Firewall d'infrastructure centralisé

et load-balancing

Adrien Urban
Responsable R&D

28 février 2013

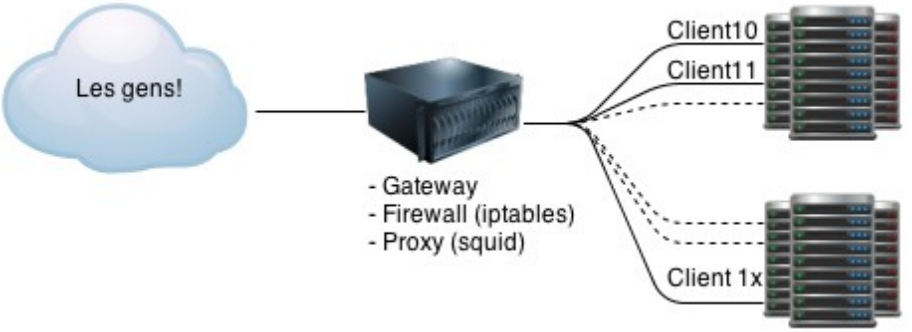
Plan

- Historique (le contexte)
- Benches
- Routing
- Firewall
- Évolutions

Historique

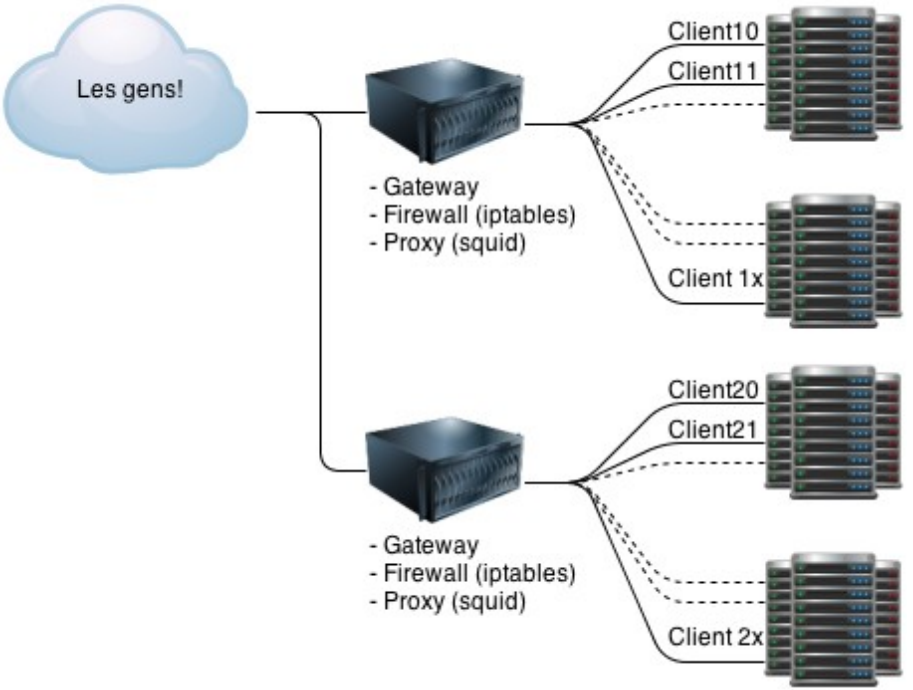
Historique – chante-sloubi (2009)

- Hebergement web
- Écosystème
  - Proxy (squid) partagé
  - Sous-réseau privé



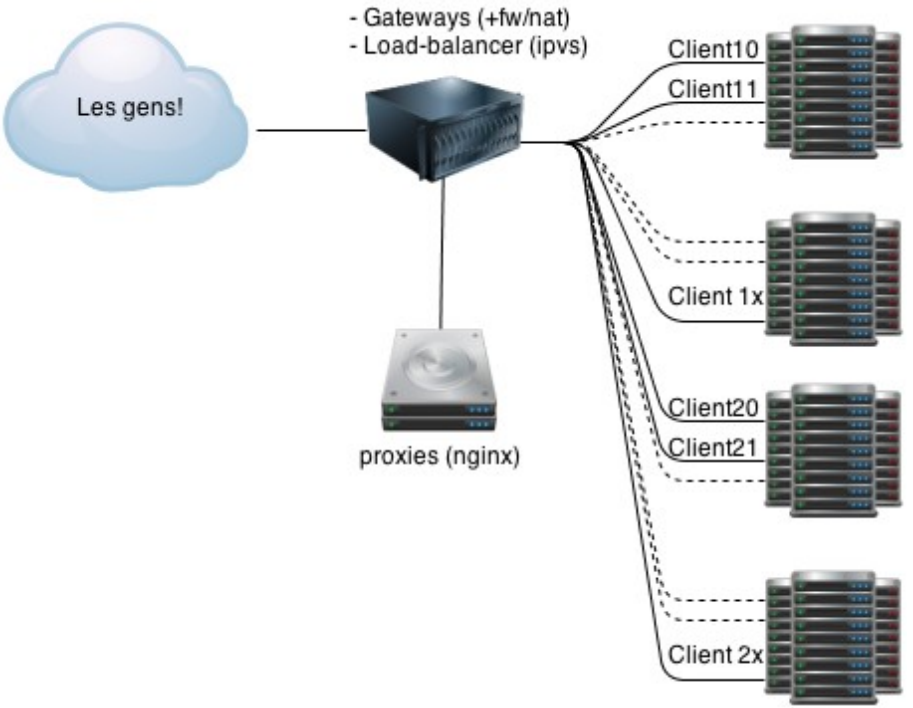
Historique – chante-sloubi (2009)

- Hebergement web
- Écosystème
  - Proxy (squid) partagé
  - Sous-réseau privé
- Ajout en sparadrap

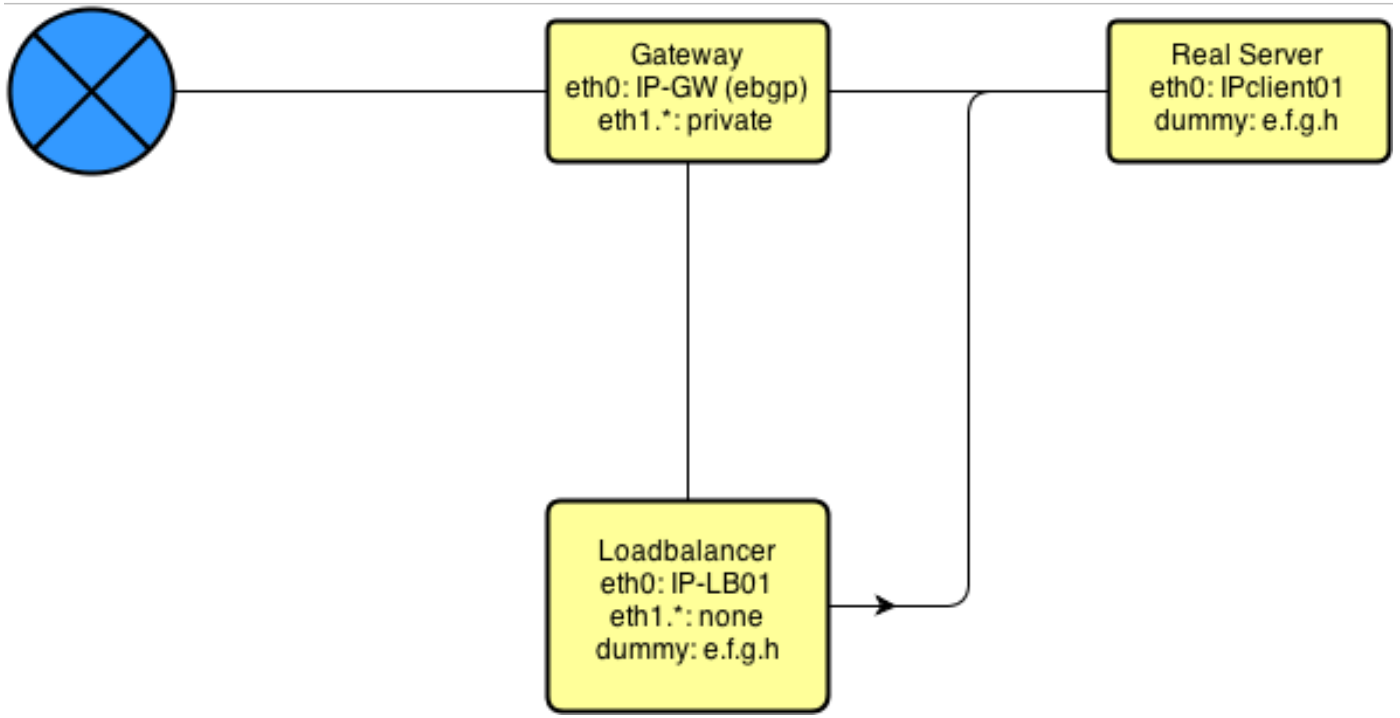


Historique – on nettoie (2010-2011)

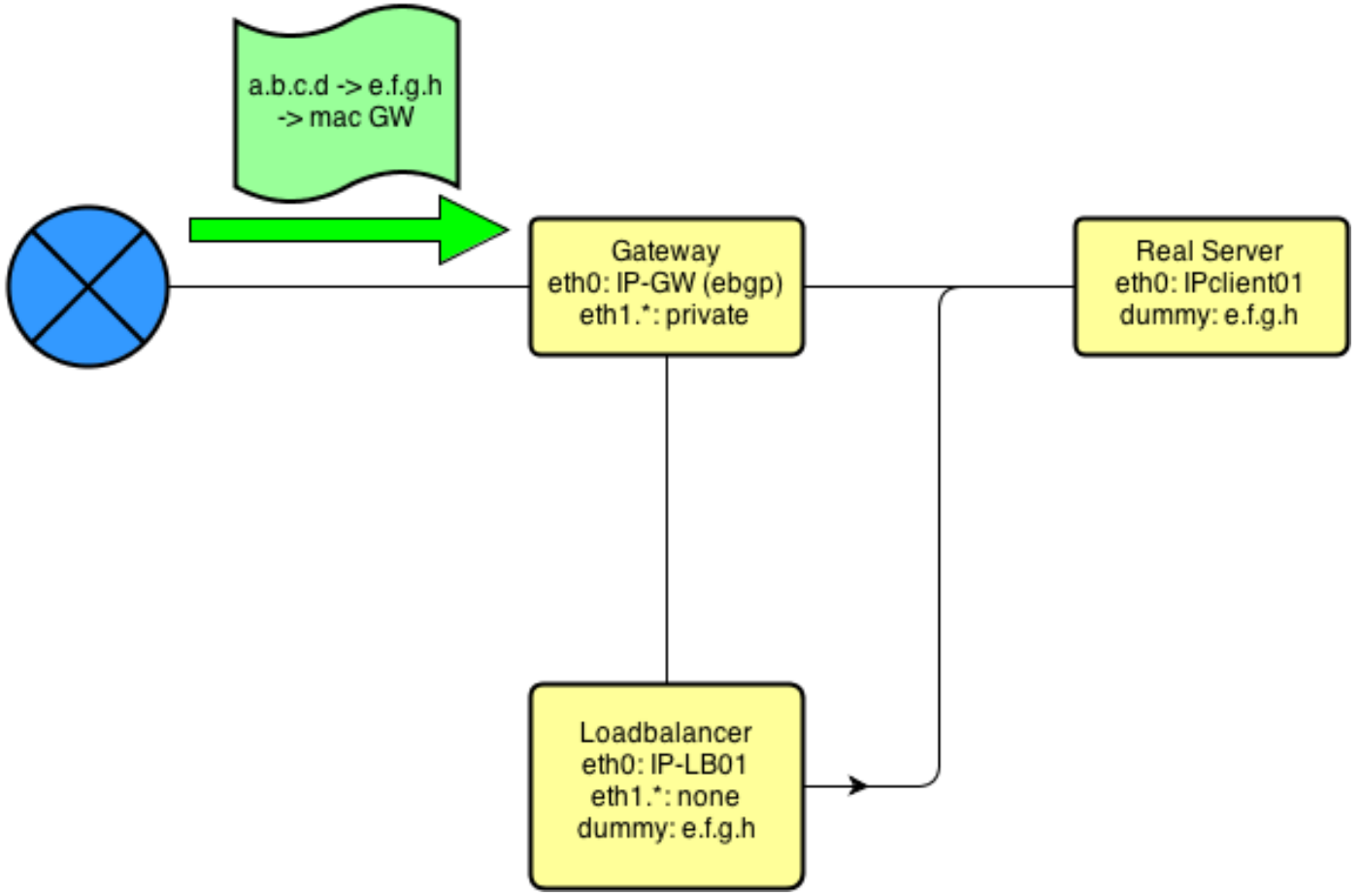
- On rationalise
  - redondance
  - qui peut monter en charge
- Technos
  - keepalived/ipvs
  - nginx



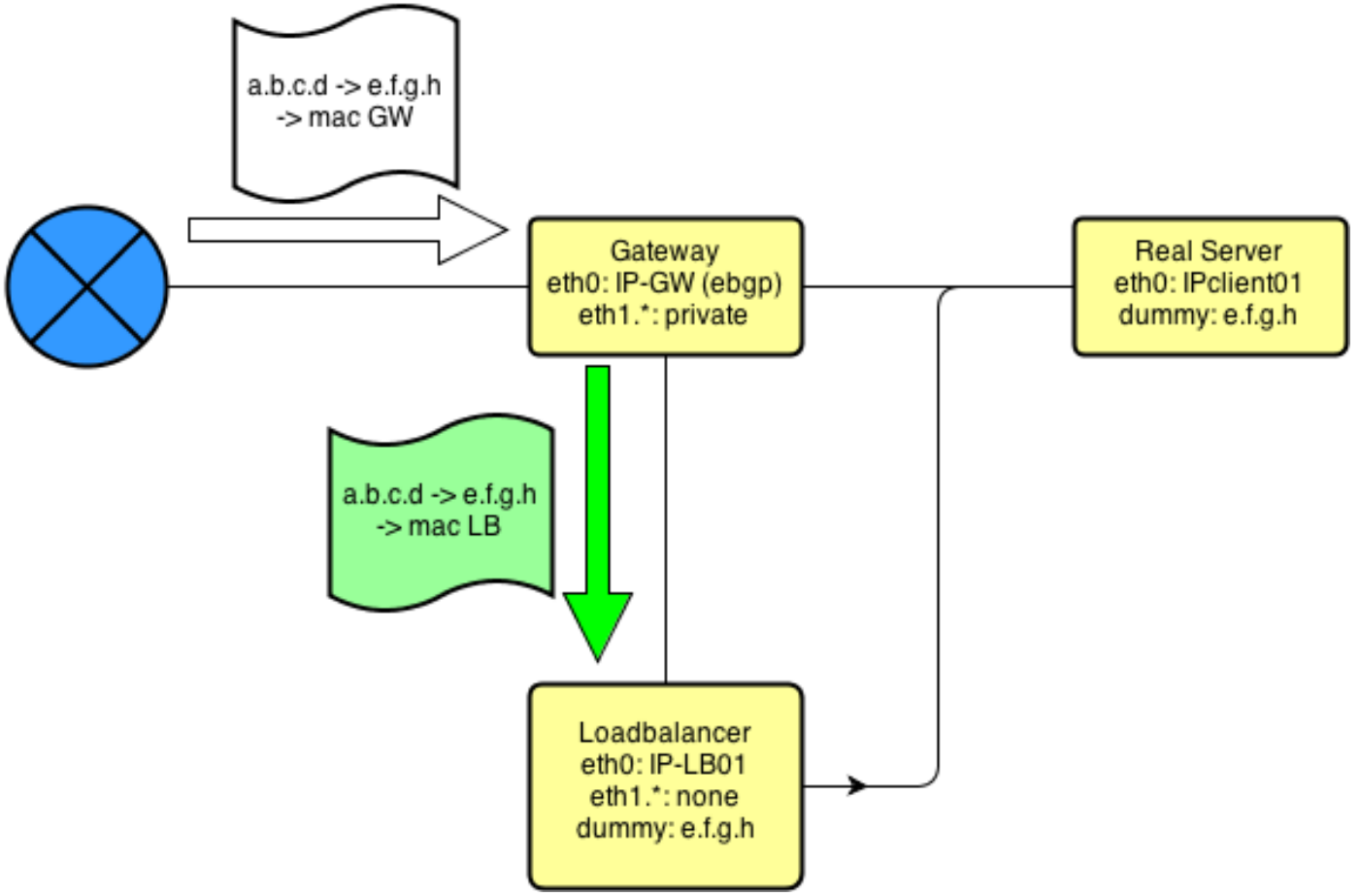
Historique – Direct Routing



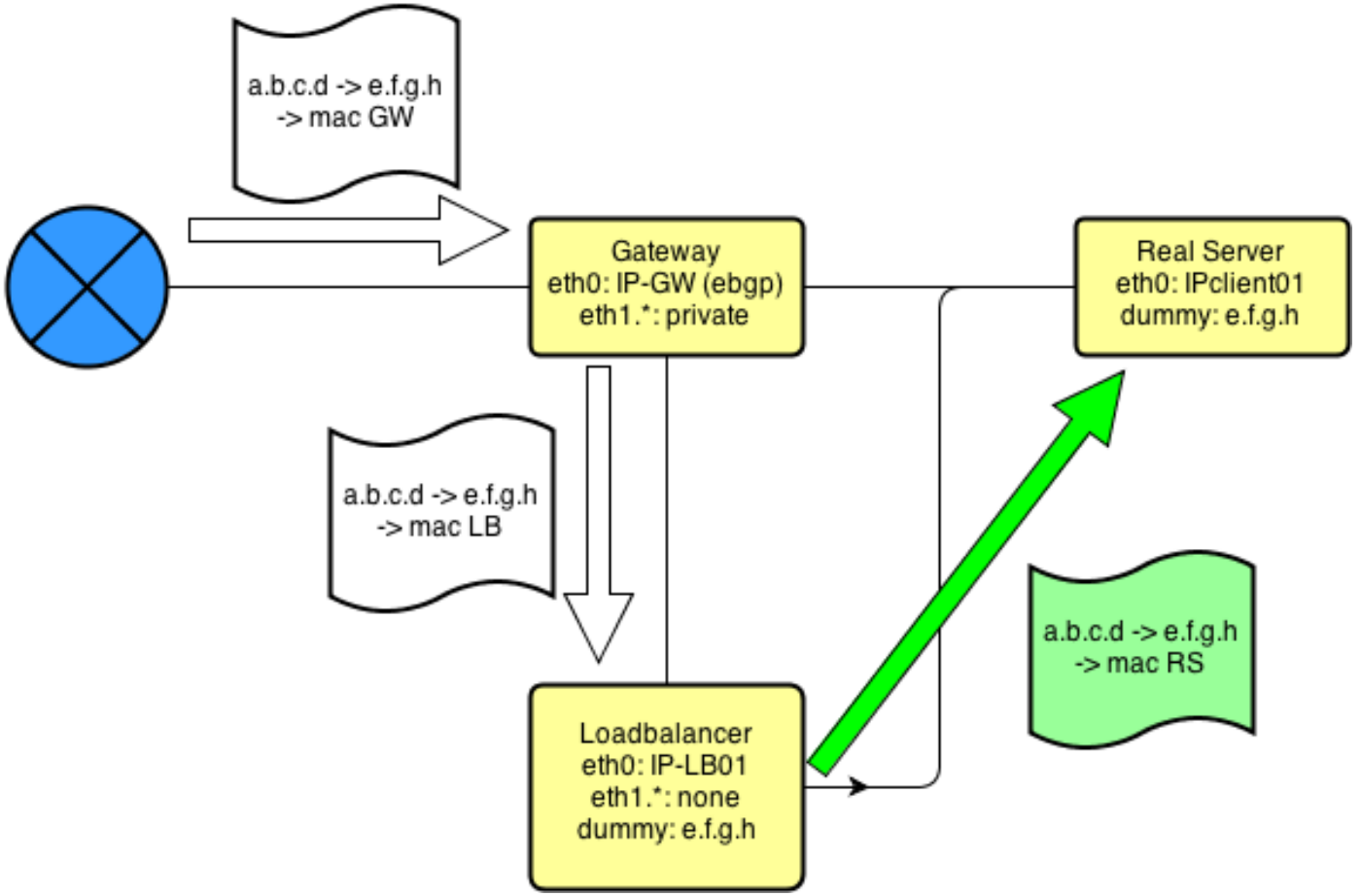
Historique – Direct Routing



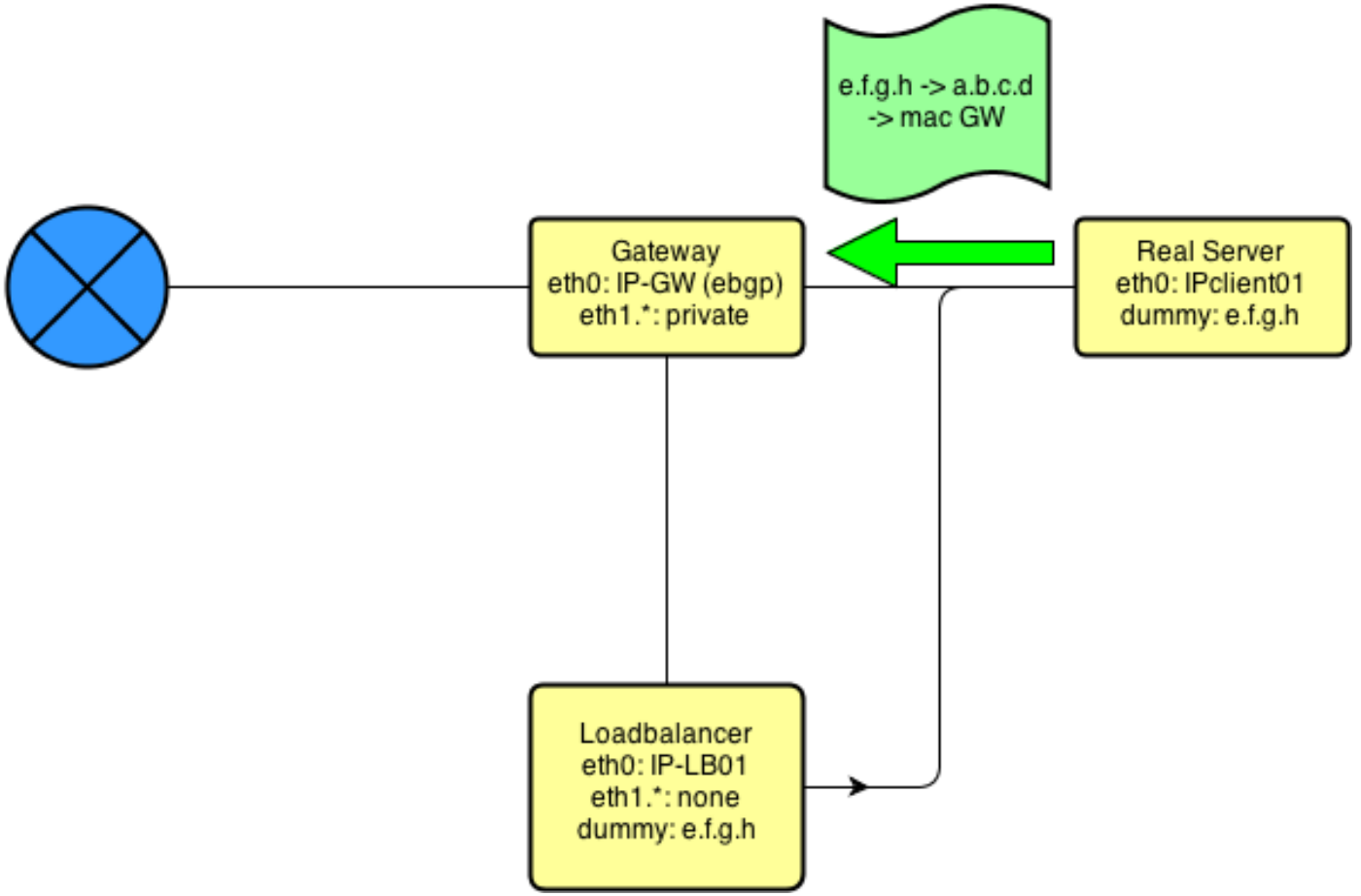
Historique – Direct Routing



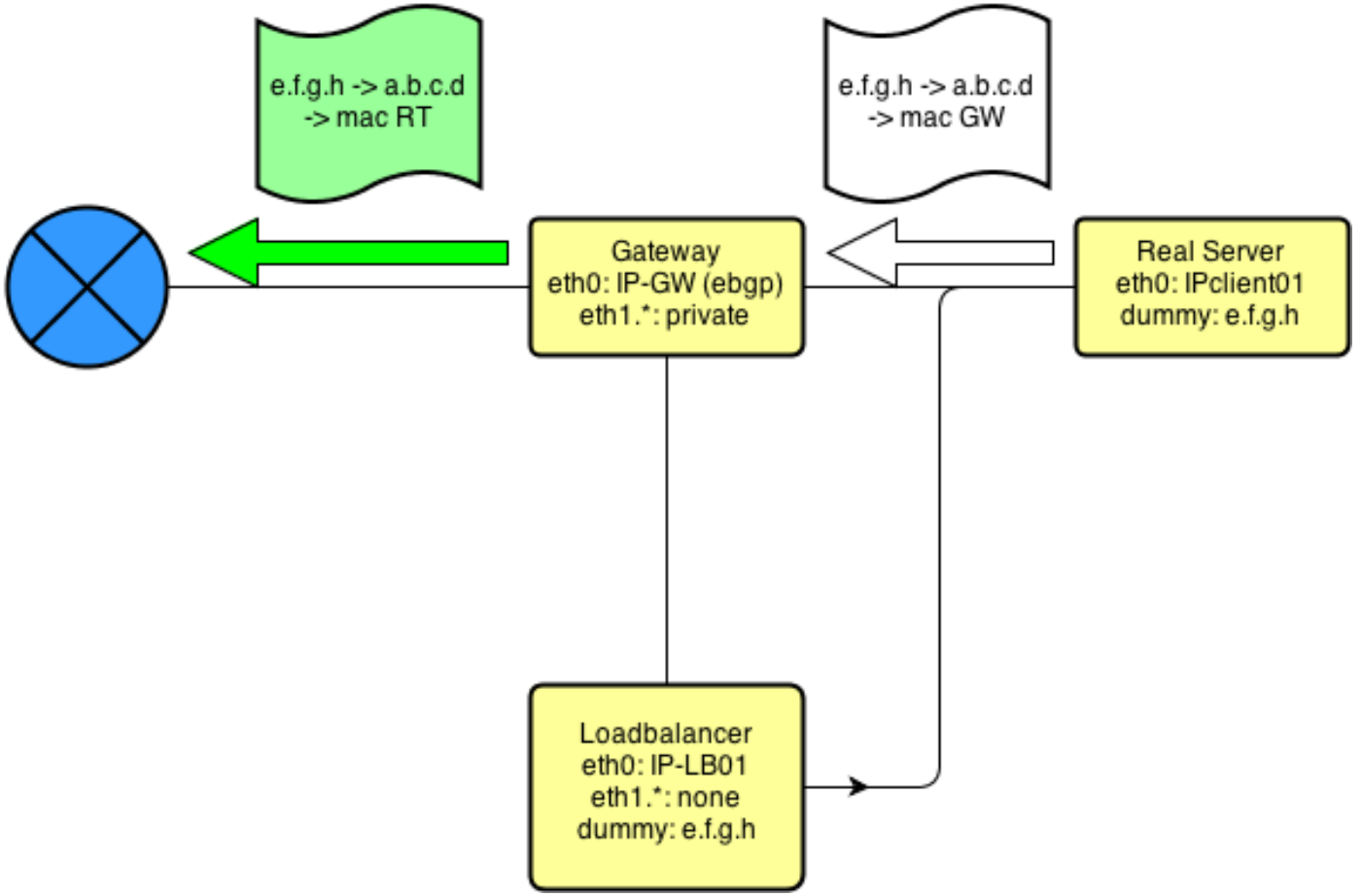
Historique – Direct Routing



Historique – Direct Routing



Historique – Direct Routing



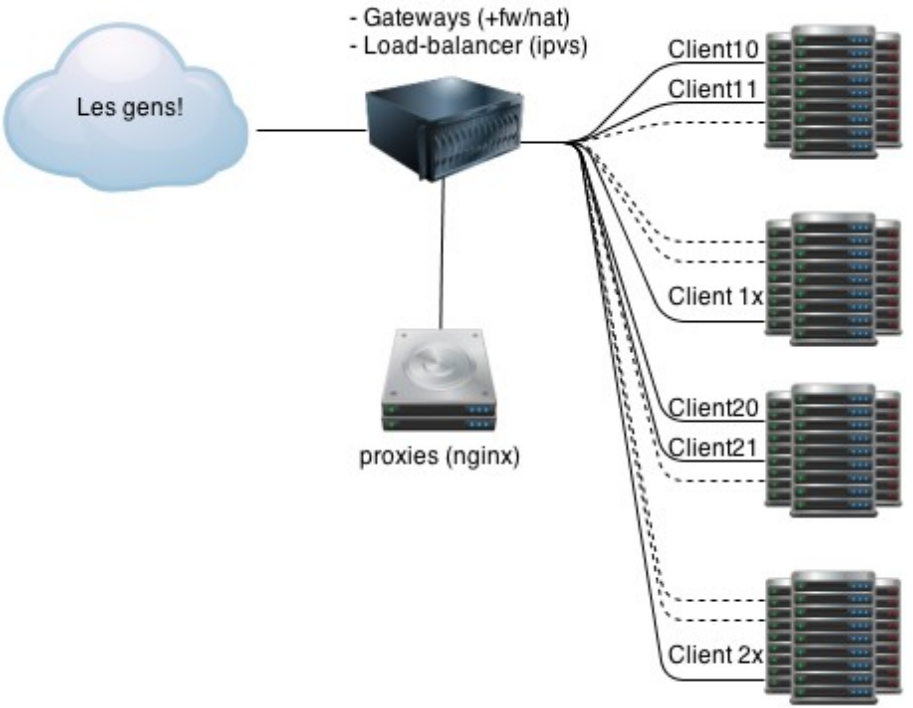
Historique – Direct Routing

Avantages

- Configuration rigoureusement identique sur l'ensemble des serveurs d'un pool
- Possibilité de faire du service détecté par IP, sans multiplier le nombre d'IP privées
- Ajout simple d'un nouveau serveur

Historique – oui, mais

- ipvs sur la gateway, besoin de patch spécifique
- iptables et ipvs ne s'aiment pas
- conntrack trop élevé
- bande passante qui augmente

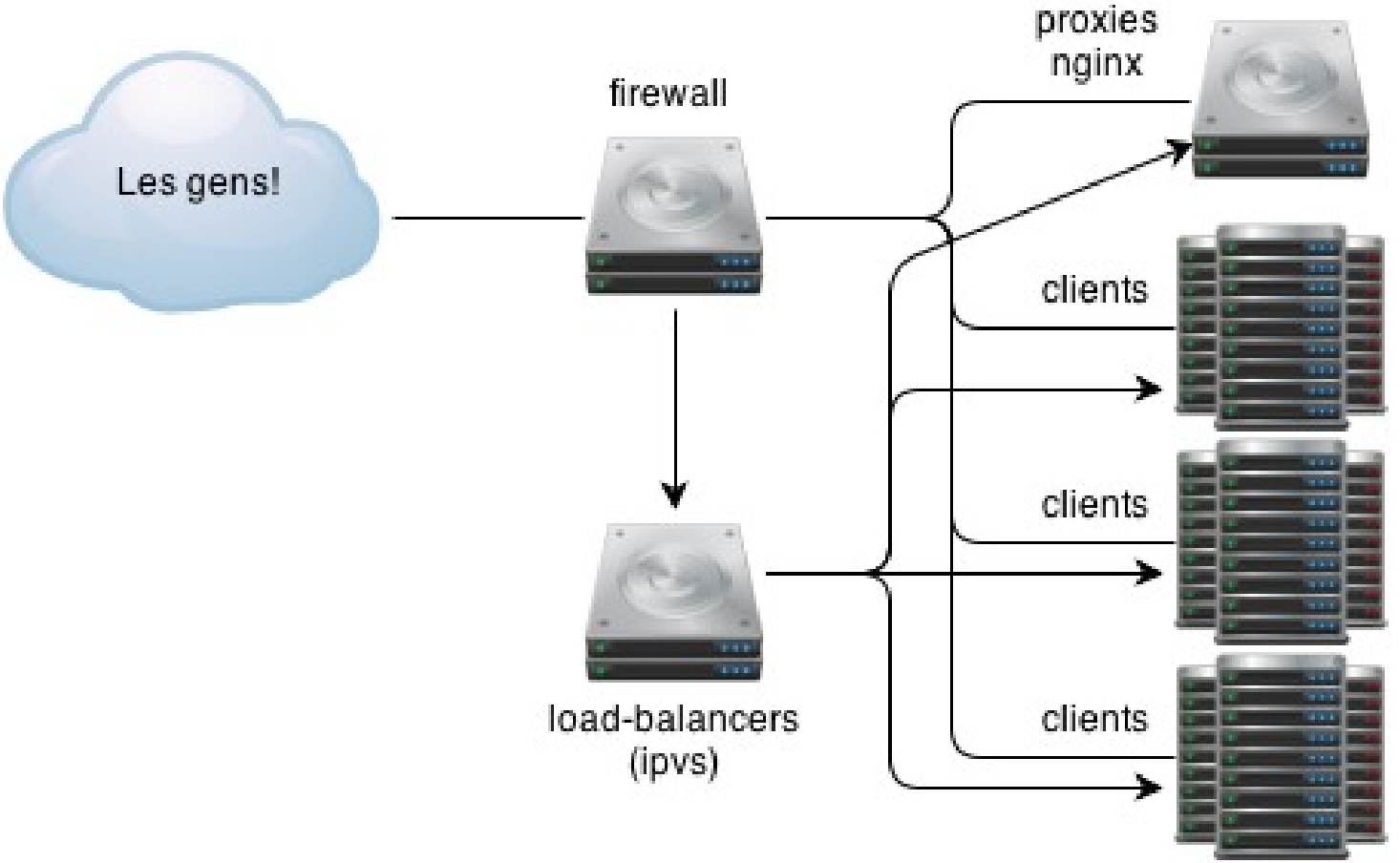


Historique – grandir (2012)

- Besoin de plus de bande passante
- Besoin de pouvoir faire du firewalling
- Besoin d'être industrialisable

- On garde les technos
- On sépare les serveurs
- On passe en 10Gbps

Historique – grandir (2012)



Bench

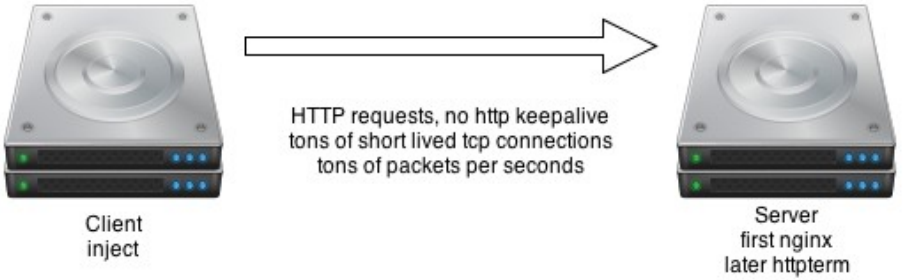
Bench

- Des serveurs 10Gbps... ok, mais que sont-ils capables de faire ?
- Combien de pkt/s ?
- Combien de connexions ?

Resultats disponibles en ligne: https://www.hagtheil.net/wiki/system/benches10gbps

Bench - baseline

1 client → 1 serveur



jusqu'à 280k hits/s

Point importants a configurer :

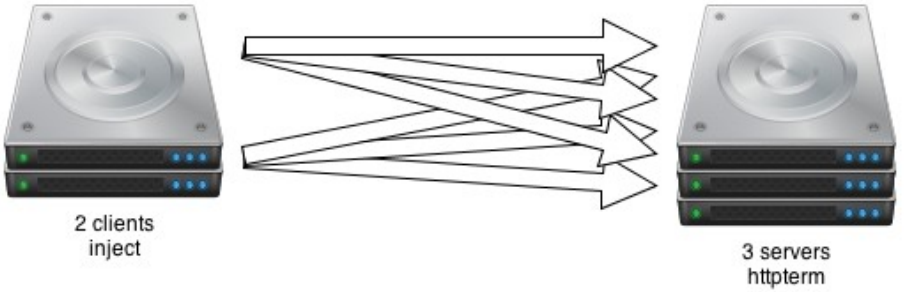
- smp affinity (et sur les processus, et sur les interruptions)
- Trouver les goulots d'étranglement (ici, applicatif)

Bench - baseline

3 client → 2 serveur

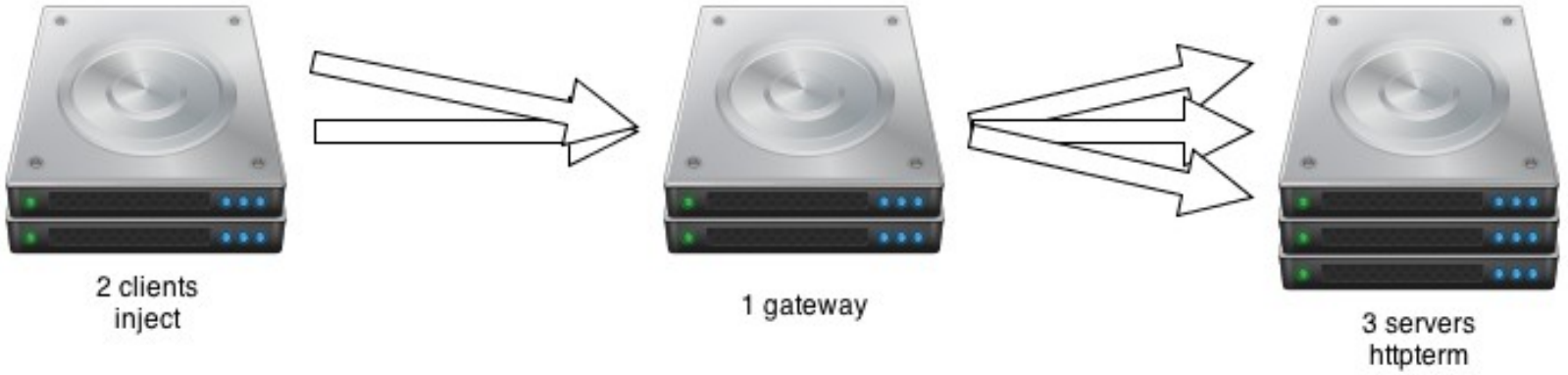
800k hits/s

5.6M pkt/s



Cette configuration nous permet d'avoir une base avant de placer une machine sur le chemin

Bench - baseline



Au travers d'une paserelle

une seule carte réseau est utilisée

pas d'impact de performances

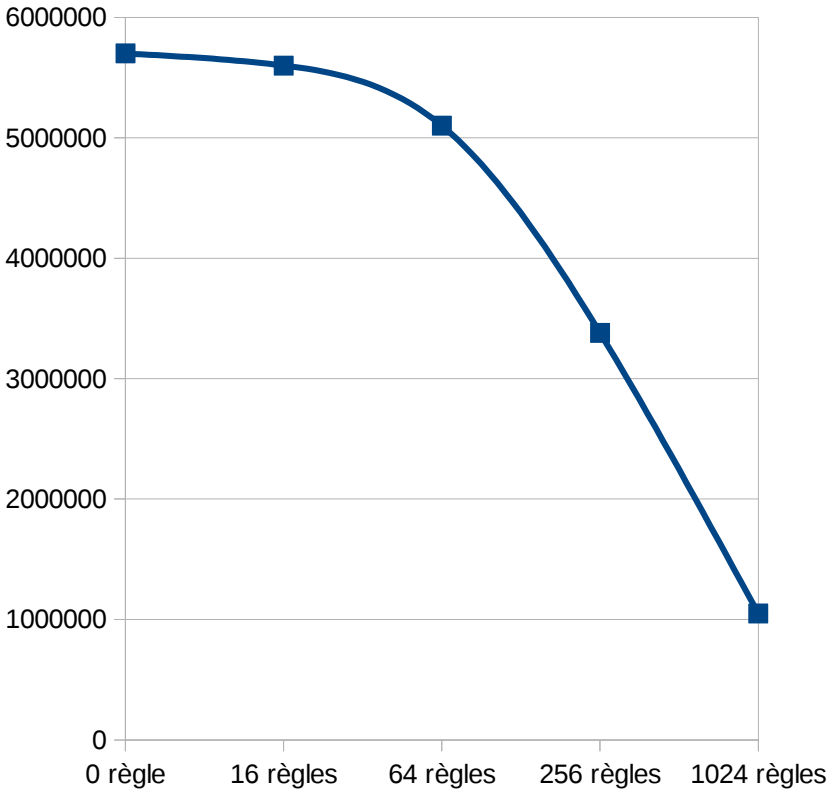
Bench – firewall - base

iptables non chargé	filter, raw et mangle (sans règle)
5.7M pkt/s	4.9M pkt/s
nat (conntrack) (sans règle)	nat + mangle NOTRACK
1.2M puis 950k pkt/s	4.4M pkt/s

Bench – firewall - matches

Règles de match sur l'adresse source

Augmenter le nombre de match diminue rapidement les performances



Bench – autres règles

256 règles, avec différents types de match

Rappel: le flux testé est du tcp port 80

256 règles ...	pkt/s
-s a.b.c.d	3.3M
-d a.b.c.d	3.2M
-p udp --dport 53	2.4M
-p tcp --dport 443	1.1M
-p tcp --dport 80	990k
-m u32 -u32 "0xc&0xffffffff=0xa0000xx"	480k

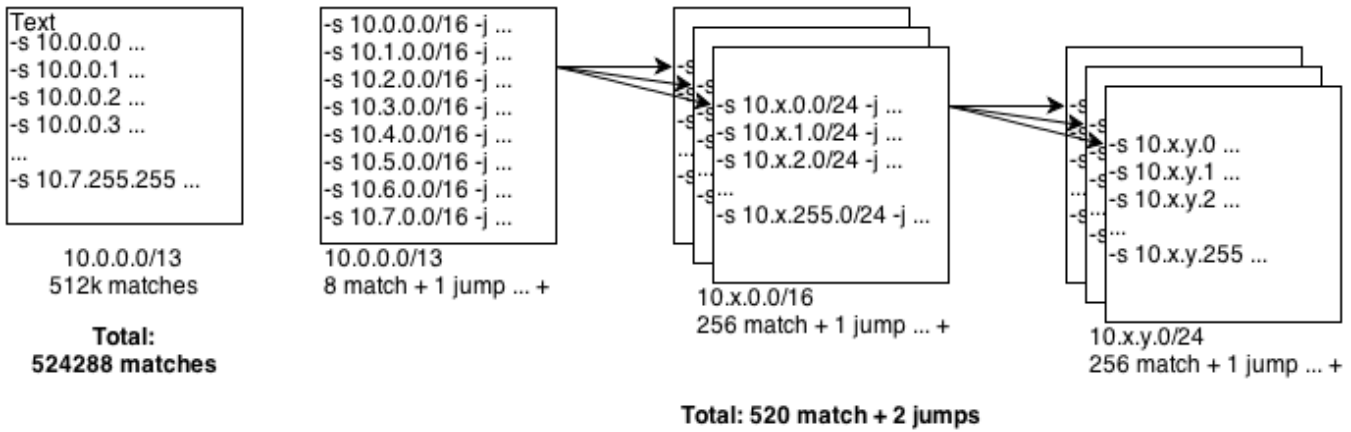
Bench – ipset

- Beaucoup de règles tue les performances
- ipset permet de matcher de nombreuses IP (jusqu'à 64k) en une règle iptables

Nombre de règles	paquets / s
1 règle ipset (vide)	3.6M
2 règles ipset	2.05M
3 règles ipset	1.45M

Bench – arbre de recherche

- Utiliser des ipset tue les performances
- Le nombre de match effectue a un impact
- Et si l'on parcourait un arbre de recherche ?



Bench – arbre de recherche

Bit par match (match/chain)	évaluation / paquet	Match et jump / paquet	paquet/s
2 (4)	39	11	3.9M
3 (8)	51	8	4.2M
4 (16)	73	6	4.0M
5 (32)	113	5	4.0M

- Match d'un /13 complet (512k IP distinctes)
- Moins couteux que 256 règles (3.3M pkt/s)
- Utilise 20GB de RAM
- Bien moins couteux qu'ipset

Bench – ipvs

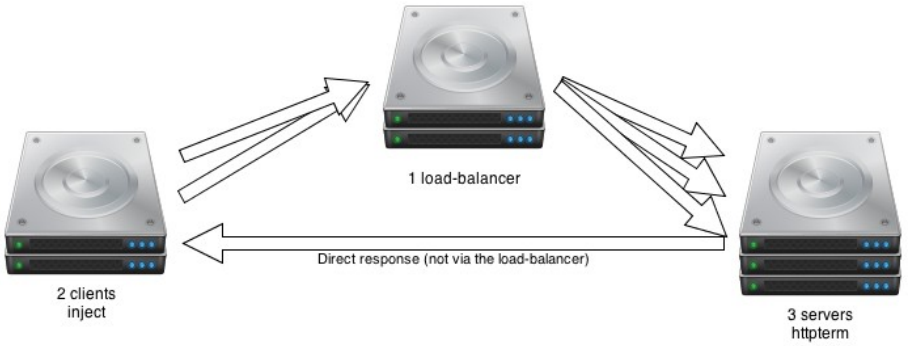
- Ne voit les paquets que dans un sens
- de nombreuses connexions très rapides, quasiment le pire cas

Résultats

- Entre 1.4 et 1.5M pkt/s
- Plus de 350k conn/s
- Plus de 40M connexions dans ipvs

Optimisations:

- smp_affinity pour les interruptions
- conn_tab_bits=20 (au chargement du module)

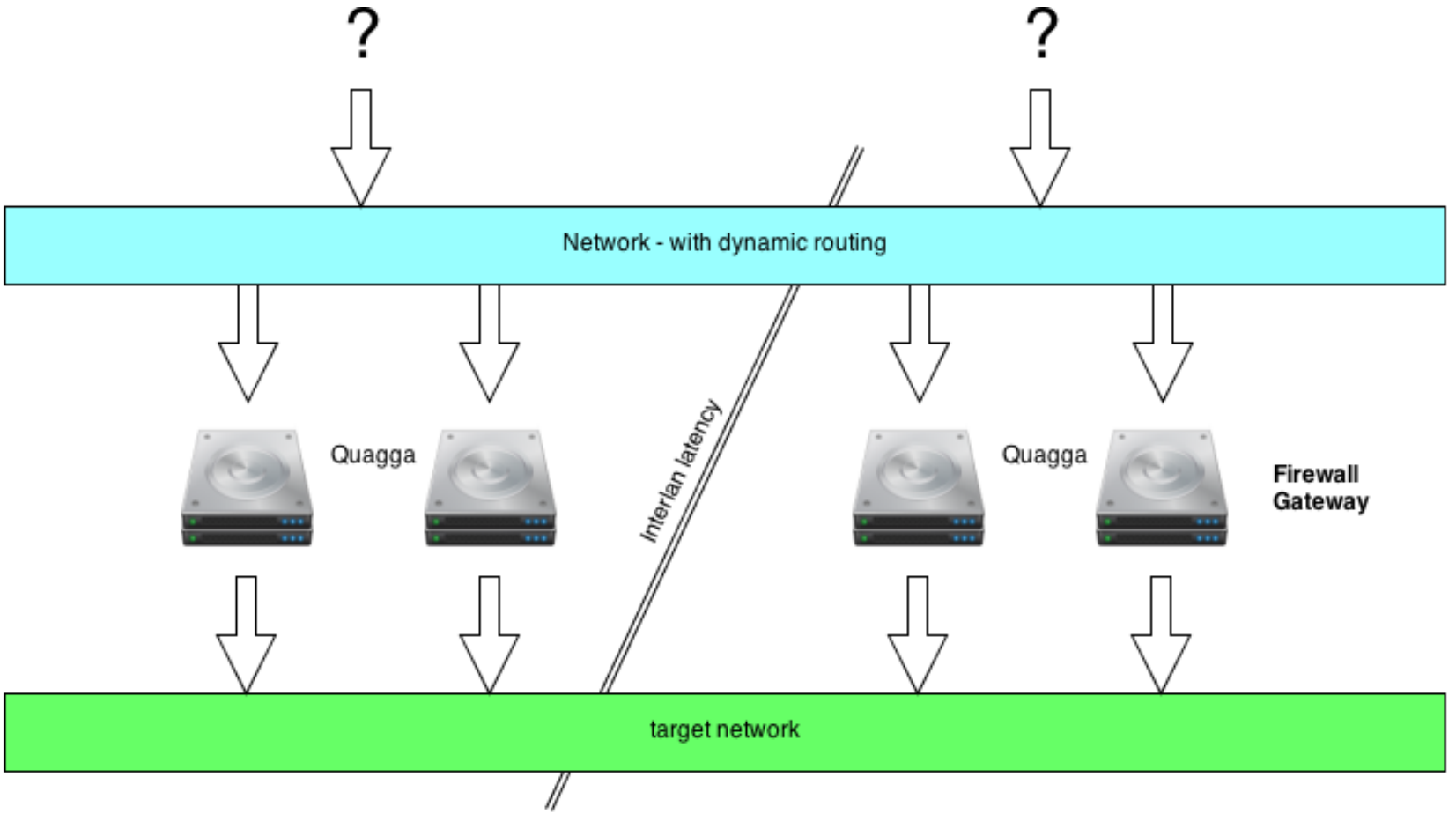


Bench - conclusion

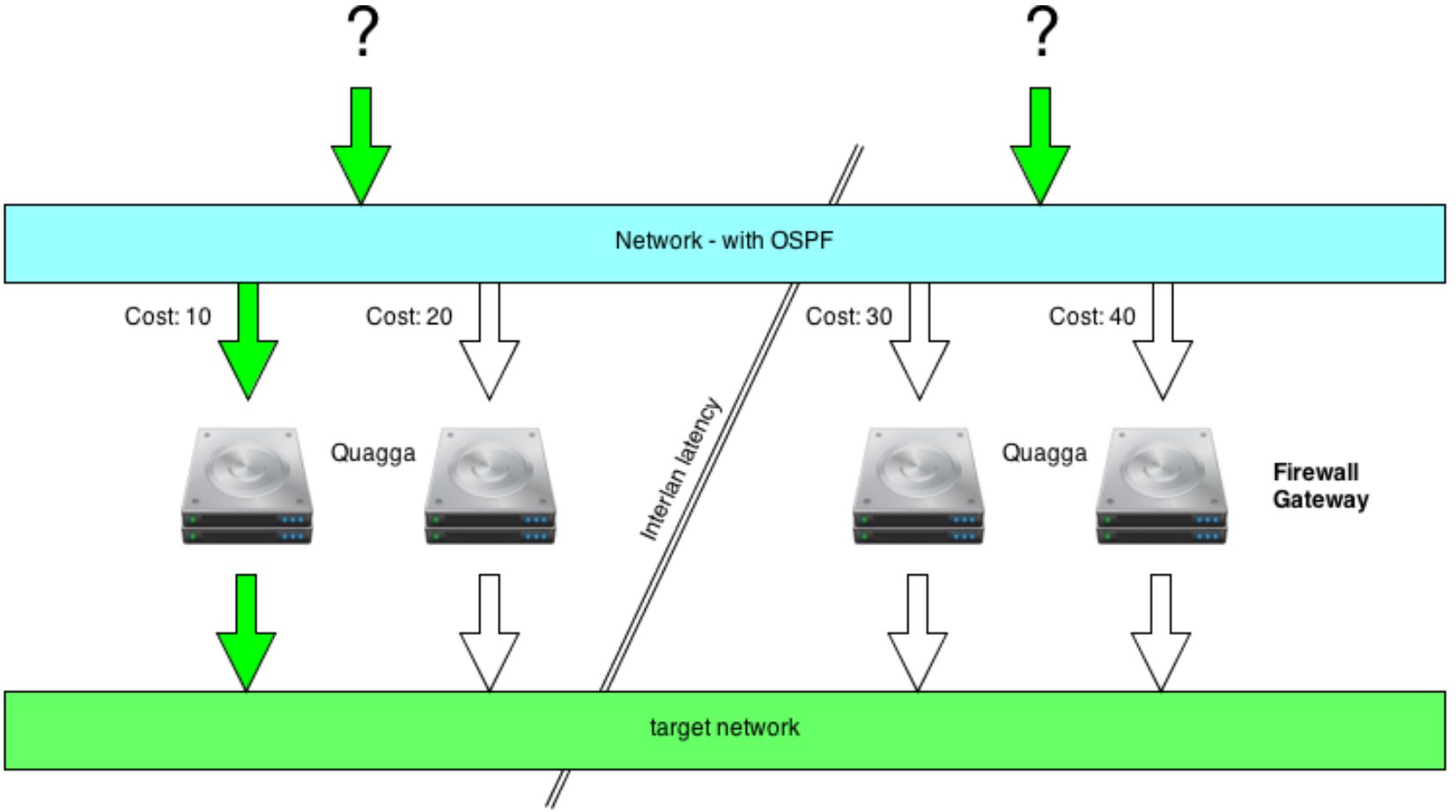
- ipvs sera limitant
- Il faut limiter le conntrack
- Il faut également limiter le nombre de règles traversées, et utiliser des arbres de recherches
- Les tests auront permis de definir l'affinité smp pour les interruptions sur ce matériel

Routing

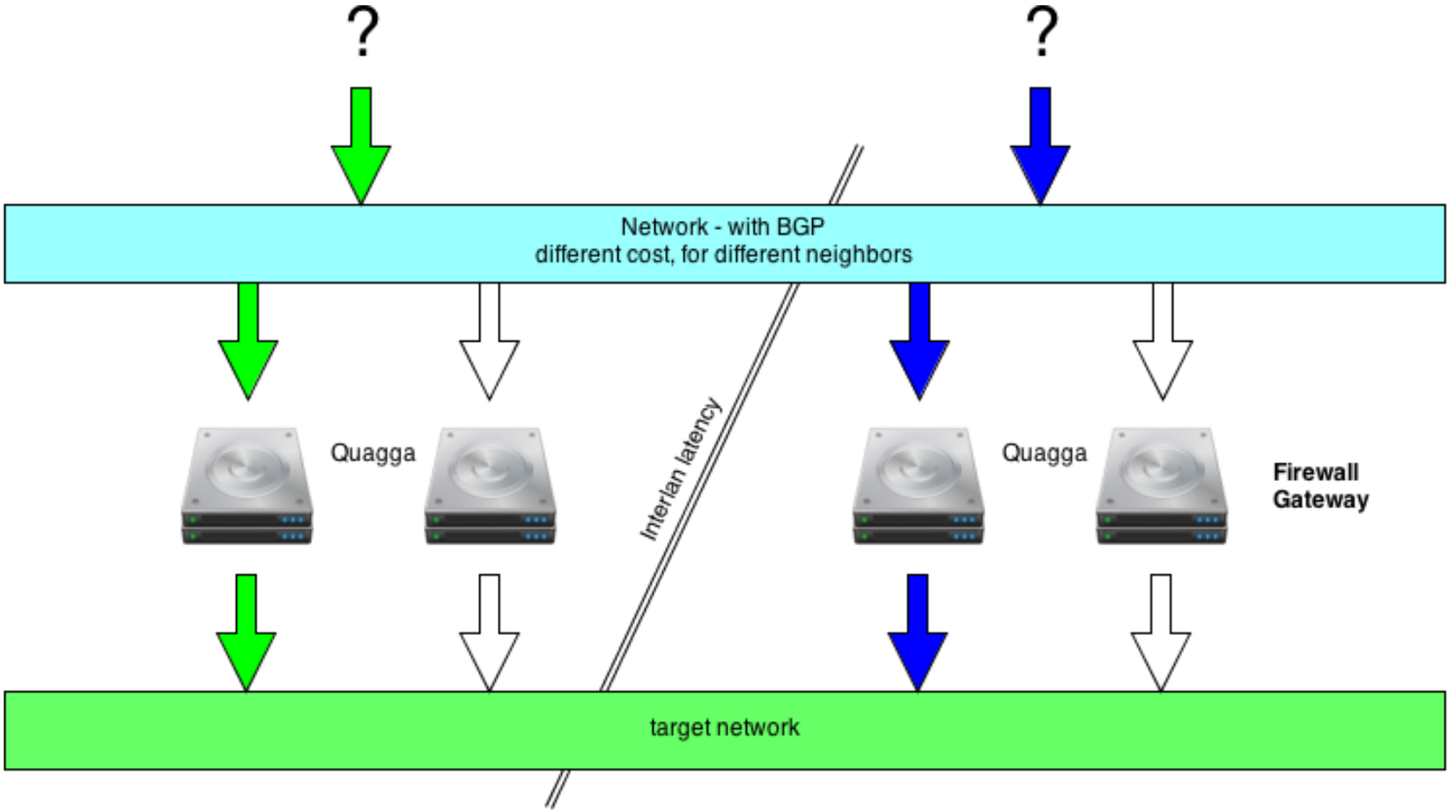
Routing – le besoin



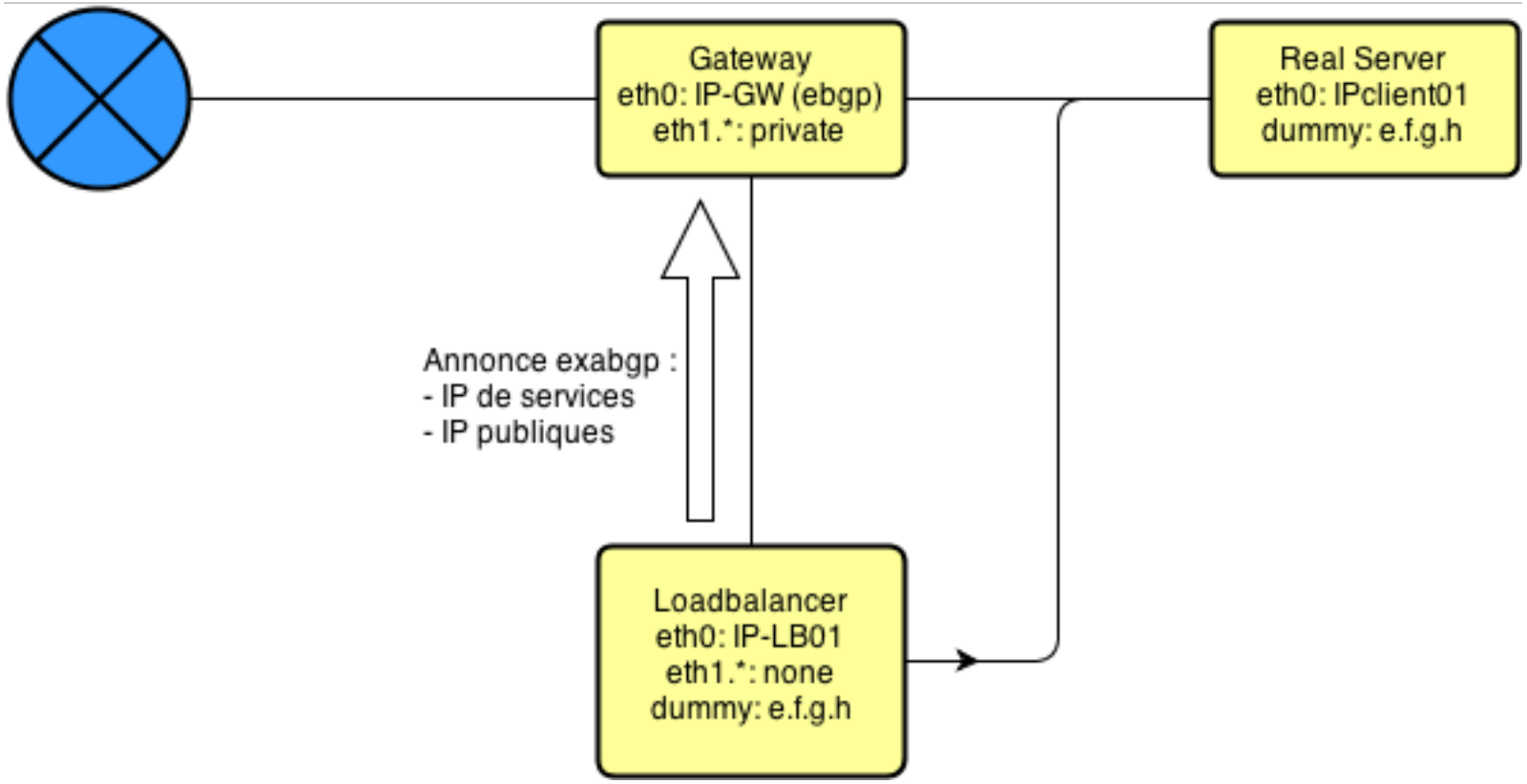
Routing – ospf



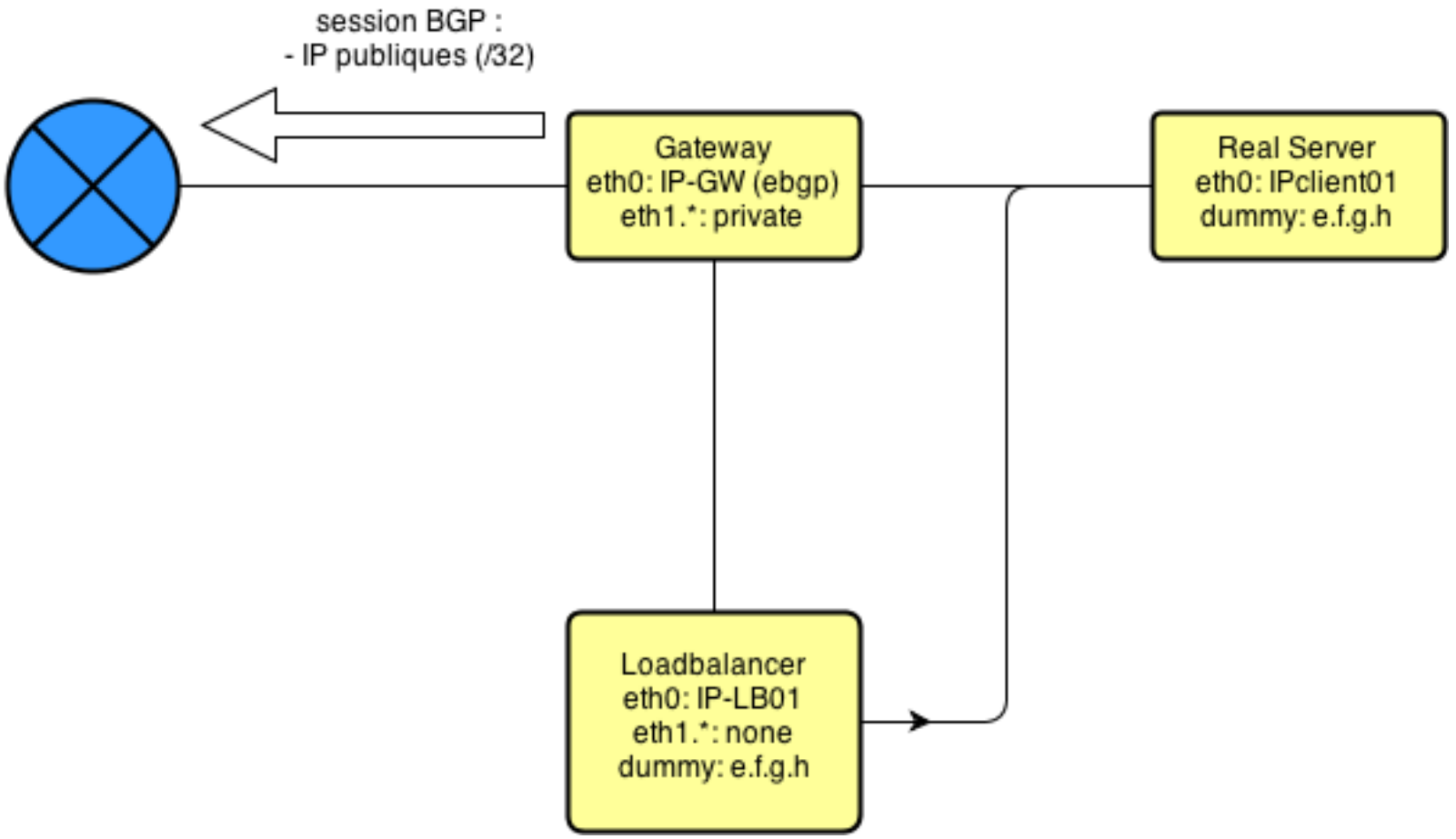
Routing – bgp


?
?
Network - with BGP
different cost, for different neighbors
Quagga
Quagga
Interlan latency
Firewall
Gateway
target network

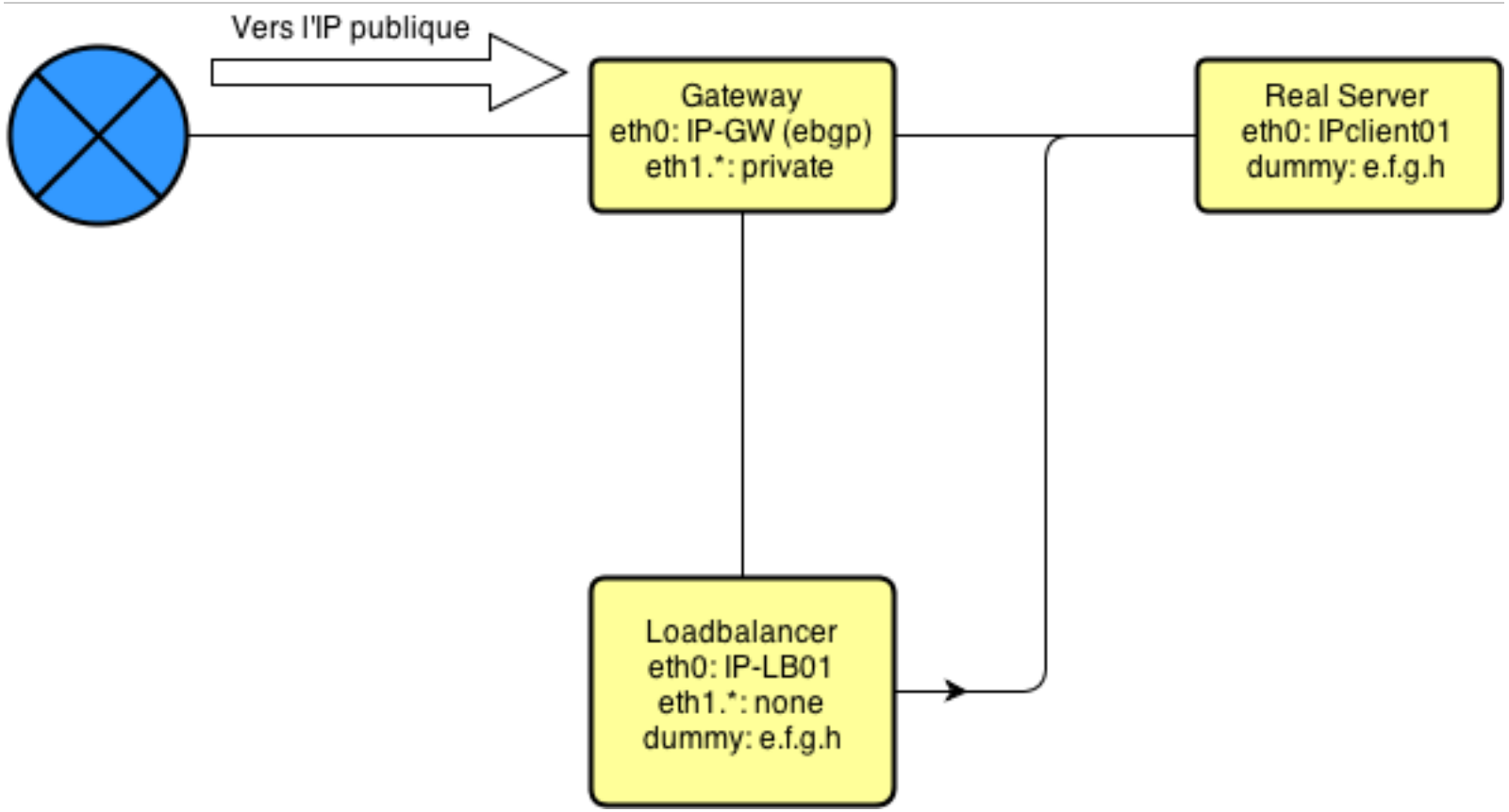
Routing – services



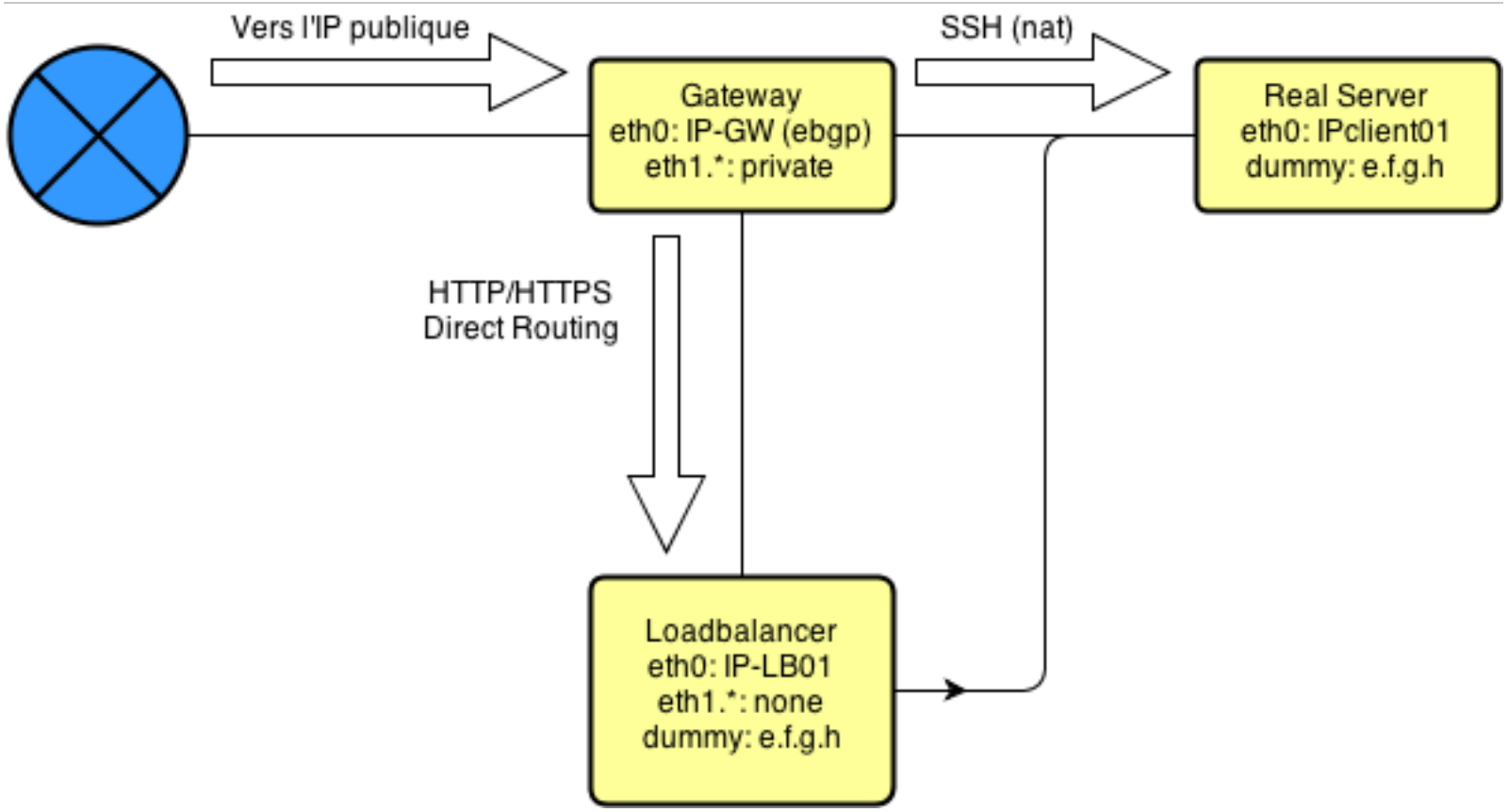
Routing – services



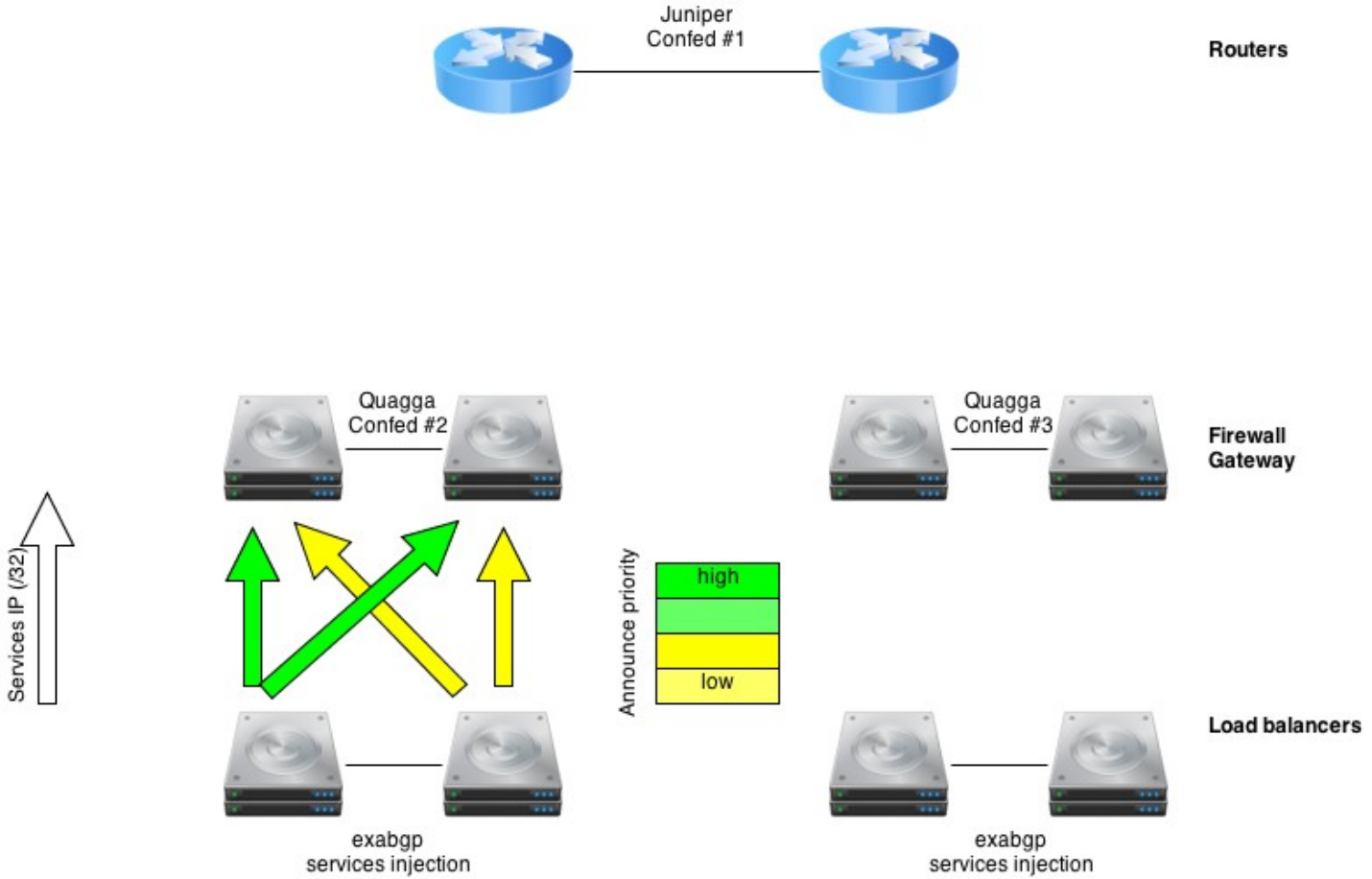
Routing – services



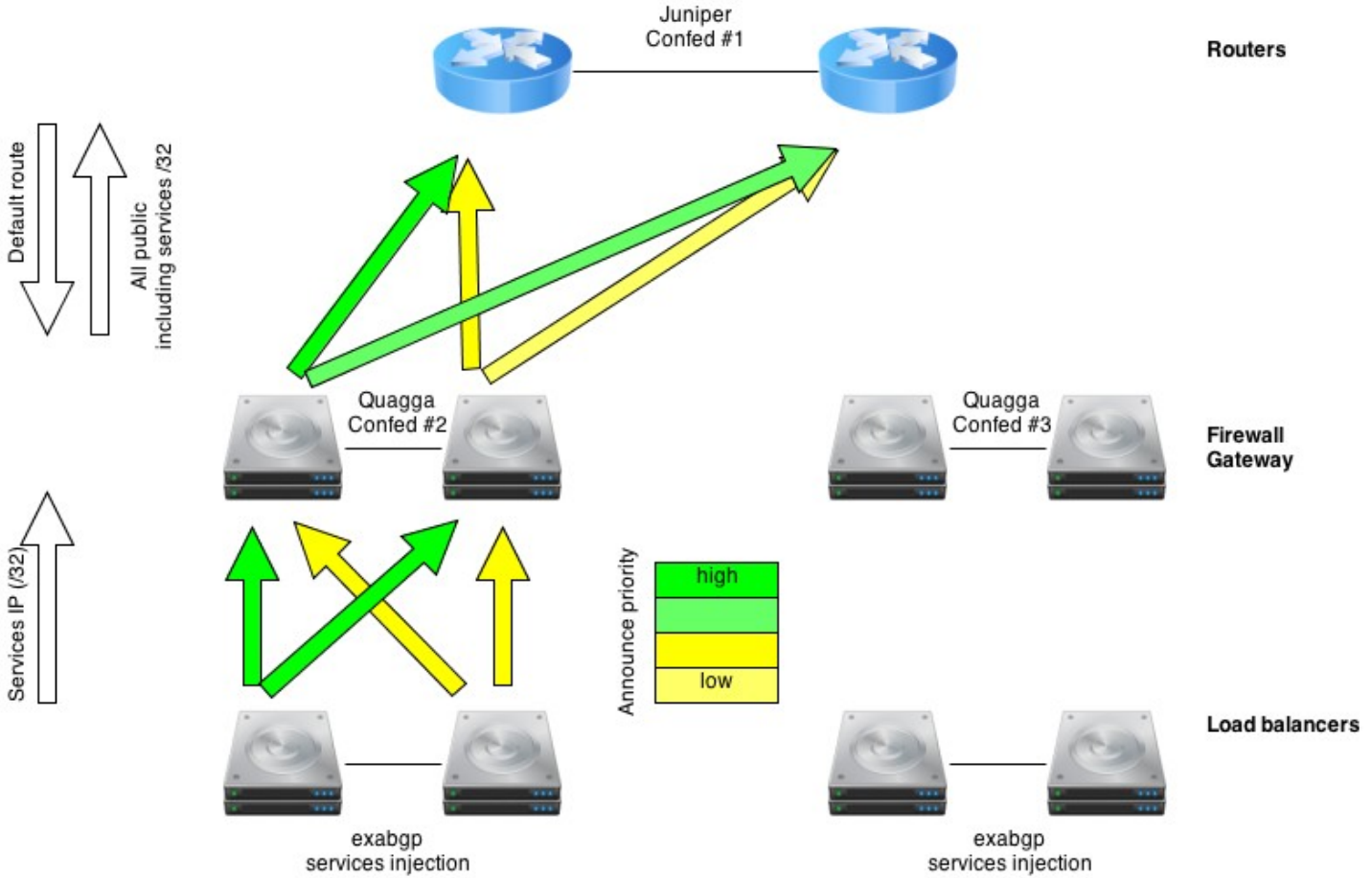
Routing – services



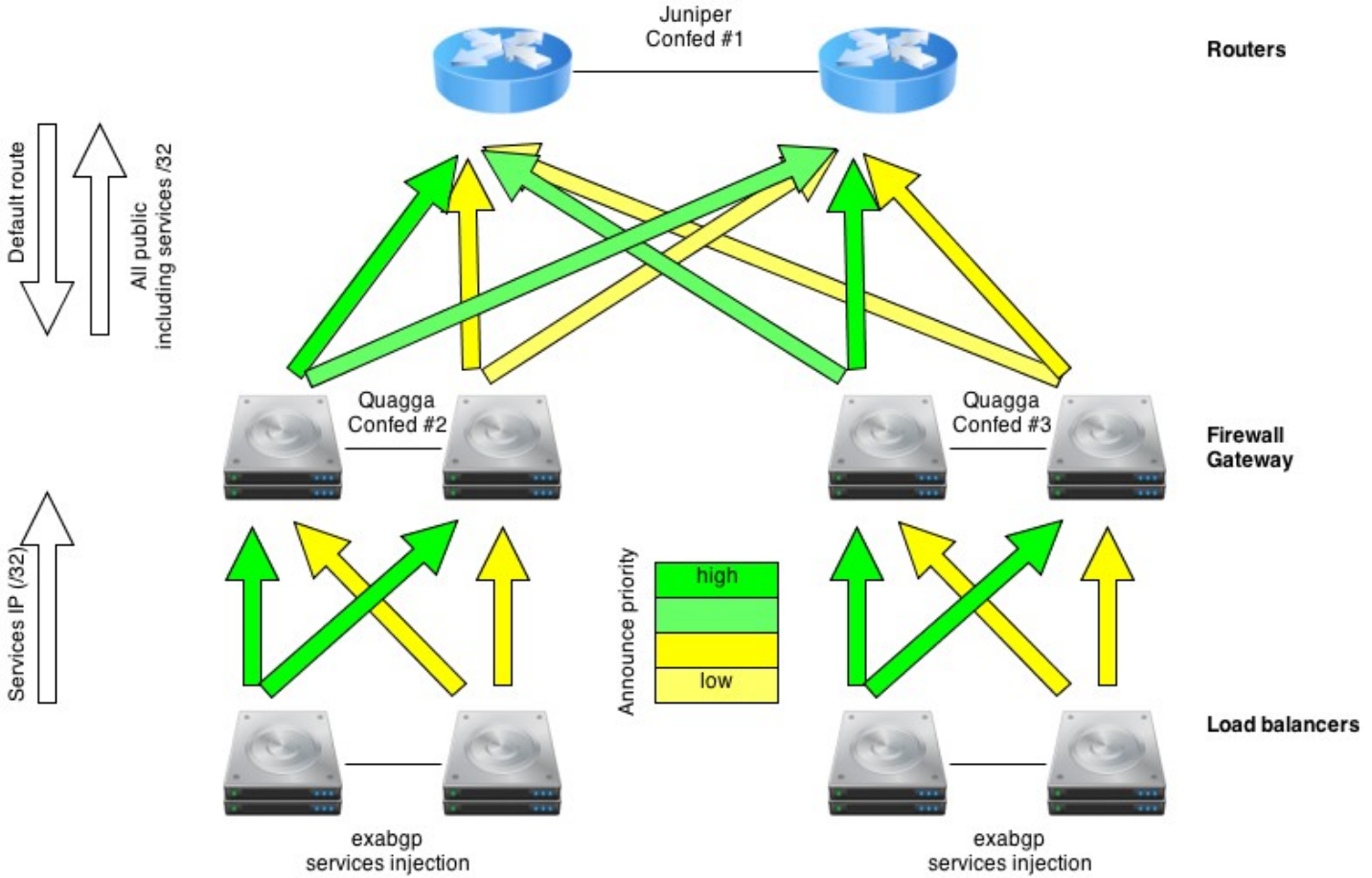
Routing – bgp + exabgp



Routing – bgp + exabgp



Routing – bgp + exabgp



Firewall

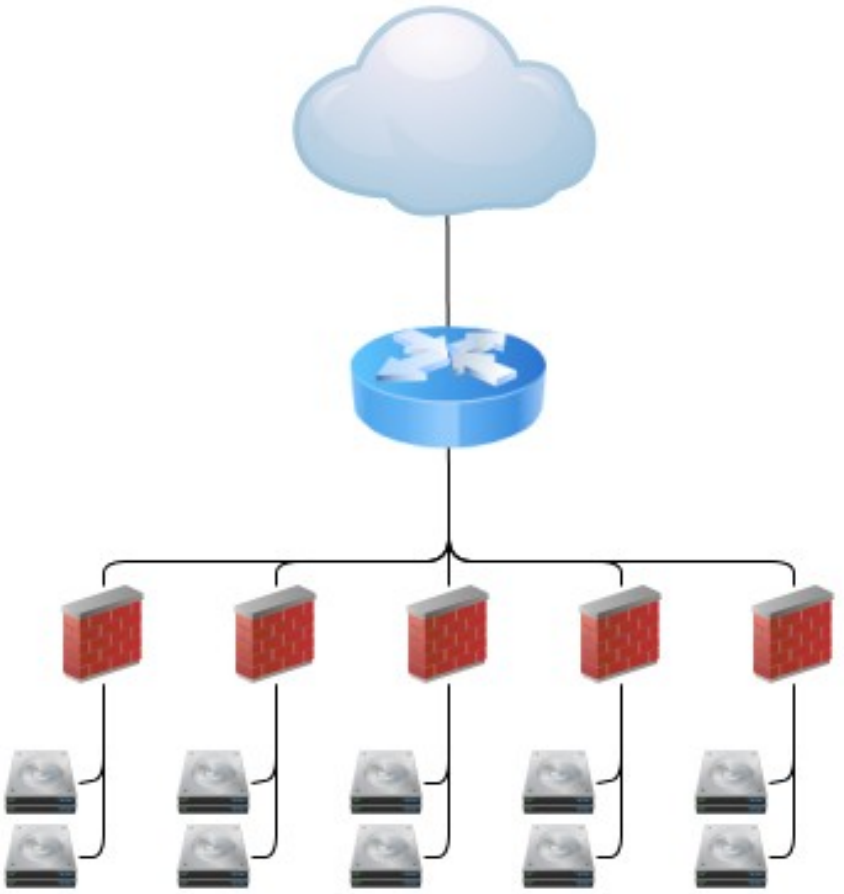
Firewall – ideal

Avantage

- Chaque zone avec un firewall en entrée et en sortie

Inconvénient

- Besoin de pleins de firewall
- Gestion décentralisée



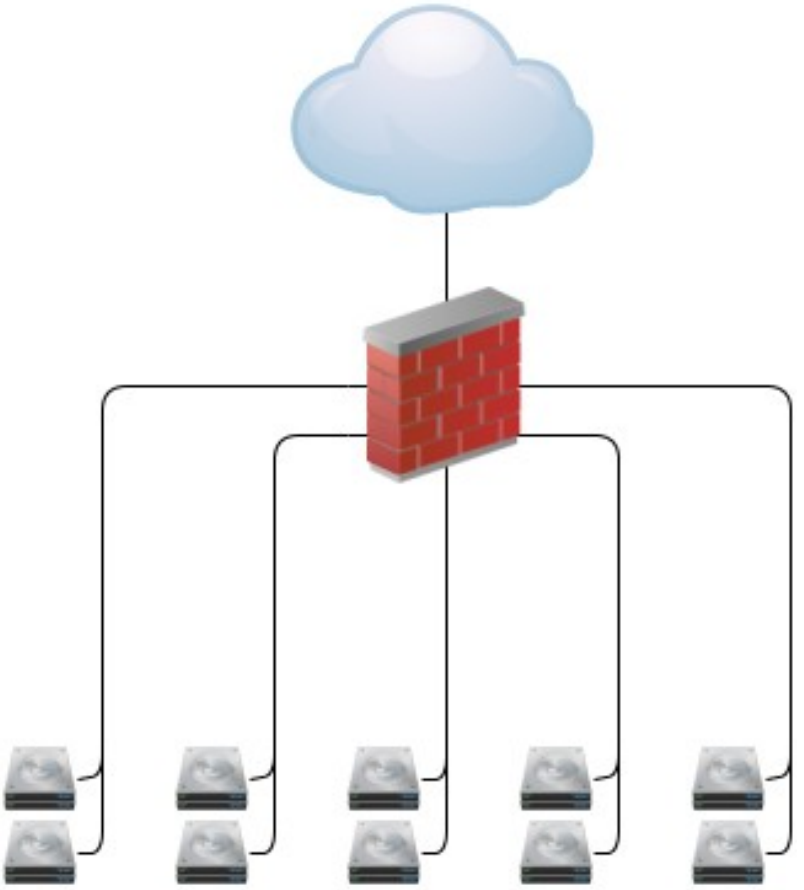
Firewall – tout en un

Avantage

- Un seul firewall
- Configuration centralisée

Inconvénient

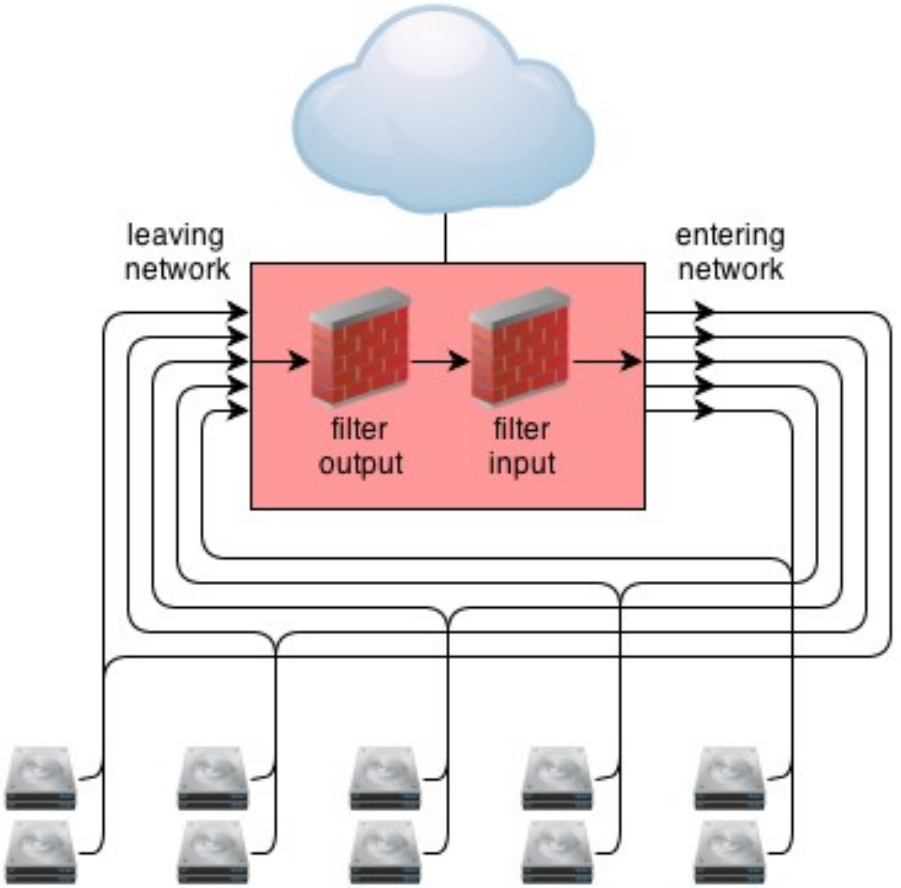
- Difficile de filtrer correctement les entrées et sorties de chaque zones, sans risque de by-pass involontaire



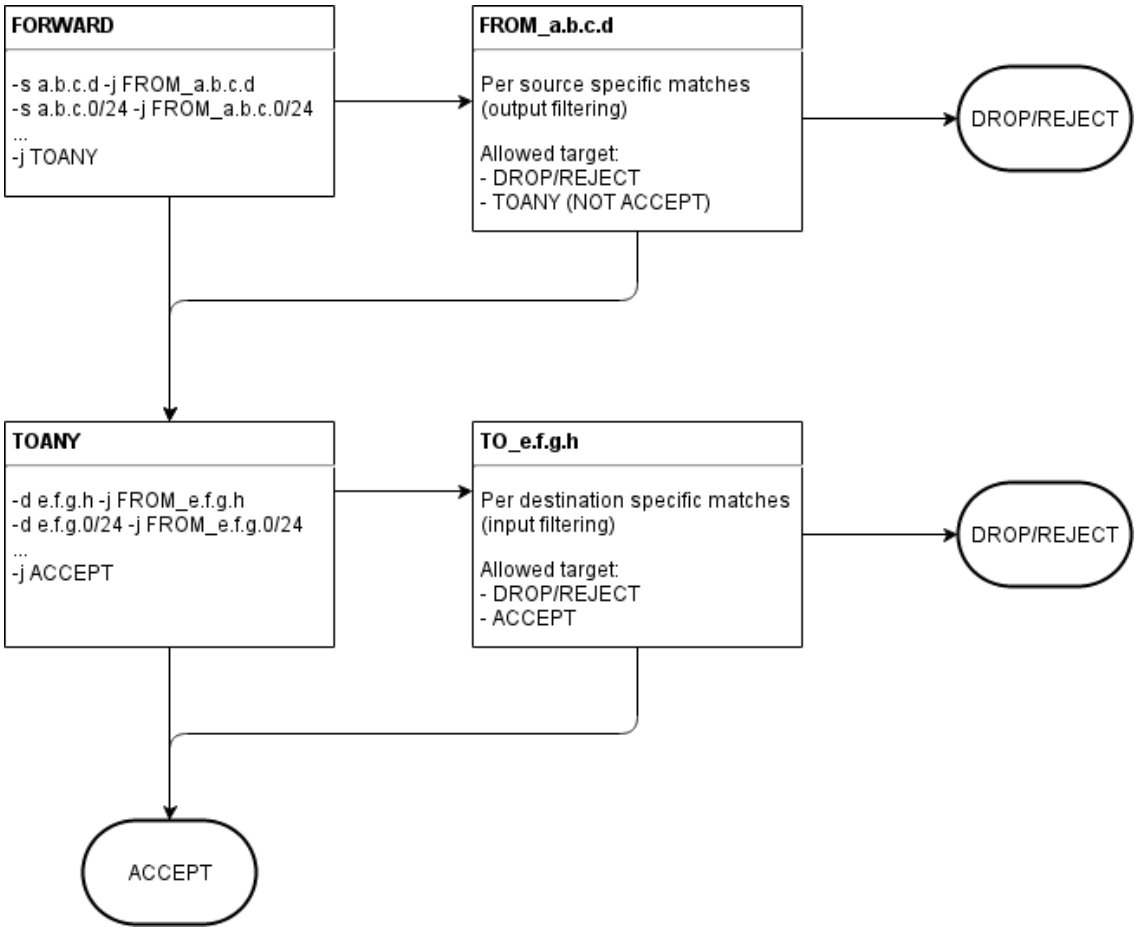
Firewall – besoin

Besoin

- Pouvoir filtrer la sortie de chaque réseau
- Pouvoir filtrer l'entrée de chaque réseau
- Le tout en un



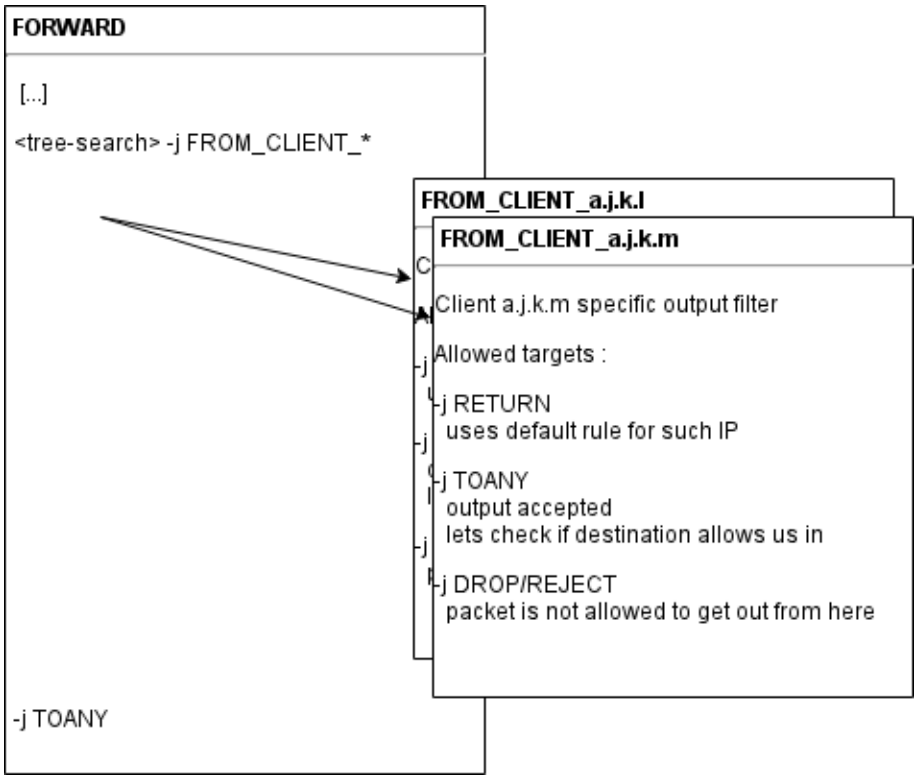
Firewall – architecture



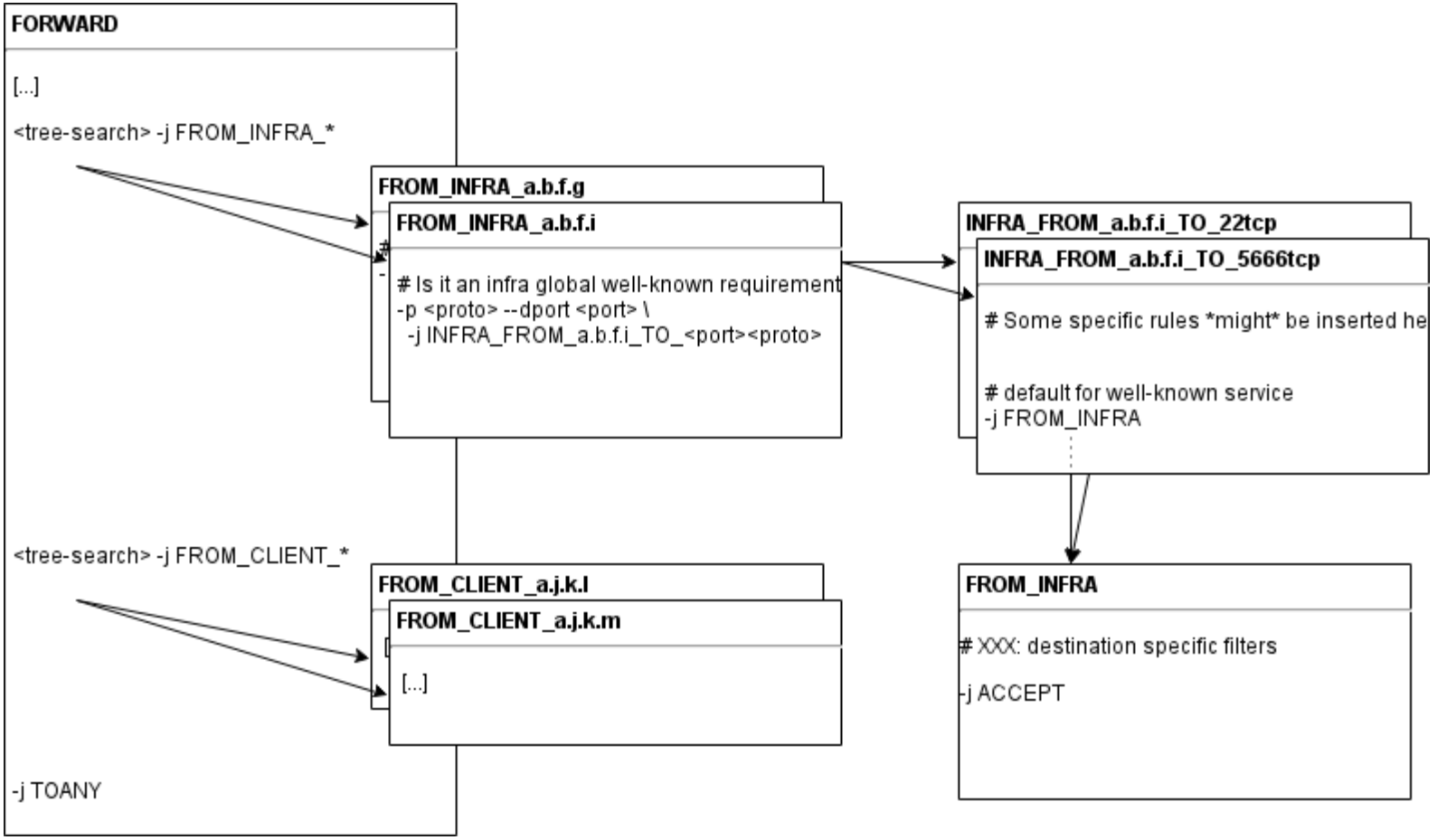
Firewall – architecture

FORWARD

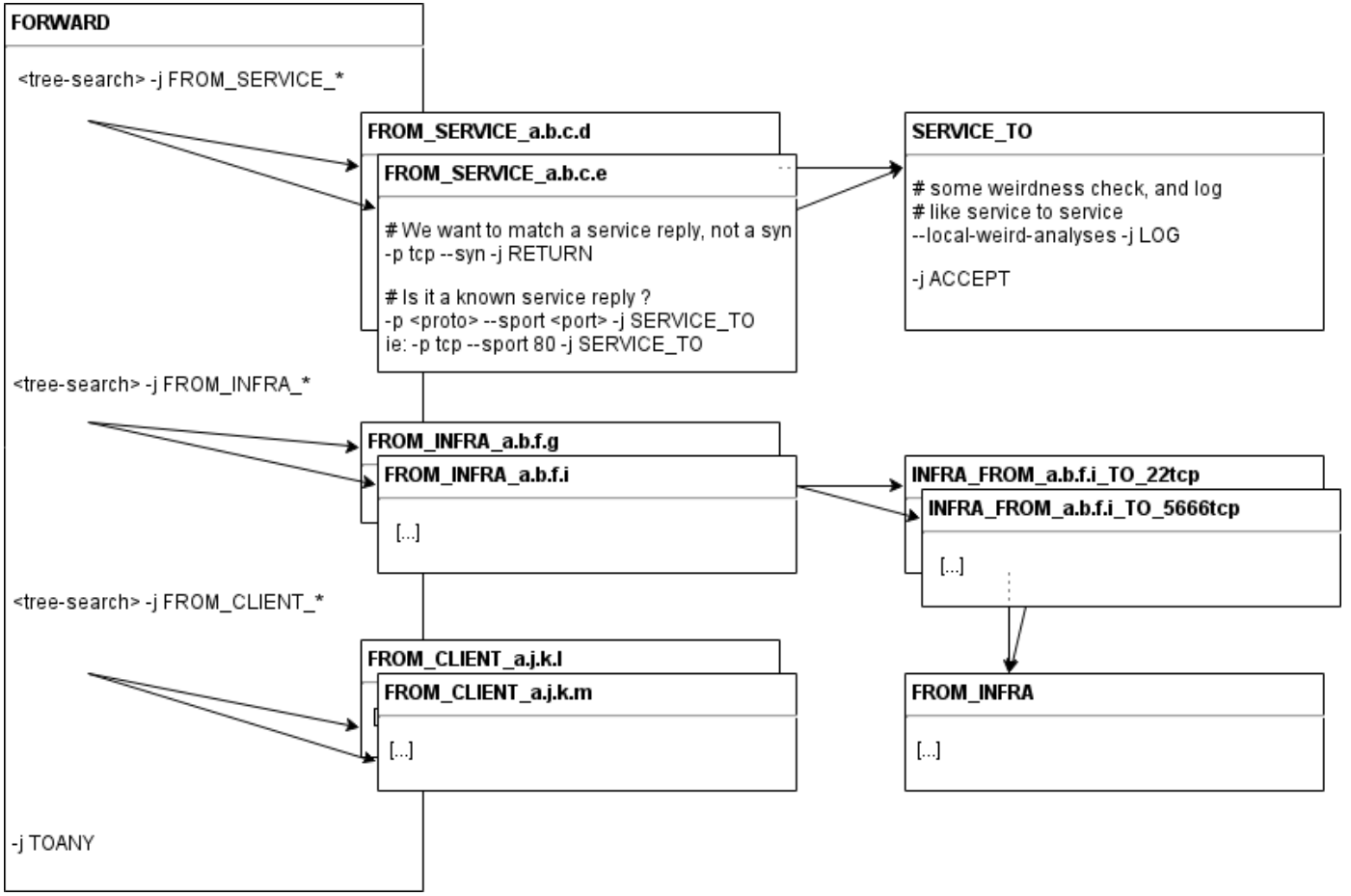
Vérification en sortie de la zone



Firewall – architecture



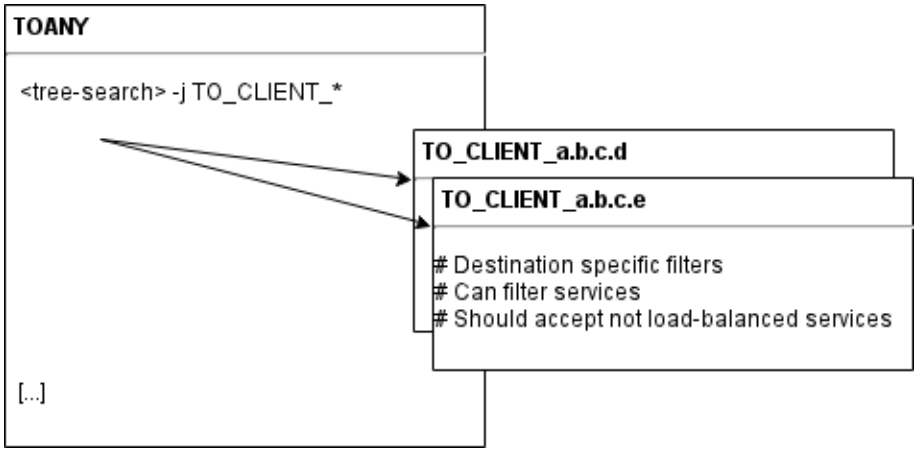
Firewall – architecture



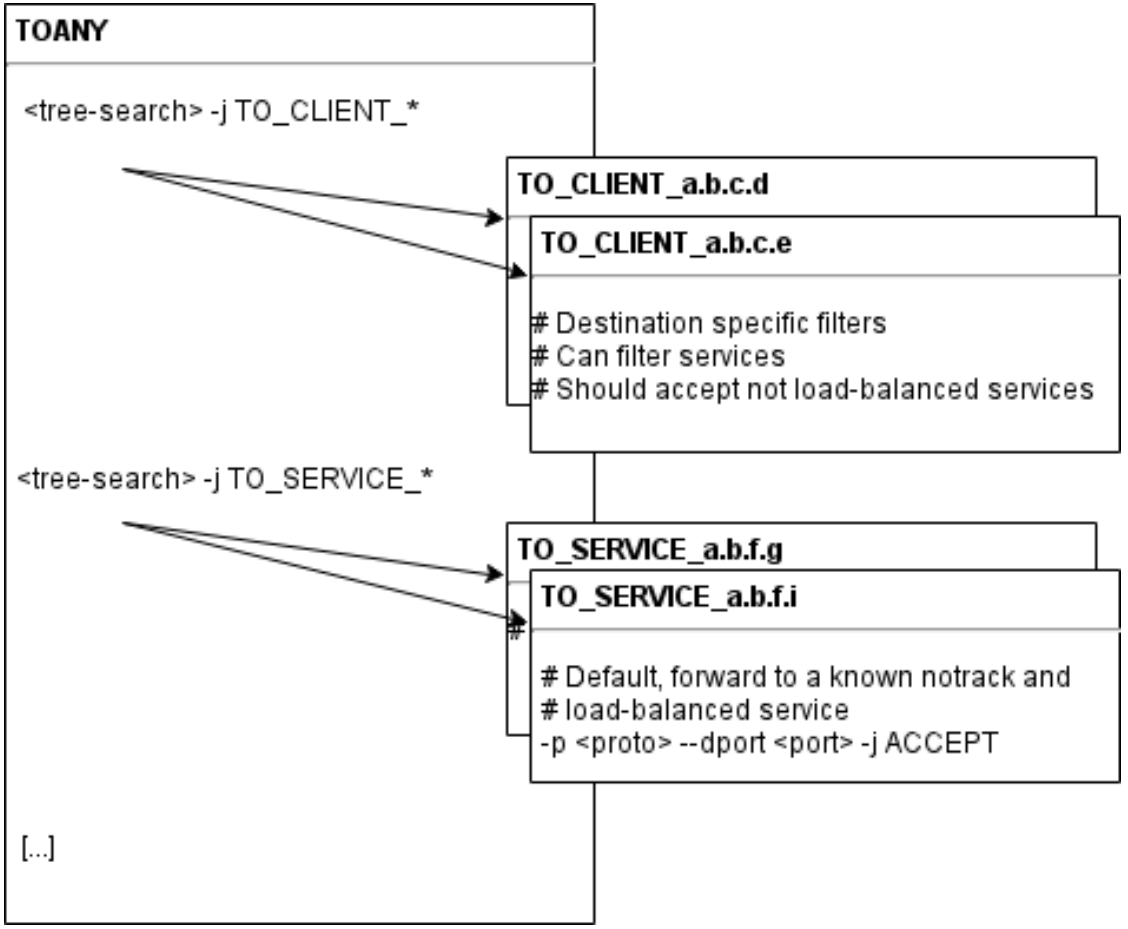
Firewall – architecture

TOANY

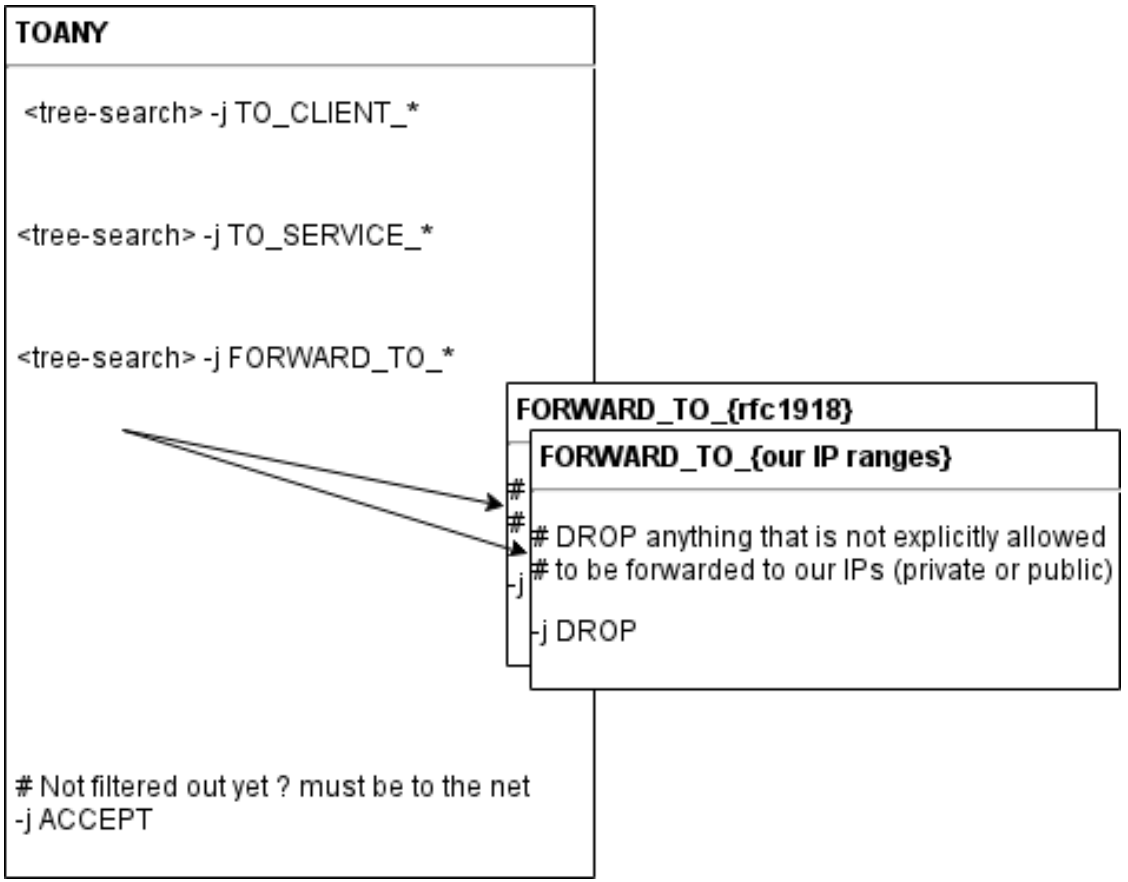
Vérification en entrée de zone



Firewall – architecture



Firewall – architecture



Firewall – génération

Besoins

- Gestion des *chains*
- Pouvoir générer un tas de règles dynamiquement (template, scripting)
- Pouvoir générer des arbres (via scripting externe)
- Modulaire (un fichier par client)

Firewall – ferm

FERM

- Gère correctement les chaines
- Possibilité de définir des fonctions
- Système d'include avec wildcard
- Possibilité d'appeler des scripts externe pour générer des morceaux de configuration

Firewall – petit manque

FERM

- Gère des variables type tableau, mais ne permet pas de boucler dessus

Firewall – cycle de génération

- Construction de l'ensemble des règles sans arbres (donc, il manque des jumps...)
- Construction des règles en générant les arbres en fonctions des chaines qui ont été créés lors du premier passage
- Renommage directement dans le résultat pour les noms de chaines trop longs

Prevision d'évolution

- Actuellement en place a un endroit dans l'architecture
- Prévision de migration du reste de l'architecture vers ce système dans les prochains mois
- D'avantage d'automatisation et de configuration identique sur toute l'infrastructure
- Valider l'évolution horizontale de la solution

Questions ?