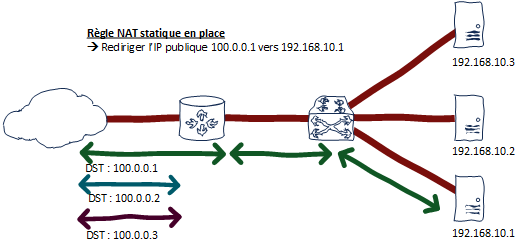
1. **Network Address Translation**

**Le NAT statique :** Permet donc de rendre une machine présente dans un LAN ou une DMZ disponible depuis [internet](https://www.it-connect.fr/cours-tutoriels/referencement-et-web/internet/). Cela ne va pas dans le sens premier de la création du NAT qui est d'économiser des adresses IPv4 car l'association d'une IP publique vers une IP privée est en un à un. On parle également de **redirection de port** lorsque l'on va rediriger uniquement un port de l'IP externe vers un port (le même ou un autre) d'une IP interne.



**Le NAT Dynamique :**

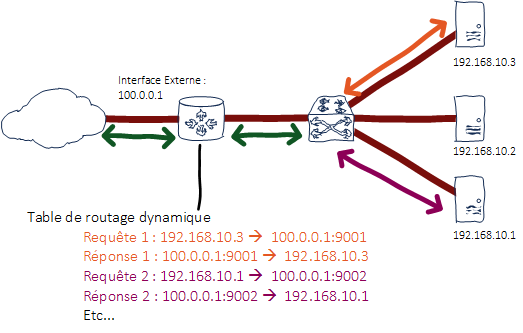
La translation d'adresse dynamique fonctionne elle dans l'autre sens et c'est le but premier de la création du NAT.

Il permet de mettre, aux yeux des éléments qui sont du côté de l'interface externe, un ensemble de machine derrière une ou plusieurs IPs.

Si l'on dispose par exemple d'une plage IP de 8 adresses comme 100.0.0.0/29 sur internet mais que l'on a 500 machines dans notre LAN, nous ne pourrons pas les rendre toutes disponibles sur internet en même temps car à force il n'y aurait plus assez d'adresses IPv4 (il n'en reste déjà plus !).

Le NAT dynamique va alors nous permettre de traduire les 500 adresses IP internes dans le lot des 8 adresses que nous avons sur internet.

Cela est dit "dynamique" car le routeur vont utiliser dynamiquement différentes correspondances IP:ports pour faire suivre les paquets et les échanges qui transitent sur les translations interne <-> externe.



1. **DHCP : Dynamic Host Configuration Protocol**

Il s’agit d’un protocole qui permet à un ordinateur qui se connecte sur un réseau local d’obtenir dynamiquement et automatiquement sa configuration IP

Le but principal étant la simplification de l’administration d’un réseau. On voit généralement le protocole DHCP comme distribuant des adresses IP, mais il a été conçu au départ comme complément au protocole BOOTP (Bootstrap Protocol) qui est utilisé par exemple lorsque l’on installe une machine à travers un réseau (on peut effectivement installer complètement un ordinateur, et c’est beaucoup plus rapide que de le faire en à la main)

Références à DHCP

Les incontournables RFCs :

RFC951 : Bootp

RFC1497 : Options vendor extensions

RFC1541 : Définition du protocole Dhcp

RFC1542 : Interaction entre Bootp et Dhcp

RFC2131 : Complément à la Rfc 1541

RFC2132 : Complément aux options vendor extensions

**Fonctionnement**

DHCP fonctionne sur le modèle client-serveur : un serveur, qui détient la politique d’attribution des configurations IP, envoie une configuration donnée pour une durée donnée à un client donné (typiquement, une machine qui vient de démarrer).

Le serveur va servir de base pour toutes les requêtes DHCP (il les reçoit et y répond), aussi doit-il avoir une configuration IP fixe.

Dans un réseau, on peut donc n’avoir qu’une seule machine avec adresse IP fixe : le serveur DHCP.

Le protocole DHCP s’appuie entièrement sur BOOTP : il en reprend le mécanisme de base (ordre des requêtes, mais aussi le format des messages). DHCP est une extension de BOOTP.

Quand une machine vient de démarrer, elle n’a pas de configuration réseau (même pas de configuration par défaut), et pourtant, elle doit arriver à émettre un message sur le réseau pour qu’on lui donne une vraie configuration. 🡺 La technique utilisée est le broadcast : pour trouver et dialoguer avec un serveur DHCP, la machine va simplement émettre un paquet spécial, dit de broadcast, sur l’adresse IP 255.255.255.255 et sur le réseau local.

Ce paquet particulier va être reçu par toutes les machines connectées au réseau (particularité du broadcast). Lorsque le serveur DHCP reçoit ce paquet, il répond par un autre paquet de broadcast contenant toutes les informations requises pour la configuration. Si le client accepte la configuration, il renvoit un paquet pour informer le serveur qu’il garde les paramètres, sinon, il fait une nouvelle demande.

Les choses se passent de la même façon si le client a déjà une adresse IP (négociation et validation de la configuration), sauf que le dialogue ne s’établit plus avec du broadcast.

**Les baux**

Pour des raisons d’optimisation des ressources réseau, les adresses IP sont délivrées pour une durée limitée. C’est ce qu’on appelle un bail (lease en anglais).

Un client qui voit son bail arriver à terme peut demander au serveur un renouvellement du bail.

De même, lorsque le serveur verra un bail arrivé à terme, il émettra un paquet pour demander au client s’il veut prolonger son bail.

Si le serveur ne reçoit pas de réponse valide, il rend disponible l’adresse IP.

C’est toute la subtilité du DHCP : on peut optimiser l’attribution des adresses IP en jouant sur la durée des baux.

Le problème est là : si toutes les adresses sont allouées et si aucune n’est libérée au bout d’un certain temps, plus aucune requête ne pourra être satisfaite.

Sur un réseau où beaucoup d’ordinateurs se connectent et se déconnectent souvent (réseau d’école ou de locaux commerciaux par exemple), il est intéressant de proposer des baux de courte durée.

A l’inverse, sur un réseau constitué en majorité de machines fixes, très peu souvent rebootées, des baux de longues durées suffisent.

N’oubliez pas que le DHCP marche principalement par broadcast, et que cela peut bloquer de la bande passante sur des petits réseaux fortement sollicités.

**Dynamique ou pas ?**

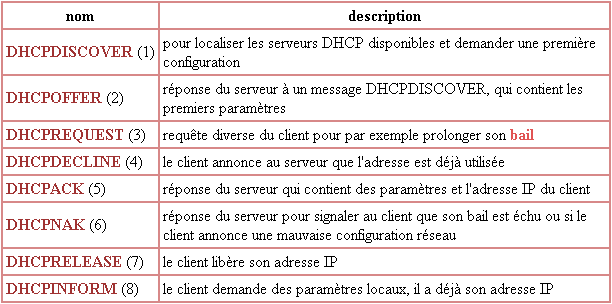
Un serveur DHCP est censé fournir des adresses dynamiques (un même ordinateur peut recevoir successivement 2 adresses différentes), mais il peut fournir une adresse IP fixe à un client bien particulier.

Ceci ne doit être utilisé que de manière modérée, sinon, le serveur DHCP ne sert à peu près plus à rien, mais cela peut se révéler utile pour fournir l’adresse IP au serveur TFTP qui va servir pour le boot à distance des machines.

**Les requêtes et les messages DHCP**

On pourrait croire qu’un seul aller-retour peut suffire à la bonne marche du protocole.

En fait, il existe plusieurs messages DHCP qui permettent de compléter une configuration, la renouveler… Ces messages sont susceptibles d’être émis soit par le client pour le ou les serveurs, soit par le serveur vers un client :



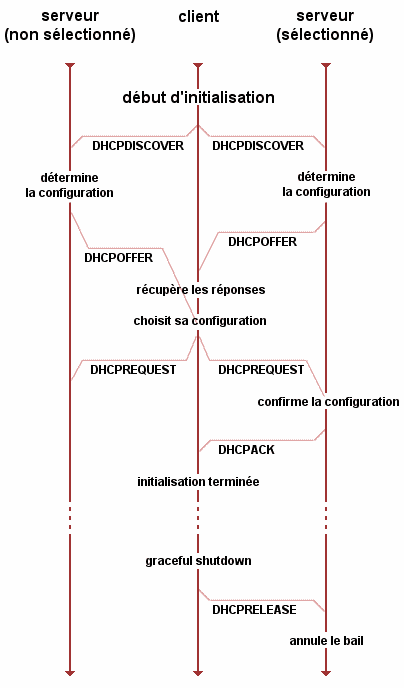
La valeur entre parenthèses est utilisée pour identifier ces requêtes dans les messages DHCP. Voir les options DHCP.

La première requête émise par le client est un message DHCPDISCOVER.

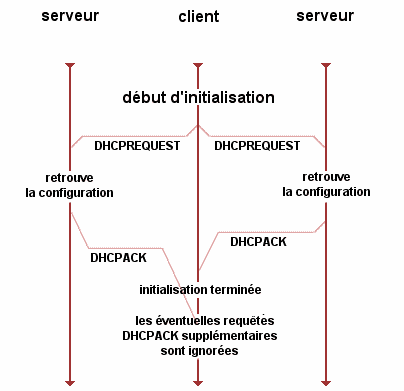
Le serveur répond par un DHCPOFFER, en particulier pour soumettre une adresse IP au client.

Le client établit sa configuration, demande éventuellement d’autres paramètres, puis fait un DHCPREQUEST pour valider son adresse IP.

Le serveur répond simplement par un DHCPACK avec l’adresse IP pour confirmation de l’attribution. Normalement, c’est suffisant pour qu’un client obtienne une configuration réseau efficace, mais cela peut être plus ou moins long selon que le client accepte ou non l’adresse IP ou demande des infos complémentaires…



Pour renouveler une adresse, le fonctionnement est le suivant (les 2 serveurs connaissent le client) :



**Où trouver un serveur DHCP ?**

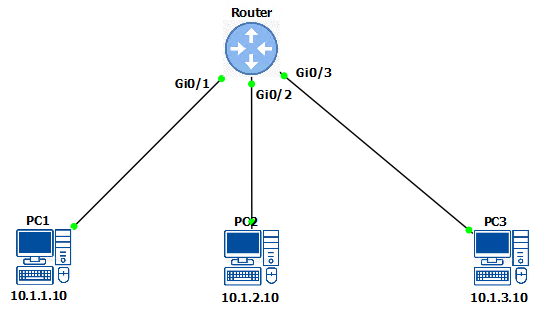
L’Internet Software Consortium développe un serveur DHCP pour le monde du logiciel libre. C’est le serveur DHCP le plus répandu, et celui qui « suit » au mieux les Rfcs. La dernière version en date est la 3.0 et elle est encore en version bêta. Les versions antérieures marchent toutefois très bien, bien que l’ISC sortent beaucoup de patchs. L’une des principales innovations de la version 3 est la possibilité de mettre à jour un DNS dynamiquement en fonction des adresses IP fournies par le serveur DHCP. Pour information, le premier draft sur le DNS dynamique date de mars 1996… Microsoft a bien entendu son propre serveur DHCP pour Windows. Seule la version pour Windows 2000 Server permet de mettre à jour les DNS dynamiquement avec DHCP. Le reste de cette section traite de l’installation et de la configuration d’un serveur DHCP sous-système Unix. L’exemple pris est celui d’un serveur fourni par l’ISC.

**Les ACL standards**

Dans ce type, l’ACL ne peut être liée qu'à l’adresse IP source du paquet. Ces ACLs sont identifiables par identifiant correspondant à un nombre allant de 1 à 99 et de 1300 à 1999.

Nous pourrons utiliser ce type d'ACL pour autoriser ou interdire un segment du réseau ou l'adresse IP d’une machine à communiquer avec un autre segment de réseau ou une autre machine.

Afin de concrétiser la notion d’ACL standard, nous allons les mettre en place sur un routeur Cisco.



Dans l’exemple ci-dessus, les trois ordinateurs, situés sur des segments réseau différents, communiquent entre eux. Le but est d’interdire au réseau « 10.1.1.0/24 » de communiquer avec le réseau « 10.1.2.0/24 » tout en ayant la possibilité de communiquer avec le réseau « 10.1.3.0/24 » pour ce faire, nous allons, sur l’interface Gi0/2, interdire les paquets provenant du réseau « 10.1.1.0/24 »

Pour parvenir aux résultats souhaités, nous appliquons trois étapes :

1 - Après s’être connecté sur le routeur en mode de configuration globale en tapant les commandes "enable" puis "configuration terminal", on commence par la création de la règle :

Router (config)#access-list 1 deny 10.1.2.0 0.0.0.255

En précisant "access-list 1" on attribue un ID à notre ACL, puis ensuite on précise que l'on veut refuser avec "deny", et enfin on précise l'adresse IP de destination (10.1.2.0) et le masque au format inversé appelé wildcards mask (0.0.0.225).

C’est-à-dire pour la règle « deny 10.1.2.0 0.0.0.255 » le routeur va refuser tous les paquets où les trois premiers octets de l'adresse IP source correspondent à ce masque générique. D'autre valeurs sont possible pour le wildcard mask, permettant d'affiner le filtrage en fonction des plages IP ou des regroupements de plages. Ainsi, un wildcard mask en 0.0.255.255 correspondra à un masque en /16; un wildcard mask en 0.0.63.255 correspondra à un masque en /18. Pour calculer la valeur, rien de plus simple, il suffit de soustraire la valeur du masque de sous-réseau à 255.

Wildcards mask ? De quoi parle-t-on ? Une petite explication s’impose. En effet, souvent, l’objectif de mettre en œuvre l’ACL c’est de faire correspondre une plage d’adresse IP (ou un sous-réseau complet) à l'ACL plutôt qu’une seule adresse IP. Cela est faisable à l’aide ce que l’on appelle Wildcards mask, ou en français masque générique (ou inversé). Notez qu’il ne s’agit pas d’un masque de sous-réseau. Par le biais de cette notation, nous avons la possibilité d’ignorer des parties de l’adresse IP, comme si elles correspondaient déjà.

Cela signifie que :

* Quand on met 0 : le routeur doit comparer cet octet comme d’habitude. La présence d'un "0" "figera" la partie correspondante de l'adresse IP
* Lorsqu'on met 255 : le routeur ignore cet octet, considérant qu’il correspond déjà.

**Deuxièmement,** comme évoqué précédemment, il faut autoriser explicitement les réseaux que l’on veut laisser passer, dans notre exemple c’est « 10.1.3.0/24 ». À défaut, il n’y aura pas de correspondance entre l’IP source contenue dans l’en-tête du paquet et les règles ACL, ce qui veut dire que le routeur va les refuser implicitement et notre réseau en 10.3.0/24 ne pourra pas non plus communiquer avec le 10.1.2.0/24.

Router(config)#access-list 1 permit 10.1.3.0 0.0.0.255

Notez qu'ici, on pourrait utiliser le mot clé "any" à la place de la plage réseau et du wildcard mask. Ce mot clé , qui veut dire "n'importe qui" peut être utilisé lorsque la règle s'adresse à tous les réseaux et hôtes. Comme on à déjà interdit en premier lieu notre premier réseau, on peut très bien estimer que le reste est permis.

Troisièmement, on sélection l’interface concernée par la règle :

Router(config)#interface gigabitEthernet 0/2

Enfin, on applique la règle en sortie :

Router(config-if)# ip access-group 1 out

Le "1" étant le numéro de l’ACL créée à la première étape, et "out" signifie que la règle s’applique sur les paquets sortants de l’interface.

**Les ACL étendues**

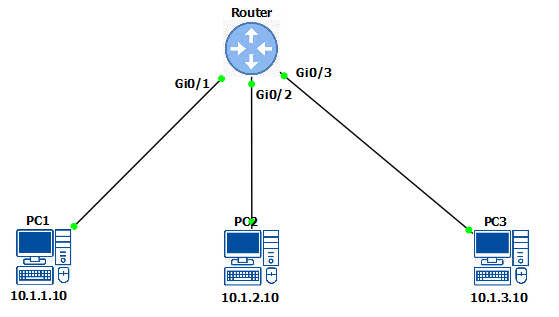
Les ACL étendues présentent plusieurs similitudes par rapport aux ACL Standards décrites dans la section précédente. Tout comme une ACL standard, **on active les ACL étendues sur les interfaces pour les paquets entrants ou sortants**, puis le routeur cherche dans la liste de manière séquentielle.

Les ACL étendues utilisent également la logique de première correspondance, car dès que la première instruction est mise en correspondance, le routeur arrête la recherche dans la liste d'ACL, en effectuant l’action définie.

En comparaison des ACL standards, **les ACLs étendues vont permettre d'analyser une plus grande variété de champs au sein de l'en-tête d'un paquet**. Cela rend les ACL étendues plus puissantes, plus précises, mais aussi un peu plus complexes.

Les ACL étendues suivent la même logique que les ACL standards, elles sont identifiables par un numéro, allant de **100 à 199** et de **200 à 2699**.

Un exemple sera plus parlant, nous allons créer une règle qui aura pour but d’**interdire le Ping de PC2 vers le PC3, tout en l'autorisant vers le PC1,** en posant les règles sur les sous-réseaux (tous en /24). Mettons ça en place en reprenant la même topologie :



Voici la configuration à appliquer sur notre routeur Cisco :

Router(config)# access-list 100 deny icmp 10.1.3.0 0.0.0.255 10.1.2.0 0.0.0.255

Router(config)# access-list 100 permit icmp 10.1.1.0 0.0.0.255 10.1.2.0 0.0.0.255

Router(config)#interface gig 0/2

Router(config-if)#ip access-group 100 in

Dans l'exemple ci-dessus, "icmp" correspond au protocole, en l'occurrence ici c'est un moyen de bloquer le ping. Ensuite, "10.1.3.0" c'est l'adresse IP d'origine suivie de son masque générique, et "10.1.2.0" l'adresse IP de destination suivie aussi de son masque générique.

Il est à noter que les ACLs étendues peuvent aussi examiner des parties d'en-têtes TCP ou UDP, en particulier les champs qui contiennent le numéro de port source et port de destination. Les numéros de port identifient le service qui envoie ou reçoit les données. Quand le mot "tcp" ou "udp" est inclus dans la règle de l'ACL, cela permet de préciser le port source et le port de destination afin d'avoir une règle plus précise.

Voici un exemple :

Router(config)# access-list 100 permit tcp 10.1.1.0 0.0.0.255 10.1.2.0 0.0.0.255 eq 21

Concernant « eq » il s’agit d’un opérateur qui veut dire égal, et le numéro de port 21 utilisé par le serveur FTP. On autorise le réseau 10.1.1.0/24 à communiquer sur le port 21 (pour le FTP) avec le réseau 10.1.2.0/24.

**Named IP Access Lists**

Comme vous le savez, toutes les listes d’accès doivent être identifiées par un nom ou un numéro. Ce type de liste d’accès est plus pratique, car on peut spécifier un nom significatif qui est facile à retenir et associer à une règle.

Les ACL nommées peuvent correspondre aux mêmes champs qu’une ACL standard et étendue, cependant, elles présentent trois grandes différences par rapport aux listes de contrôles d’accès numérotées, voyons cela ensemble :

L’utilisation de noms au lieu de chiffres pour identifier la liste ACL facilite la mémorisation et l'identification de l'ACL.

Utilisation de sous-commandes ACL, et non de commandes globales, pour définir l'action et les paramètres correspondants.

Utilisation des fonctionnalités d'édition de l'ACL qui permettent de supprimer des lignes individuelles de l'ACL et d'insérer de nouvelles instructions à une liste d'accès nommée.

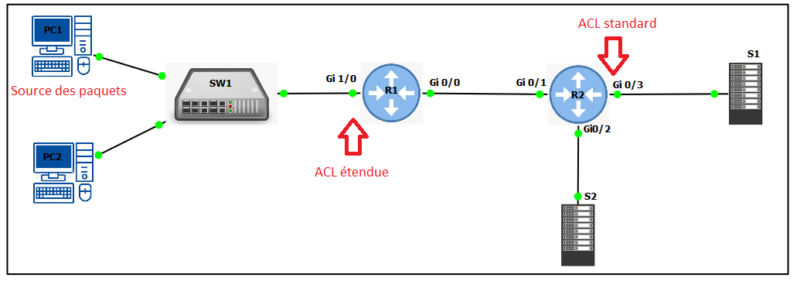
**Placement des ACL**

Bon, nous savons maintenant qu'il y a deux types d'ACL : standards et étendues. On sait aussi qu'il y a deux possibilités pour les placer : en entrée ou en sortie d'une interface. Mais alors, où placer tel type d'ACL? Sur quel routeur? Quelle interface?

Dans tous les cas, il existe une règle simple et immuable : une ACL standard se placera toujours au plus proche de la destination, une ACL étendue se placera toujours au plus proche de la source.

Pourquoi? Tout simplement, car une ACL standard, comme vous avez pu le constater, est très restrictive (blocage de tout le trafic d'un réseau ou d'un hôte), un placement au plus proche de la source risque de limiter fortement les possibilités de celles-ci. En revanche, l'ACL étendue filtrant au niveau de la couche 4 (ports TCP/UDP), son placement au plus proche de la source permet d'éviter de faire transiter des paquets qu'on ne souhaite pas voir sur le réseau, ainsi, on économise du temps de calcul.

Un petit exemple, en reprenant le schéma de la partie II :



Admettons que le PC1 soit la source, à savoir que c'est à partir de lui que je vais déterminer mes règles. Je veux interdire à ce PC1 d'accéder au réseau du serveur S1, je vais donc placer l'ACL standard sur l'interface Gi 0/3 de R2 en sortie . En revanche, si je veux interdire à PC1 d'accéder en SSH au serveur S1, je vais placer mon ACL étendue sur l'interface Gi 1/0 de R1 en entrée.

**Conclusion**

Nous avons vu ensemble comment les ACLs peuvent être utiles pour contrôler le réseau afin de filtrer certains flux. Les ACL permettent un filtrage au niveau du réseau et peuvent être complétées avec le filtrage applicatif (Proxy) qui va venir au niveau de la couche Application du modèle OSI. Le filtrage de paquet via ACLs opère niveaux 3 et 4 du modèle OSI, ce qui permet de faire déjà beaucoup de choses, mais peu ne pas suffire dans certains cas.