat --append POSTROUTING --source 10.0.0.0/24 --destination 100.0.0.0/24 --jump MASQUERADE
```

```
iptables --table nat --append PREROUTING
--destination 10.0.0.2
--match statistic --mode random --probability 0.33
--jump DNAT --to-destination 100.0.0.1
```

similar to the round robin case, but the rule applies with a certain probability

3) Pour implémenter la politique **« (source) IP address-based »** :

```
iptables --table nat --append PREROUTING --destination 10.0.0.3 --source 10.0.0.10 --jump DNAT --to-destination
100.0.0.1
iptables --table nat --append PREROUTING --destination 10.0.0.3 --source 10.0.0.11 --jump DNAT --to-destination
100.0.0.2
iptables --table nat --append POSTROUTING --source 10.0.0.0/24 --destination 100.0.0.0/24 --jump MASQUERADE
```

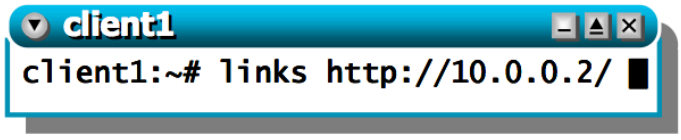
```
iptables --table nat --append PREROUTING
--destination 10.0.0.3 --source 10.0.0.10
--jump DNAT --to-destination 100.0.0.1
```

the server is deterministically chosen based on the client's (**source**) address

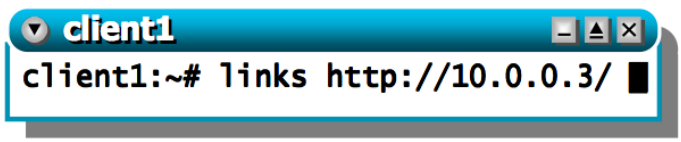
Expérimentation :

- **Question 1 (Q1)** : Choisissez un client (d'abord client1, ensuite client 2), utilisez le browser *links*, et dirigez les requêtes vers un des 3 VIP exposés par les *web switches :*

to experiment round robin balancing



to experiment random balancing

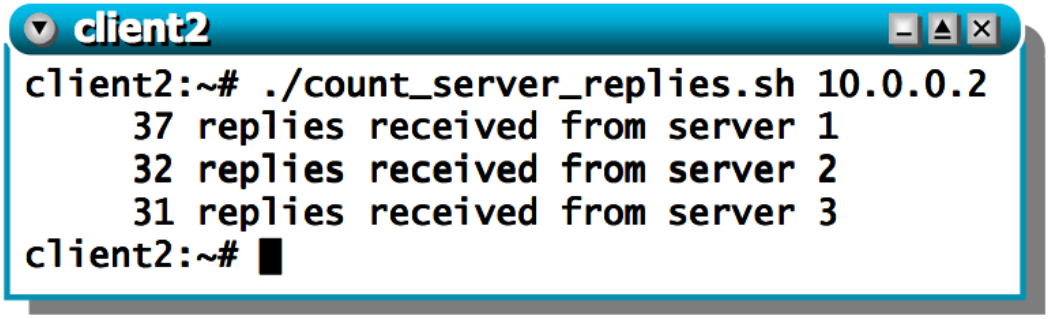


to experiment per-client-ip balancing

Une fois contacté un des VIPs, vous recevrez une page html qui dira quel serveur vous a fourni cette page
- Le *load balancing* peut être vérifié en faisant le *reloading* de la page (ctrl+R), mais attention ...

- ... par default, toutes les requêtes HTTP en *links* utilisent la même connexion TCP(HTTP 1.1 est persistent, par default) !
- … et puisque *iptables* trace les connexions TCP, toutes les requêtes HTTP qui sont envoyées dans la même connexion TCP sont dirigées vers le même serveur physique
- donc, pour apprécier réellement le *load balancing*, il s'avère nécessaire de fermer la connexion TCP et d'en rétablir une nouvelle, à travers les commandes :
- **ctrl+S**, et ensuite **ctrl+R**

- **Question 2 (Q2)** : Utilisez maintenant les scripts qui se trouvent déjà dans chaque client. Ce scripts envoient 100 requêtes HTTP (chacune sur une connexion différente) à une adresse IP spécifiée.
- il donne come report en output le nombre de pages qui ont été servies par chaque server physique.



- Après le boot, chaque *web switch* montre automatiquement les statistiques concernant le nombre de fois que les "*iptables rules*" ont été utilisées (matching)

- **Question 3 (Q3)** : vérifiez le champ **pkts** dans la chaine **PREROUTING**