DE LA RECHERCHE À L'INDUSTRIE

www.cea.fr

Architecture réseau

Découpage, filtrage, redondance et surveillance

Romain Carré

CEA/DAM

Master SeCReTS, 14/11/2018

Sommaire

Définition

L'architecture réseau est l'organisation d'équipements de transmission, de logiciels, de protocoles de communication et d'infrastructure permettamt la transmission des données entre les différents composants. Elle met en jeu des notions de topologie (forme), et de typologie (intégration).

Elle fait notamment intervenir :

- concentrateurs, commutateurs, routeurs
- pare-feu, relai, proxy
- 1000BASE-T, ADSL, Bluetooth, Fibre Optique
- FibreChannel, Ethernet, WiFi, IPv4, TCP, UDP
- étoile, bus, anneau, maillage, *leaf / spine*
- LAN, MAN, WAN

Architecture centralisée

Principe du client / serveur :

- Le serveur met à disposition une ressource (donnée, service)
- Les clients demandent la ressource au serveur
- Pour TCP et UDP, le serveur écoute sur un port donné
- Connexion toujours à l'initiative du client (par convention)
- DHCP, SSH, DNS, NTP, HTTP, FTP...

Dans ce modèle, le serveur est critique :

- Sécurité (mise à jour, réduction de surface d'attaque, **bastion**)
- Disponibilité (tolérance aux pannes, sauvegarde, redondance)
- Transitivité (un serveur peut jouer le rôle de client à son tour)

Modèle décentralisé :

- Pas de serveur unique : les éléments de l'architecture sont tous potentiellement clients et serveurs à la fois
- Calcul parallèle, *peer-to-peer*, mDNS, WiFi *ad-hoc*...
- Avantage : absence de point central critique

Problématiques :

- Fiabilité des différents éléments
- Confiance en chaque élément
- Complexité relative d'administration

Modèles hybrides :

- Cluster (DB), Tor, « Cloud »

Pourquoi faire de l'architecture réseau ?

Avantages :

- **Administration :** regrouper les machines en zones homogènes
- **Cloisonnement :** séparer les différentes populations d'utilisateurs
- **Sécurité** : mieux filtrer et maîtriser les flux entre les zones / machines
- **Performances :** limiter les flux inter-zones pour ne pas saturer les liens
- **Fonctionnalités :** certaines abstractions permettent plus de souplesse

Contraintes :

- **Financières :** l'architecture idéale peut coûter cher (très cher)
- **Matérielles :** équipement à disposition, taille des locaux, normes
- **Humaines :** technicité, complexité des architectures, difficulté d'exploitation

→ **Il faut trouver le bon compromis entre une architecture idéale et réalisable.**

Cloisonnement : principes

Travail préliminaire :

- Regroupement des machines en zones (délimitation de périmètres)
- Définition de chaque zone et son rôle (stations, DMZ, extérieur, ...)
- Définition des flux nécessaires entre chaque zone
- Définition des moyens à mettre en place pour :
  - Le cloisonnement (pare-feu, ACLs)
  - Surveiller le réseau (supervision, logs)

Le cloisonnement permet :

- De restreindre les accès et donc les risques de compromission
- De limiter les risques de propagation de *malwares*
- D'isoler après-coup une zone contaminée (mise en quarantaine)
- De définir des politiques plus fines pour chaque zone (intra / inter)
- D'étendre plus aisément le système d'information (scalabilité)

Cloisonnement : principes

Principe de l'unicité des rôles :

- une machine = une et une seule fonction (serveur dédié)
- Une zone réseau est constituée de machines ayant :
  - des rôles similaires
  - des besoins comparables

Principe de moindre privilège :

- Seulement les flux strictement nécessaires sont autorisés
- Tous les autres sont interdits

Réalisation :

- Sous-réseaux IP, VLANs
- Filtrage, Routage

→ **Exemple concret au tableau : DNS, mail, web, intranet, SSH**

Découpage selon l'accessibilité

Pour une machine donnée, les questions à se poser :

- Accessible depuis l'extérieur ? (serveurs publics)
- Accessible depuis le réseau interne ? (clients, serveurs internes)
- Besoin d'un accès à internet ? À des zones externes ?
- Besoins spécifiques de services du réseau interne ?
- Précautions particulières en terme de sécurité ?
- Flux spécifiques ? (NTP, VRRP, IPSec, zones expérimentales, ...)

En fonction des réponses :

- Adressage public ou privé
- Placement derrière un proxy, reverse-proxy, pare-feu

Les besoins sont intimement liés à la fonction de la machine.

Découpage selon la fonction

Types de machines :

- Station (poste client, porte d'entrée potentielle)
- Serveur (à dissocier de son interface DRAC)
- Équipement réseau (interface d'administration)
- Équipement de sécurité (pare-feu, proxy)
- Équipement industriel (flux très spécifiques)

Au sein des serveurs, différentes fonctions :

- Authentification (LDAP, Kerberos, Radius)
- Stockage et Sauvegarde (NFS, SMB, *appliances*)
- Mail, DNS (publics, internes)
- NTP, DB, HTTP, Supervision, ... multitude des genres !

Découpage selon la population

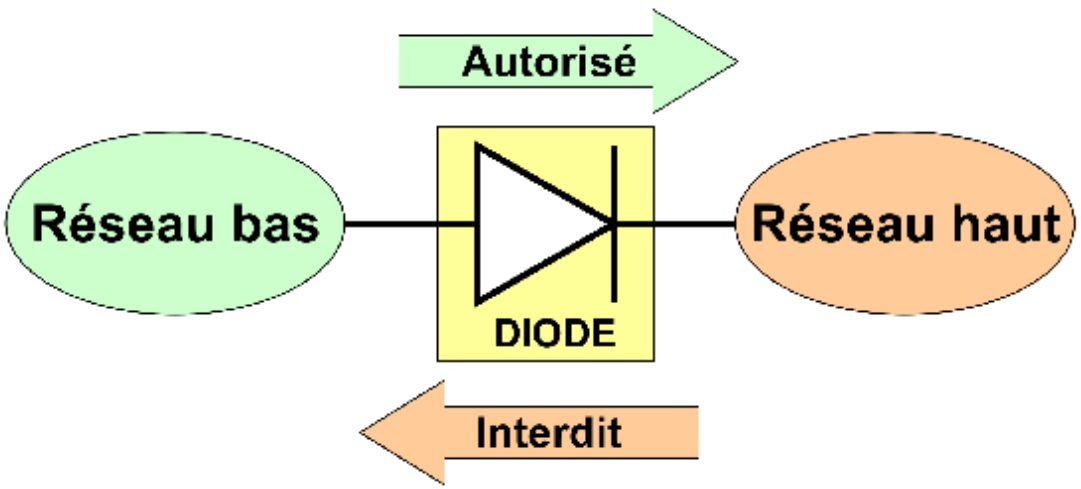
Types d'utilisateurs :

- Utilisateurs classiques, avec des problématiques uniques :
  - des contraintes de disponibilité forte ($\rightarrow$ *hotline*)
  - pas informaticiens *a priori*, vulnérables (surf, mails)
  - certains ont un usage à la limite de la malveillance
- Personnel administratif : restriction aux suites bureautiques (qui souvent travaillera d'ailleurs sous Windows, ce qui implique une série de services propres : Exchange, AD, KMS, WSUS, ...)
- Stagiaires, thésards, prestataires (flux « expérimentaux »)
- Personnel extérieur à l'entreprise (postes dédiés, AP WiFi)
- Administrateurs (Réseau, Système, Sécurité)

À chacune de ces populations correspondra des accès différents.

Couche Physique [OSI 1] :

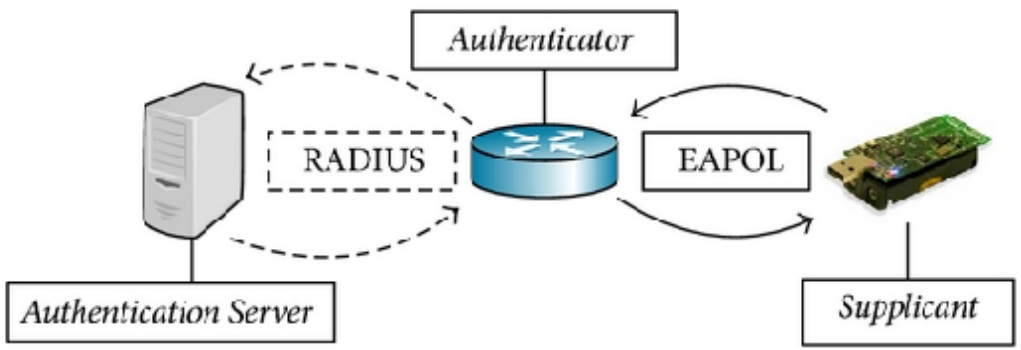
- Filtrage simpliste, analogue au composant électronique
- Ne laisse passer le flux que dans un seul sens
- Ne permet d'utiliser que des protocoles non connectés
- Hairgap : `https://github.com/cea-sec/hairgap`



Authentification EAP

Couche Liaison [OSI 2] :

- Protocole d'authentification extensible de matériel communiquant
- Permet de déterminer si un équipement est reconnu sur le réseau
- Filtrage possible, relativement moins simpliste qu'avec MACSEC
- Il consiste à autoriser ou refuser l'établissement de connexion
  - EAP-TLS pour WPA / WPA2 (basé sur TLS et les certificats x509)
  - EAP-SIM pour la téléphonie mobile sur couche 1 GSM
  - EAP-GTC pour les *token cards* (basé sur OTP)



Couche Réseau [OSI 3] :

- Protocole réseau d'amorçage qui permet de découvrir sa propre IP
- DHCP peut filtrer les équipements sur leur adresse physique

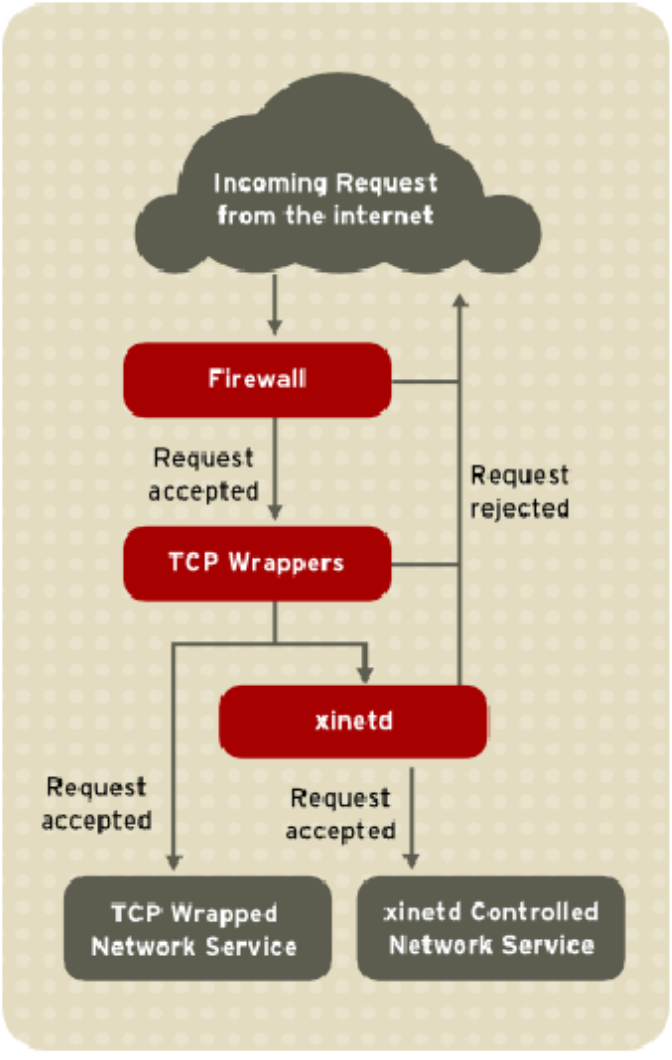
On peut imaginer une configuration en pseudo-langage comme celle-ci :

```
subnet 192.168.0.0 netmask 255.255.255.0 {
    range 192.168.0.10 192.168.0.19;
    host pc     {
        hardware ethernet 01:02:03:04:05:06;
        fixed-address 192.168.0.1;
    }
    host pirate {
        hardware ethernet 01:02:03:07:08:09;
        deny booting;
    }
}
```

Couche Transport [OSI 4] :

- Système de contrôle d'accès pour des démons réseau (sur TCP)
- S'intercale entre le pare-feu et les services sur la machine
- Complètement transparent pour les deux extrémités (invisible)
- Permet d'autoriser ou de refuser l'accès à une ressource
- Permet en autres fonctionnalités de bénéficier du *Port Knocking*
- Configuration du super-serveur *tcpd* dans les fichiers `/etc/hosts.allow` et `/etc/hosts.deny`

```
ALL : LOCAL
in.ftpd : 192.168.0.1
in.sshd : .uvsq.fr
```



Si le démon est lié à la bibliothèque *libwrap.so*, on peut l'instancier à travers *xinetd* (appelé super-démon). Sinon il est autonome, et sera directement appelé via *tcpd*.

Pare-feu (*firewall*) [OSI 3/4] :

- Filtrage de flux entre deux machines / zones :
  - Paquets TCP port 22 (SSH) allant de Z_1 vers Z_2
  - Paquets UDP port 53 (DNS) allant de Z_1 vers le serveur DNS de Z_2
  - Paquets TCP dont l'établissement de la connexion a déjà été autorisé
- Traduction d'adresses et de ports (NAT) :
  - Les machines de la zone Z_1 seront vues comme l'IP I_2 depuis l'extérieur
  - Les paquets à destination de I_2 port P_2 seront reconstruits pour être ré-acheminés vers I_3 port P_3
- Action jusqu'à la couche 4 inclue (TCP, UDP, SCTP, VRRP, ESP, ...)
- Normalement, pas de connaissance de la couche applicative
  (pas de filtrage sur des critères HTTP par exemple, ~~*proxy*~~)
  Quelques exceptions cependant si négociation de ports (FTP)

Types de pare-feu :

- ***Stateless*** : pas de notion d'état, tous les paquets sont traités indépendamment les uns des autres (raison historique, implémentation rapide et concise, utilisé dans l'embarqué quand les ressources mémoire et CPU sont limitées, mais **à proscrire** au XXIe siècle : certaines **attaques et techniques d'évasion** sont rendues possibles grâce à ce genre d'équipement)
- ***Stateful*** : gestion des états, connexions TCP, de certains protocoles basés sur UDP, défragmentation IP, possibilité de mettre en place de la *QoS*...

Politique par défaut :

- ***Permis par défaut*** : tout est autorisé par défaut, seuls certains flux considérés dangereux sont interdits (si l'administrateur est légèrement laxiste ou oublie quelque chose, c'est la compromission assurée → **dangereux**)
- ***Interdit par défaut*** : tout est interdit par défaut, les flux autorisés sont alors explicitement spécifiés (ce qui est une politique largement préférable)

Types de filtrage

Globalement basé sur les couches 3 et 4 :

- Type de protocole (TCP, UDP, ICMP, ESP, VRRP, ...)
- Adresses IP source / destination (ou interface réseau)
- Ports source / destination pour TCP et UDP
- États des connexions / échanges (pour les *stateful*)

On trouve aussi parfois :

- Champs IP spécifiques (fragmentation, TTL...)
- Flags ou options TCP
- Types ICMP

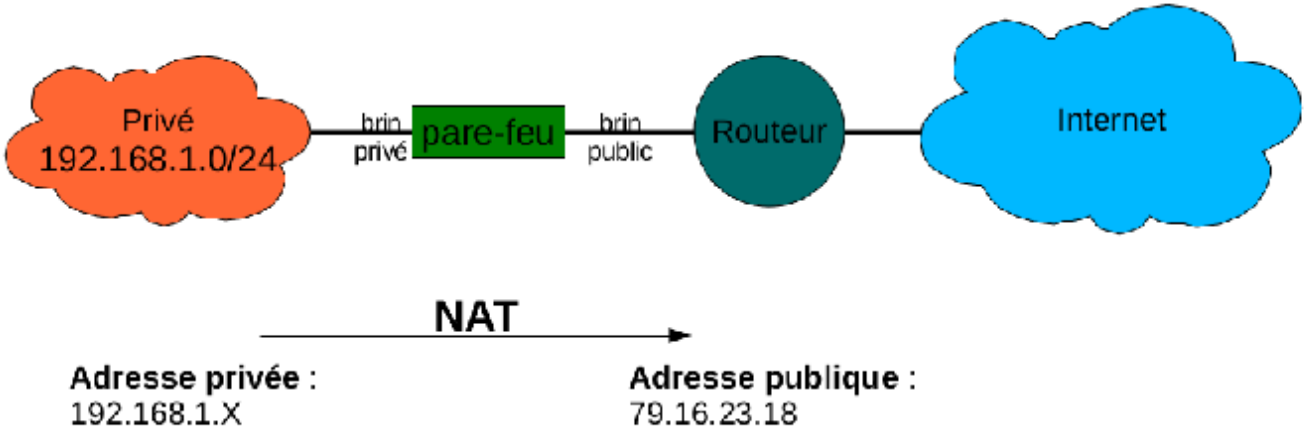
Ainsi que des fonctionnalités secondaires :

Qualité de Service (*QoS*), Journalisation (*logs*), Blocage évolué (*return-RST*), ...

NAT (*Network Address Translation*)

Types de NAT :

- **SNAT** (Source NAT) : aussi appelé simplement NAT. L'adresse source est modifiée (utile pour faire communiquer des plages non routables, ou quand on veut masquer la topologie interne d'une DMZ par exemple)
- **DNAT** (Destination DAT) : aussi appelé RDR. L'adresse destination (celle qui est la destination du premier paquet) est modifiée (globalement pour donner une IP visible « publique » à un service sur une IP dite « privée »)
- Les paquets retour sont modifiés en conséquence (translation inverse).



Pare-feux usuels

NetFilter :

- Intégré au noyau Linux depuis la version 2.4
- Outil de configuration en espace utilisateur : `iptables`
- Modulaire, Gestion poussée des protocoles applicatifs

PacketFilter :

- Intégré aux noyaux *BSD depuis la version 3.0
- Outil de configuration en espace utilisateur : `pfctl`
- Monolithique, Langage puissant mais limité aux couches 3 / 4

Alternatives commerciales :

- Check Point, Cisco, Fortinet, Juniper, Sophos, Stormshield, ...
- Intègrent souvent des fonctionnalités supplémentaires : IPS, proxy
- Souvent trouvés sous forme d'*appliance* (machine livrée installée)

Mandataire (*proxy*)

Principe :

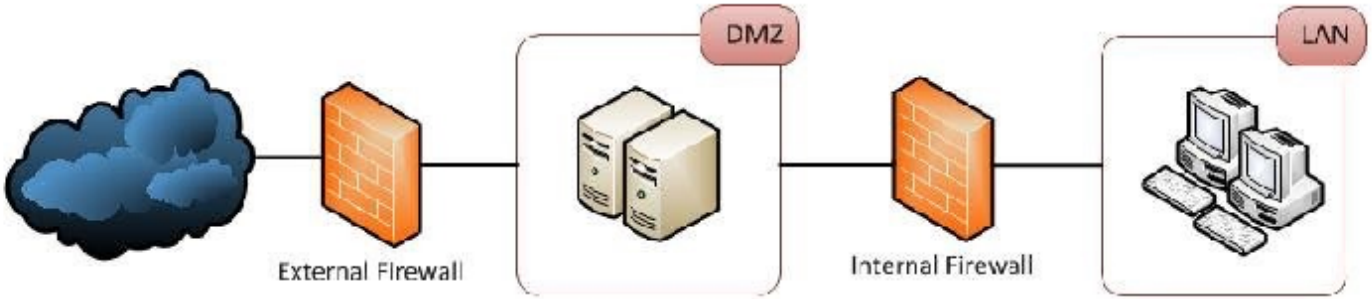
- Peut filtrer / modifier la **Couche Applicative [OSI 7]** des paquets reçus
- Le client demande la ressource au proxy plutôt qu'au vrai destinataire
- Si le protocole concerné impose de mentionner le destinataire réel (comme HTTP ou FTP), alors il est possible que le client utilise un proxy sans même en avoir connaissance (on parle alors de **proxy transparent**)

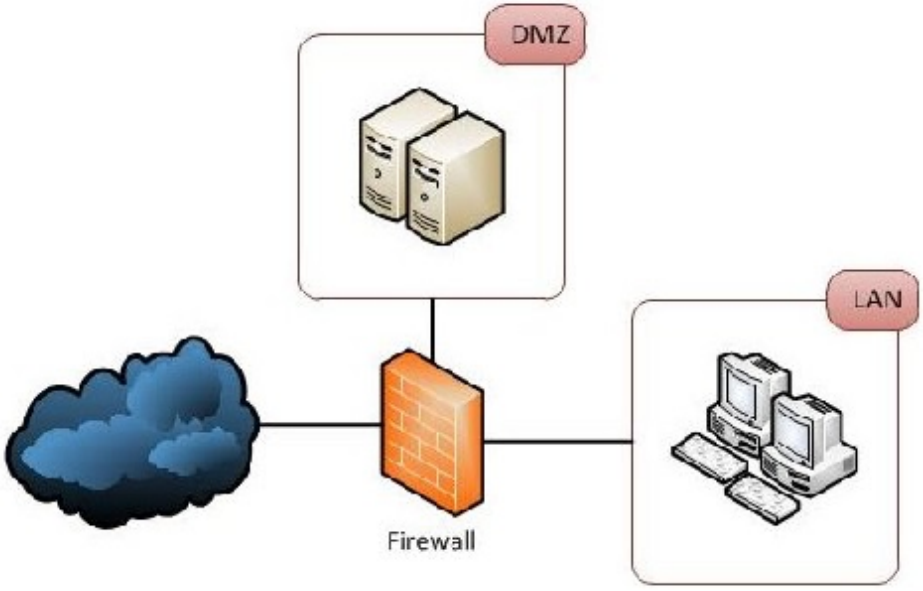
Rôle :

- Donner aux clients un accès à un service applicatif sans leur permettre un accès réseau direct (utile pour éviter les « *from any to any port http* »)
- Filtrer et journaliser les accès à du contenu (URL HTTP, type MIME, ...)
- Améliorer les performances grâce à un système de cache

Exemples : `squid` (proxy HTTP), `dnscache` (proxy DNS), protocole SOCKS

DMZ (*DeMilitarized Zone*)





DMZ (*DeMilitarized Zone*)

Principe :

- Certains services doivent être accessibles depuis plusieurs zones réseau de sensibilités différentes (c'est notamment souvent le cas d'un serveur mail)
- DMZ : zone « tampon » particulière où sont placés ces types de services
- Nécessité potentielle de créer plusieurs DMZ en fonction de leur niveau de sensibilité ou du sens usuel des flux (DMZ entrante, sortante, ...)

Les serveurs de la DMZ sont souvent exposés au monde extérieur :

- Sécurité renforcée (confidentialité, intégrité, disponibilité)
- Blindage du système (bastion) : mises à jour, exposition minimale
- Filtrage ultra strict (éventuellement complété par un pare-feu local)
- Surveillance (logs, IDS), Redondance, etc.

Redondance

Objectif : améliorer la disponibilité d'un service

- *failover* : tolérance à la panne d'un noeud sur le réseau
- *loadbalancing* : répartition de charge, absorbe les pics

À plusieurs niveaux :

- Géographique : duplication / distribution de salles machines / sites
- Infrastructure : double alimentation électrique, production de froid, ...
- Stockage : disques en RAID-1, *NetApp Metro Cluster*, *glusterfs*, ...
- Réseau : *bonding* local d'interfaces réseau, doublement d'opérateurs, utilisation de protocoles de redondance, duplication d'infrastructures, ...
- Service :
  - Plusieurs serveurs quasi-identiques assurent la même fonctionnalité
  - Notion de maître-esclave (ou de services primaire et secondaires)
  - Lorsque le maître ne peut fournir le service, l'esclave est interrogé

Routage statique :

- l'administrateur réseau doit paramétrer les routeurs pour leur donner des ordres de routage : sur quelle interface envoyer les datagrammes
- c'est une modification statique de la table de routage des routeurs
- c'est fastidieux, pas très efficace, et ne convient qu'à de petites structures
- si la configuration du réseau change : incident, coupure, changement de matériel, surcharge, alors il faut, pour maintenir la continuité de service, que chaque routeur adapte sa table de routage à la nouvelle configuration

→ **Routage Dynamique**

- routage externe **EGP** (dont BGP, avec extension multi-layer MPLS)
- routage interne **IGP** (dont OSPF)

BGP - *Border Gateway Protocol* (Couche Réseau **[OSI 3]**) :

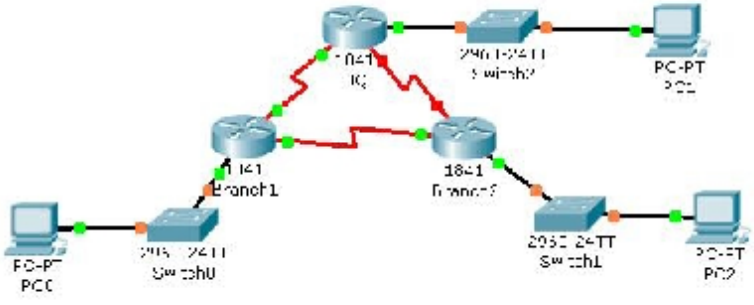
- permet de router les AS (*autonomous system*) entre eux (uniquement)
- n'utilise pas de métrique, contrairement aux protocoles de routage interne
- fonde les décisions de routage sur :
  - les chemins parcourus, ainsi que les attributs des préfixes
  - un ensemble de règles de sélection définies par l'administrateur de l'AS
- on le qualifie de protocole à vecteur de chemins (*path vector protocol*)
- les erreurs de configuration sont globalement cataclysmiques :
  - incident YouTube Pakistan en 2008
  - incident attribut BGP expérimental en 2010

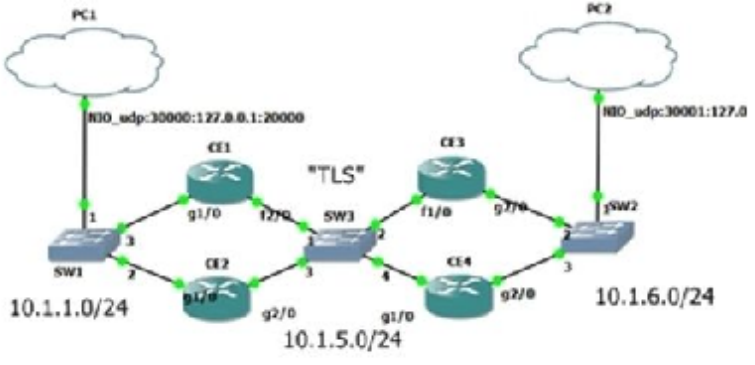
```
R4#show ip bgp summary
BGP router identifier 40.0.2.1, local AS number 100
BGP table version is 1, main routing table version 1

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
14.0.0.1        4   300       7       7        1    0    0 00:03:18        0
```

OSPF - *Open Shortest Path First* (Couche Réseau **[OSI 3]**) :

- Algorithme de routage hierarchique basé sur *Dijkstra* (plus court chemin)
- Gère la redondance dynamique des chemins pour atteindre une route
- Si un noeud est perdu, la convergence rétablit la connectivité
- Quand des chemins ont le même coût (ECMP), *loadbalancing* possible
- On passe alors d'un arbre couvrant à un graphe couvrant





FHRP (*First-Hop Redundancy Protocols*)

Exemples :

- HSRP : *Hot Standby Router Protocol* (Cisco)
- GLBP : *Gateway Load Balancing Protocol* (Cisco)
- VRRP : *Virtual Router Redundancy Protocol* (HSRP+)
- CARP : *Common Address Redundancy Protocol* (libre)

Principe :

- Mécanisme de redondance réseau par élection de passerelle
- Opère également au niveau IP, par encapsulation dans une 4e couche
- Bascule sur un équipement opérationnel si le *master* actuel est défaillant
- Partage des tables d'états *via* d'autres protocoles (`pfsync` pour PF)
- Mode *failover* pour les 4, avec *loadbalancing* pour GLBP et CARP

FHRP (*First-Hop Redundancy Protocols*)

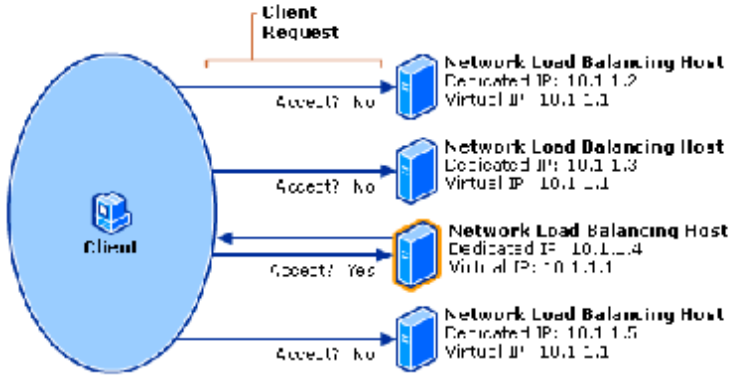
Fonctionnement :

- Partage d'une (ou plusieurs) IP(s) virtuelle(s) entre les noeuds
- Élection du *master* grâce à un message en multicast (224.0.0.X)
- Seul le *master* répond sur l'IP partagée
- Changement de *master* en cas de défaillance
- Les alias virtuels portés doivent être exactement identiques
- Notion de préemption pour faire basculer les interfaces simultanément
- Possibilité d'attribuer des poids aux noeuds pour influencer l'élection
- Mécanisme d'authentification pour éviter le *hijacking* de passerelle

→ Le délai de bascule dépend de la fréquence du *probing* en multicast.

NLB - *Network Load Balancing* (Couche Réseau **[OSI 3/4]**) :

- Protocole implémenté par le *Windows Load Balancing Service* (WLBS)
- Restreint à la répartition de charge TCP / IP uniquement
- Toutes les machines du cluster doivent être sur le même sous-réseau
- Elles partagent une IP (et MAC) commune sur une interface virtuelle
- Ne permet de redonder que des services *stateless* (mail, web)
- Pas de *gateway* → Utilisation d'un *hub* (un *switch* masquerait le trafic !)



Équivalents améliorés

Problèmes des méthodes précédentes :

- Elles ne testent pas réellement un service mais une adresse IP
- Exemple : pinger 193.51.33.8 pour savoir si le serveur HTTP (TCP/80) donne bien la page de garde de `www.uvsq.fr` (non pertinent)

→ Solution : incorporer des informations applicatives pour affiner les décisions.

Relayd : redondance applicative sur une passerelle OpenBSD

- DNAT le trafic vers le *master*
- Choix en mode *failover*, *roundrobin*, *source-hash*, *least-states*, *random*
- Déterminé sur la base de critères applicatifs (GET / → HTTP/200)

Keepalived : redondance applicative autonome

- Adresse IP virtuelle migrante (principe de la patate chaude)
- Les noeuds communiquent entre eux avec VRRP pour l'élection

Au niveau applicatif

Pour certains services, la redondance fonctionne « toute seule » (*by design*).

RRDNS : *Round-Robin DNS*

- Régulièrement utilisé par des services massifs (*Google*, *Youtube*, *Twitter*)
- Une requête DNS ne renvoie pas constamment la même réponse
- **Avantage** : fonctionne sans modification de structure du réseau
  (et les machines n'ont pas besoin non plus d'être sur le même sous-réseau)
- **Inconvénient** : le délai avant récupération sur faute est intrinsèquement lié à la propagation des enregistrements et à la durée de validité du cache (TTL)
- En pratique, il s'agit de plusieurs enregistrements A pour un même nom DNS

MySQL :

- Mécanisme propre de synchronisation des données entre tous les serveurs
- Les requêtes peuvent être traitées par n'importe lequel indépendamment

Parce qu'on est quand même au XXI^{e} siècle, quelques mots sur `IPv6` :

- conception finalisée en 1998, standardisé dans la RFC 8200 en 2017
- né du constat de l'épuisement des plages d'adresses disponibles en IPv4
- une adresse IPv6 est longue de 128 bits (16 octets), au lieu de 32 bits
- exemple : `2001:0db8:0000:85a3:0000:0000:ac1f:8001`
- le système de translation d'adresse donc n'est plus nécessaire
- pas de somme de contrôle dans l'entête, la longueur n'inclut pas l'entête
- entêtes optionnels pour la fragmentation et les décisions de routage
- la fragmentation ne se fait plus sur les routeurs (grâce à `ICMPv6`)
- adaptation des protocoles : GRE, DHCPv6, DNS, BGP, OSPF, TCP, ...
- la plupart des FAI sont maintenant compatibles IPv6 et routent le trafic
- la loi (2016-1321 art.16) encourage la migration des administrations

Surveillance / Monitoring

Objectifs :

- Surveiller l'état global du réseau :
  - Routes, états des passerelles (*master* / *slave*), équipements
  - Espace disque, charge CPU, charge réseau, uptime, ...
  - Disponibilité (état et temps de réponse des services)
  - Performances (latence, bande passante, vidage des files)
  - Anomalies (changement de comportement, configuration, utilisation)
- Détection d'intrusion :
  - *Intrusion Detection System* IDS (Snort, Suricata, AV, ...)
  - Flux réseau bloqués (logs du pare-feu, ACLs des routeurs)
  - URLs ou domaines bloqués (logs du proxy, des serveurs DNS)
- Réponse à incident :
  - Récupération d'informations (fichiers transférés par exemple)
  - Archivage des flux réseaux, *Netflow*, des logs des machines
  - Investigation (*forensics*, *passive DNS*, corrélation de cas)

Surveillance / Monitoring

On ne parlera pas ici des logs système (qui ne concernent pas directement le réseau), mais on y reviendra dans le cours de Sécurité Système, car cruciaux.

Outils connus pour la surveillance :

Nagios, *Splunk* (commercial), ou la suite ELK (*Elasticsearch*, *Logstash*, *Kibana*)

Adaptation aux spécificités :

- Surveillance des taux de *download* / *upload*
- Surveillance de la messagerie si elle est critique
- Tentatives d'authentification échouées
- Analyse de comportements inattendus *via* les logs
- Sur une plateforme de commerce électronique en ligne :
  nombre de transactions, montant moyen, paiements refusés, ...

Scan du réseau :

- `Nessus`, `OpenVAS`, `MBSA` : scan authentifié de vulnérabilités
- `nmap` / `masscan`, `IVRE` : scan réseau non authentifié

Outils de forensic :

- `Autopsy` / `Sleuthkit`

Des questions ?

Commissariat à l'énergie atomique et aux énergies alternatives CEA/DAM
Centre de Bruyères-le-Châtel | 91297 Arpajon Cedex
T. +33 (0)1 69 26 40 00 | F. +33 (0)1 69 26 40 00
Établissement public à caractère industriel et commercial
RCS Paris B 775 685 019