RESEAUX -HLIN611

Anne-Elisabeth Baert - baert@lirmm.fr

2017-2018

Table des matières

1 Chapitre 5 – Configuration de Réseaux et Adressage **2**
- 1.1 Le Besoin . . . 2
- 1.2 Retour sur l'Adressage . . . 2
- 1.3 Notion de masque . . . 4
- 1.4 Problèmes du routage . . . 4
- 1.5 Généralisations . . . 7

2 Chapitre 6 – Grande Traversée des Paquets **8**
- 2.1 Encore un problème de couches ? . . . 8
- 2.2 Recherche de l'Adresse Physique . . . 9

3 Chapitre 7 – Gestion d'Erreurs **11**
- 3.1 Erreurs Liées au Routage . . . 12
- 3.2 Utilisation Détournée . . . 14
- 3.3 Compléments Datagramme IP . . . 15

4 Chapitre 8 – Routage **16**
- 4.1 Introduction au Routage . . . 16
- 4.2 Algorithmes à Vecteurs de Distances . . . 18
- 4.3 Algorithmes à états de liens . . . 20
- 4.4 Autres algorithmes . . . 21

5 Chapitre 9 Grandes Applications – Serveurs de Noms **22**

1 Chapitre 5 – Configuration de Réseaux et Adressage

1.1 Le Besoin

Problème
On peut imaginer un réseau de Classe C sans répartition en sous-réseaux, sans trop de difficultés. Quoique, si l'on en a besoin, serait-ce possible à réaliser ?

Par contre, il est absurde de construire un réseau de classe B ou (pire) A sans le répartir en sous-réseaux.

Comment faire ?
On devrait pouvoir plutôt adapter l'organisation du réseau aux services demandés.

Organisation

Partager un réseau en sous-réseaux permet :

de faire correspondre l'organisation du réseau avec l'organisation administrative en services :

— les personnes d'un même service S_0 ont besoin de correspondre entre eux plus souvent qu'avec d'autres services (est-ce vrai ?) ;

— ils ont alors besoin de ***leur*** sous-réseau ;

— bien sûr, ceci ne doit pas empêcher le communications entre différents services, donc entre les sous-réseaux.

Organisation

Partager un réseau en sous-réseaux permet :

d'améliorer le fonctionnement global du réseau :

— lorsque tous les hôtes d'un réseau sont sur une seule liaison physique, alors toute comunication entre deux hôtes bloque la ressource réseau globale (pas de parallélisme possible) ;

— la séparation en sous-réseaux permettra de n'affecter qu'un sous-réseau lorsque deux hôtes d'un même sous-réseau communiquent entre eux ; le parallélisme devient possible : deux hôtes H_1 et H_2 peuvent communiquer sur leur sous-réseau SR_1 sans perturber la communication entre H_3 et H_4 sur SR_2.

1.2 Retour sur l'Adressage

Principe de l'Adresse Réseau

Une adresse réseau :

partie réseau	partie hôte

La partie *hôte* est à disposition de l'administrateur local. Qui peut en profiter pour créer des sous-réseaux.

Une adresse réseau et son sous réseau :

réseau	sous-réseau	hôte

La longueur attribuée à la partie *sous-réseau* va déterminer le nombre de sous-réseaux possibles et par conséquent le nombre d'hôtes dans ce sous-réseau.

Exemple de Partage

Un exemple sur 2 bits :
Deux bits de sous-réseaux permettent de configurer au plus 4 sous-réseaux, avec 64 hôtes au plus par sous-réseau.

Des adresses réservées
Il est d'usage de réserver deux adresses d'hôte :

- celle désignant *le réseau* (l' adresse hôte entière à 0 binaire) ,
- celle désignant *tous* (l'adresse hôte entière à 1 binaire).

Exemple sur une adresse

Un exemple sur $192.36.125.0$
On a $192.36.125.0$ attribuée à une institution. Si l'administrateur eut en faire 4 sous-réseaux, on aura la répartition suivante en binaire :

réseau	sous-réseau	hôte
11000000 00100100 01111101	00	000000 à 111111
11000000 00100100 01111101	01	000000 à 111111
11000000 00100100 01111101	10	000000 à 111111
11000000 00100100 01111101	11	000000 à 111111

Un exemple sur $192.36.125.0$
On a $192.36.125.0$ attribuée à une institution. Si l'administrateur eut en faire 4 sous-réseaux, on aura la répartition suivante en décimale :

SR n^o	adresse réseau	adresse tous	adresses hôtes
1	192.36.125.0	192.36.125.63	192.36.125.1 à 192.36.125.62
2	192.36.125.64	192.36.125.127	192.36.125.65 à 192.36.125.126
3	192.36.125.128	192.36.125.191	192.36.125.129 à 192.36.125.190
4	192.36.125.192	192.36.125.255	192.36.125.193 à 192.36.125.254

VOTAR

- En affectant 2 bits aux sous-réseaux, Quelle répartition de sous réseaux peut on faire ?
  - on pourrait aussi construire 1 sous-réseau de 128 adresses d'hôtes et 2 sous-réseaux de 64.

1.3 Notion de masque

Definition 1. Un masque est ...

.

Rapidité et efficacité !
Cette opération est nettement plus rapide qu'une suite de décalages.

Exemple de Masque

Examples 2. On prend un réseau de classe C, sans sous-réseaux, par exemple 192.34.38.0. Le masque 255.255.255.0 permet d'extraire l'adresse réseau à partir de l'adresse de tout hôte. Soit un hôte H d'adresse 192.34.38.212 ;

	192	34	38	212
et	255	255	255	0
s'écrit	11000000	00100010	00100110	11010100
et	11111111	11111111	11111111	00000000
résultat	11000000	00100010	00100110	00000000
soit	192	34	38	0

Attention

Des 0 ***et des*** 1
Un masque n'est pas nécessairement constitué d'une suite consécutive de 1, suivie d'une liste de 0.

En fait, dans la configuration des réseaux il est très commode d'utiliser des masques constitués d'une suite de 1 suivie d'une suite de 0, parce que les parties réseaux et sous-réseaux sont « à gauche ».

1.4 Problèmes du routage

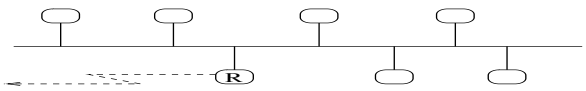
Pourquoi le Routage a besoin de Masques ?

Algorithmes de routage
On utilise des masques dans l'algorithme de routage (cf. couche réseau) pour répondre lors du traitement d'un paquet à la question :

Est-ce que le destinataire du paquet est sur le même (sous-)réseau que moi-même ?

On verra qu'en fait la question est un peu différente, mais elle se généralise facilement.

Examples 3. Soit un réseau de classe C, 192.34.38.0 sans sous-réseaux, connecté au monde extérieur par un routeur R. Une représentation dans le cas d'un réseau à diffusion (par exemple, ethernet) serait :



La table de routage classique, simplifiée, d'un hôte quelconque H_0 se présente ainsi :

Destination	Contact	Interface
192.34.38.0	direct	eth0
autre (*défaut*)	192.34.38.1	eth0

Les adresses et périphériques
`eth0` désigne ...
192.34.38.1 est l'adresse réseau du routeur.

Késako ?
Cette table dit que :

— pour tout paquet destiné à un hôte local, H_1 par exemple, il faut expédier le paquet directement à H_1 ; ceci veut dire que la couche liaison de H_0 mettra dans l'adresse de destination l'adresse liaison (dite aussi adresse physique) de H_1 ;

— pour tout paquet destiné à un hôte **non** local, H_{ext}, il faut expédier le paquet à 192.34.38.1, ici le routeur ; ceci veut dire que la couche liaison de H_0 mettra dans l'adresse de destination l'adresse liaison du routeur.

Des questions ....

Question :
Comment peut-on savoir qu'une adresse de destination fait partie du réseau local ou non ?

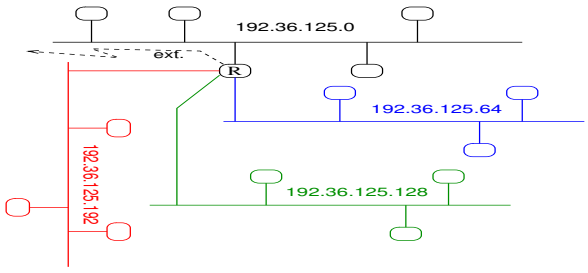
Réponse :
...
Si le résultat est identique, alors les deux hôtes sont sur le même réseau.

Question :
Quel masque faut-il appliquer pour que le routage se passe correctement dans tous les cas, quelle que soit la répartition en sous-réseaux ?

Spoil
Prochain cours ...

Masques de Sous-Réseaux

Example 4. Soit le réseau 192.36.125 divisé en quatre sous-réseaux, SR_1, SR_2, SR_3, SR_4 interconnectés par un routeur R.



Masques de sous réseaux

Le Problème ...
Un routage correct doit permettre à tout hôte d'acheminer directement un paquet destiné au même sous-réseau et de passer par le routeur pour toute autre adresse, extérieure ou appartenant à un des autre sous-réseaux. Le routeur doit pouvoir distinguer les divers sous-réseaux.

Une solution
On ajoute un masque pour ...

Masques de sous réseaux

Sur un hôte quelconque
Dans le sous-réseau 192.36.125.0

Destination	Contact	Masque	Interface
192.36.125.0	direct	255.255.255.192	eth0
autre (*défaut*)	192.36.125.1	???	eth0

Sur un hôte quelconque
Dans le sous-réseau 192.36.125.64

Destination	Contact	Masque	Interface
192.36.125.64	direct	255.255.255.192	eth0
autre (*défaut*)	192.36.125.65	???	eth0

Question
À quoi correspondent les adresses 192.36.125.1, 192.36.125.65 ?

Table du Routeur

table du routeur

Destination	Contact	Masque	Interface
192.36.125.0	direct	255.255.255.192	xxx0
192.36.125.64	direct	255.255.255.192	xxx1
192.36.125.128	direct	255.255.255.192	xxx2
192.36.125.192	direct	255.255.255.192	xxx3
autre (*défaut*)	$x.y.z.t$	???	xxx4

Questions :

— À quoi corespond $x.y.z.t$?

— Que représentent les interfaces xxx1 à xxx4 ?

Notation

Limitation
On peut constater qu'une adresse IP est insuffisante pour déterminer la taille du réseau correspondant. Par exemple, $192.36.125.0$ ne dit pas s'il s'agit d'un réseau découpé ou non.

Definition 5. On associe aux adresses de réseau le masque correspondant, par la notation : *adresse/masque* où *masque* désigne la longueur de la chaîne de bits à 1.

Example 6. $192.36.125.0/26$ désigne le réseau d'adresse $192.36.125.0$ avec un masque contenant 26 bits à 1, c'est-à-dire le masque $255.255.255.192$.

Toutes les valeurs de masque sont possibles, de $/1$ à $/32$.

1.5 Généralisations

Réseaux de Taille Intermédiaire

Le problème :
Que doit faire une organisation ayant besoin d'un réseau de plus de 254 hôtes, tout en ne justifiant pas d'un réseau de classe B ?

Ce problème est d'autant plus important que la classe B est saturée et qu'il y a actuellement peu de chances d'obtenir une telle adresse.

Solution :
Se faire attribuer plusieurs réseaux de classe C et jouer sur les masques et le routage afin de rendre cette attribution acceptable.

Sur-adressage

Definition 7. On vient de voir comment découper un réseau en sous-réseaux. Mais parfois on a besoin de faire l'opération réciproque : **associer plusieurs adresses obtenues en un seul réseau**. On parle alors de *sur-réseau*.

Des trous ...
Dans ce cas, il faudra obtenir des adresses *compatibles*, ...

Examples 8.
— $192.34.38.0$ et $192.34.39.0$ peuvent être associées avec un masque de ...
; on dit qu'elles sont *compatibles*.
— $192.34.38.0$ et $211.56.72.0$ ne sont pas compatibles : impossible de créer un réseau homogène avec ces deux adresses, avec un routage correct, sauf si on crée une table de routage avecautant de lignes que d'hôtes.
— $192.34.38.0$ et $192.34.37.0$ ne sont **pas** compatibles, à moins d'avoir obtenu **aussi** $192.34.36.0$ **et** $192.34.39.0$!

2 Chapitre 6 – Grande Traversée des Paquets

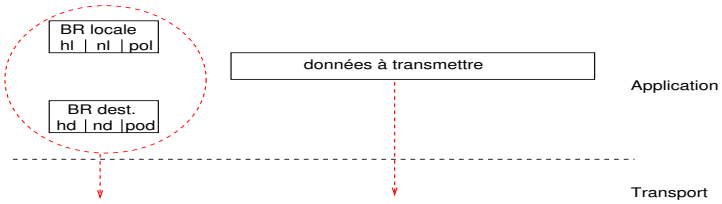
2.1 Encore un problème de couches ?

Rôle de l'application

Expédition
Lors d'une expédition, l'application expéditrice prépare et fournit à la couche en dessous (ici le transport) :

- le contenu du message (le *paquet vu par l'application*) à expédier
- les triplets des adresses des boîtes réseau *source* et *destination*.

Analyse dans l"application avant l'expédition (`send()` ou `sendto()`) : l'adresse de la BR de destination est déterminée.

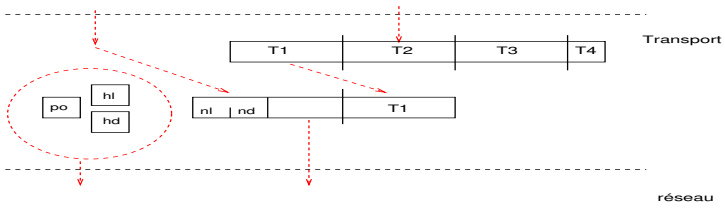


Rôle du Transport

Principe :
Chaque couche construit son paquet ; c'est ce qu'elle sait faire. Elle utilise ce qui lui est nécessaire et transmet à la suivante les éléments non utilisés jusque là.

La couche transport

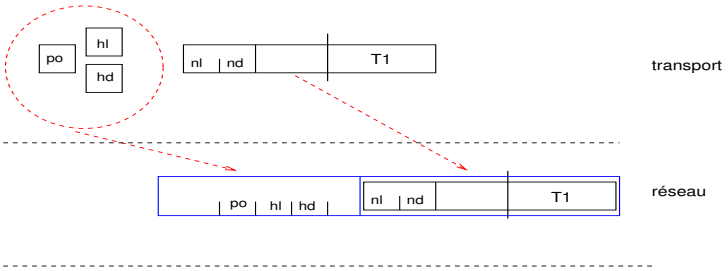
- utilise les numéros de BR inclus dans les adresses et seulement les numéros,
- découpe la données si nécessaire : déjà vu dans l'encapsulation.



Rôle de la Couche Réseau

La couche réseau

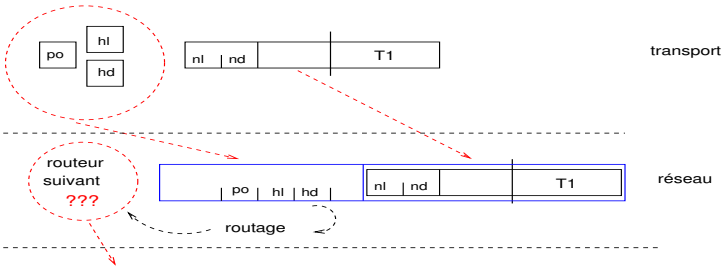
- utilise les adresses réseau (les numéros IP dans notre cas),
- redécoupe la donnée si nécessaire (penser aussi aux routeurs qui relient des réseaux de caractéristiques différentes)



Rôle du Routage

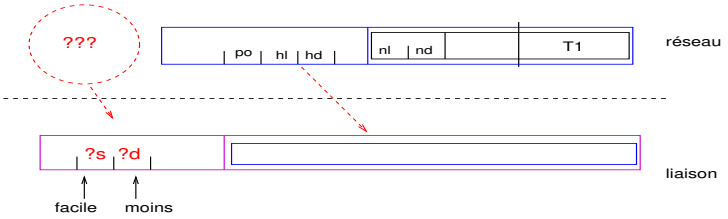
Le paquet ? :
Le paquet de *bout en bout* est constitué, mais à qui le faire suivre ?

— La couche réseau résoud le problème du routage ; elle trouve donc l'adresse **réseau** du destinataire suivant.



Rôle de la Couche Liaison

La couche Liaison
Le **problème** : le voisin d'en dessous aura besoin de l'adresse du niveau liaison du destinataire local pour acheminer la donnée. Connaissant l'adresse réseau, comment obtenir l'adresse liaison ?



Solution
Si le problème ci-dessus est résolu, la couche liaison pourra utiliser son propre protocole pour acheminer le paquet au destinataire suivant.

2.2 Recherche de l'Adresse Physique

Solution Statique

Correspondance d'adresse
Une solution possible consiste à avoir une table de correspondance pour l'ensemble des hôtes du réseau local, par exemple dans un réseau local type *ethernet* :

adresse réseau	*adresse physique*
201.202.203.1	$8:0A:B2:84:7F:04$
201.202.203.2	$0:12:34:8F:EE:AA$

Problèmes
Une telle solution résout le problème, mais présente tous les défauts d'une table statique dès qu'une mise à jour doit être effectuée : toutes les machines doivent être mises à jour de façon coordonnée.

Ces mises à jour peuvent devenir fréquentes dans le cas d'affectation d'adresses de réseau dynamiquement (voir *dhcp*).

Solution Dynamique

Definition 9. La solution proposée actuellement est de construire la table précédente dynamiquement. Le protocole ***ARP*** (Address Resolution Protocol) est utilisé pour cette construction.

Principes

- Diffuser à tout le réseau local l'adresse réseau du destinataire (local) en demandant à celui qui possède cette adresse de répondre en donnant son adresse physique.
- Chaque hôte va maintenir sa propre table de correspondance dite table *ARP*, comme dans l'exemple précédent.
- Une durée de vie sera associée aux données, permettant de ne pas ignorer un hôte dont une des adresses a été modifiée. On parle de ... .

Paquets ARP

Format des paquets ARP :

entête	type opération	adresse φ expéditeur
adresse réseau expéditeur	adresse φ cible	adresse réseau cible

type opération deux types sont possibles, requête (question) et réponse.

adresse φ adresse physique. Dans une requête ARP, l'adresse physique de la cible est évidemment absente.

Remarques :

- Ce même format de paquet peut être utilisé pour obtenir une adresse réseau à partir d'une adresse physique.
- La cible remplit le champ manquant, inverse expéditeur et cible, change le type de *requête* en *réponse* et renvoie le paquet.

3 Chapitre 7 – Gestion d'Erreurs

%frame

Table des matières

Présentation du Problème

Constat : l'acheminemqent de datagrammes dans l'Internet se fait **au mieux**, sans garantie de livraison.

Action : Si un routeur ne peut acheminer un datagramme alors il tente d'en avertir l'hôte expéditeur.

ICMP (*Internet Control Message Protocol*)est le protocole d'annonce d'erreurs.
Il est utilisé par le logiciel de la couche réseau (IP), non seulement dans le sens *routeur* $\rightarrow$ *hôte*, mais aussi par des hôtes ou routeurs pour des utilisations *détournées* comme par exemple des tests d'accessibilité.

Remarque : noter qu'un routeur ne peut annoncer l'erreur qu'à l'hôte source (seule adresse figurant dans le paquet IP). C'est le logiciel de la couche réseau sur l'hôte source qui traite l'erreur ou la fait suivre à l'application correspondante.

Types d'Erreurs

Les exemples suivants permettent de voir l'étendue des dégats et de constater qu'annoncer une erreur à la source n'est pas toujours la bonne solution.
Un routeur peut se trouver dans une situation désagréable comme :

— pas de chemin vers l'adresse destination dans sa table de routage,

— l'hôte de destination n'existe pas (détection par le dernier routeur),

— le réseau par lequel il veut acheminer est en panne ou congestionné,

— obligation de détruire le datagramme, par exemple, suite à une erreur du code de contrôle, ou à une durée de vie dépassée.

ICMP intègre aussi la possibilité d'obtenir diverses informations entre routeurs, entre hôtes ou les deux. Une des utilisations les plus connues est

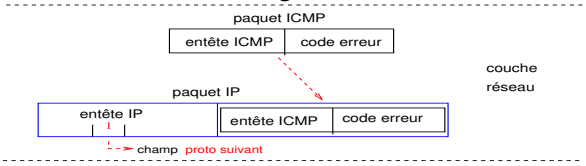
- la demande d'écho et
- la réponse associée à cette demande,

par le logiciel `ping`.

Où Traiter ?

ICMP fait partie de IP. C'est-à-dire que dans la couche *réseau* il y a du logiciel et des paquets ICMP au même titre que IP.

Les paquets ICMP sont acheminés dans des datagrammes IP. On en déduit l'encapsulation suivante :



Noter que le champ *protocole suivant* dans l'entête IP est utilisé pour désigner le *suivant*, soit dans la couche transport (tcp, udp, autre), soit ICMP, avec une valeur différente bien sûr.

Noter aussi qu'une erreur dans l'adresse source du datagramme va aboutir à la perte de l'annonce d'erreur.

Rappel du Paquet IP

Un rappel de la forme d'un paquet IP

octet 1	octet 2	octet 3	octet4
Vers. | lg. ent.	type service	lg. paquet	
Identification		drapeaux	place frag.
durée vie	proto. suiv.	contrôle entête	
adresse IP source			
adresse IP destination			
options . . .			
. . .			bourrage
Données . . .			

Paquet ICMP

•L'entête de tout paquet ICMP est de la forme :

type	code	contrôle
8bits	8bits	16 bits

•Le champ *type* désigne le type d'erreur. **Exemples** :

8	demande d'écho
0	réponse écho

3	destination inaccessible
4	congestion
11	dépassement durée de vie

•Le champ *code* comporte une information complétant le type d'erreur. **Exemples** :

0	réseau inaccessible

1	hôte inaccessible
6	réseau inconnu

•Dans tous les cas d'erreur, ICMP ajoute dans la donnée les 64 premiers bits du datagramme ayant provoqué l'erreur. Plus généralement, la donnée permet de compléter plus explicitement les indications de l'entête.

3.1 Erreurs Liées au Routage

Table des matières

Destination Inaccessible

• Lorsqu'un routeur ne peut pas délivrer ou faire suivre un datagramme, il construit un message d'erreur ICMP, avec dans le champ *type* la valeur 3, dans le champ *code* une valeur de 0 à 12, calcule la somme de contrôle et ajoute au paquet ICMP les 64 premiers bits du datagramme, extrait l'adresse de l'hôte source Hs puis détruit ce datagramme non routable.

Ce paquet est encapsulé dans un datagramme IP, contenant en source le routeur expéditeur et en destinataire Hs, avec dans le champ *protocole suivant* le code 1, désignant ICMP.

L'hôte source peut ainsi analyser *plus sérieusement* la cause du rejet et faire suivre à l'application un retour d'erreur.

•Noter qu'un routeur peut faire suivre des datagrammes **sans se rendre compte** que la destination est inaccessible.

Exercice : Donner deux exemples démontrant ce phénomène, l'un concernant un hôte destinataire (penser à ethernet par exemple pour répondre), l'autre concernant un routeur destinataire.

Dépassement de Durée de Vie

Associer une durée de vie au datagramme IP permet de faire en sorte qu'un datagramme ne puisse circuler indéfiniment dans l'Internet sans arriver à destination.

Est-ce possible ? Oui, pour des erreurs de routage provoquant des aller-retours d'un datagramme entre deux routeurs, chacun ayant malheureusement une interprétation erronnée des informations de routage, ou pire, une boucle de routage entre plusieurs routeurs (voir le chapitre sur le routage).

Solution : le champ *durée de vie* contient dans sa forme la plus simple (l'actuelle, dans IPV4), le nombre maximum de routeurs que le datagramme peut traverser. Chaque datagramme IP se voit appliquer le principe suivant :

Algorithme TTL

Appelons *TTL* le champ *durée de vie* du datagramme IP.

L'hôte source du datagramme initialise ce champ à une valeur déterminée, dans le logiciel de la couche réseau.

Chaque routeur applique ensuite l'algorithme suivant :

```
TTL - - ;
si (TTL == 0) alors
    expédier message ICMP (dépassement TTL) à hôte source
    détruire datagramme ;
```

Les Échos

La demande d'écho dans ICMP permet aux routeurs de savoir si les routeurs voisins sont actifs ou non. Lorsqu'un routeur reçoit un message ICMP de *demande d'écho*, il doit répondre par un message ICMP de *réponse écho*.

Cette caractéristique est utilisée non seulement entre routeurs, mais aussi entre hôtes pour tester leurs présences, comme nous l'avons déjà vu pour le logiciel `ping`.

Noter que ping visualise la valeur du champ *Durée de Vie* et affiche aussi le temps d'aller-retour du datagramme.

Exercice : Pour quelles raisons est-ce que la durée d'aller-retour du premier datagramme dans `ping` est souvent supérieure aux suivants ?

Et si ICMP Provoquait une Erreur ?

Remarque Importante : Tout paquet ICMP est encapsulé puis routé dans un datagramme IP. Dès lors, ce datagramme peut subir les mêmes avatars que tout datagramme IP, perte, congestion, abandon.

Les pertes et erreurs engendrent des pertes et des erreurs (d'après Rez O.)...

Dans leur sagesse, les concepteurs ont décidé qu'on ne devait construire un message ICMP relatif à un datagramme contenant déjà un message ICMP...

Conséquence : voici encore une raison pour laquelle des protocoles comme `TCP` doivent inclure des garanties, ajouter des délais, tenir actifs les circuits virtuels, et alerter les applications avec des moyens complémentaires.

3.2 Utilisation Détournée

Table des matières

Détournenemnt de TTL

Le comportement des routeurs relativement au champ *Durée de Vie*, permet d'en faire une utilisation détournée, afin de déterminer le chemin d'accès à un hôte.

La commande `traceroute` applique un algorithme dont le principe est :

Algorithme 1 : traceroute(Hdest)

```
HdestNonAtteint = vrai ;
TTL=0 ;
tant que (HdestNonAtteint) faire
    TTL + + ;
    expédier (datagramme, Hdest) ;//demande écho par exemple ;
    si (réponse ICMP) alors  afficher (expéditeur erreur ICMP) ;
    ;
    sinon si (réponse de Hdest) alors  HdesNonAtteint=Faux ;
    ;
    ;
```

Question : Est-ce vraiment un chemin correct ?

Analyse de Traceroute

Exercice : prendre le schéma de réseau suivant et montrer que l'algorithme précédent peut afficher des chemins faux ou pire, inexistants. On suppose que S cherche un chemin vers D et que les Rx représentent des routeurs.

On peut définir

— *faux* par : le résultat donné ne sera pas un chemin suivi par un paquet,

— *inexistant* par : le chemin affiché contient au moins un arc (ou un sommet) inexistant.

3.3 Compléments Datagramme IP

Fragmentation

Un datagramme IP peut être *fragmenté*, c'est-à-dire découpé en morceaux, sur un ou même plusieurs routeurs, en fonction des caratéristiques des réseaux que le routeur interconnecte.

Chaque fragment circule comme un datagramme indépendant, donc peut suivre un chemin différent d'un autre fragment.

Conséquences :

— le réassemblage ne peut se faire que sur le hôte destinataire final,

— dans la couche IP qui doit attendre la réception de tous les fragments, tout en acceptant entre temps d'autres datagrammes,

— chaque fragment doit contenir les informations nécessaires à l'identification du datagramme d'origine et à l'insertion correcte du fragment dans ce datagramme.

l'Avenir de la Fragmentation

Noter que dans IPV6, cette notion de fragmentation a été abandonnée ! C'est aux hôtes et aux protocoles de plus haut niveau de *se débrouiller* pour que le datagramme chemine correctement sans découpage.

Autrement dit, on simplifie le routage, en se déchargeant des problèmes embêtants sur les voisins.

C'est aux voisins de chercher un chemin acceptable ; s'il y a un problème entraînant le non acheminement pour cause de longueur excessive, on recevra un message d'erreur. Il faudra chercher un autre chemin.

4 Chapitre 8 – Routage

4.1 Introduction au Routage

Le Problème du Routage

Déterminer en fonction de l'adresse réseau du destinataire final d'un datagramme le prochain destinataire.

On peut compléter très légèrement le tableau déjà vu dans le cas d'un hôte quelconque sur un réseau local.

Destination	Contact	Masque	Interface
201.202.203.0	direct	255.255.255.192	eth0
autre	201.202.203.1	0.0.0.0	eth0

Contact indique soit le prochain routeur, soit une destination sur le même réseau.

Problèmes

— Cette table peut prendre des dimensions gigantesques dans le cas d'un routeur censé connaître l'ensemble des destinations de l'Internet ou d'un sous-ensemble.

— Comment construire cette table ?

Routage Statique

Le routage statique constitue une solution simple à la construction de la table : elle est figée et modifiée uniquement par une intervention d'un administrateur.

Cette solution est parfaitement bien adaptée à un réseau local avec un seul routeur assurant la connectivité vers le monde extérieur.

L'utilisation d'une route par défaut permet de passer rapidement le relai d'un hôte à un routeur, d'un routeur à un autre routeur. On comprend mieux pourquoi des incohérences sont possibles. Et il y a pire, par exemple des boucles...

Routage Dynamique

Dans le cas d'un routeur reliant plusieurs réseaux, un routage adaptatif ou dynamique permettra de tenir compte de :

— l'infrastructure des réseaux connectés,

— l'arrivée et la réparation de pannes, la création de nouveaux liens,

— la charge des réseaux (congestions, oscillations),

— de la qualité de service requise, etc.

Une distribution *intelligente* des adresses de réseaux permettrait de réduire la taille des tables de routage (voir routage hiérarchique). Hélas, ce n'est pas le cas, du moins ceci n'a pas été fait systématiquement, dans l'Internet.

Connaissance Partielle et Erreurs

Constat : Dans tous les cas, le résultat de l'algorithme de routage est l'adresse du *suivant*.

On espère que les routeurs sont cohérents entre eux, c'est-à-dire que le *suivant* peut continuer à acheminer correctement le paquet. Sinon, on aura des **erreurs de routage**.

Ceci reste vrai même si un routeur connaît le chemin complet, car on ne peut pas **forcer** une décision sur un **autre** routeur ; sauf cas spécifiques de tests de chemins, ce serait néfaste de le forcer.

Traitement des Erreurs - Un Début

On peut empêcher un paquet de vivre indéfiniment dans l'Internet :

Principe :

— Un champ *durée de vie (ttl)* est attaché à chaque paquet (cf. entête du paquet IP) ; il est initialisée par l'hôte source du paquet ;

— Chaque routeur décrémente la durée de vie ;

— Le routeur qui arrive à une durée de vie nulle ou négative détruit le paquet.

Actuellement, la durée de vie est mesurée en *nombre de routeurs traversés*, dit aussi *nombre de sauts*.

---

ttl- - ;
si *(ttl* $>$ *0)* **alors** router paquet ;
;
sinon expédier (erreur routage) à hôte source ;
;

Classes d'Algorithmes

Plusieurs classes d'algorithmes dynamiques existent en fonction de l'étendue des réseaux reliés.

Globalement, on a besoin de trouver des chemins dans un graphe dynamique. Il faut se poser les questions de

— l'efficacités des algorithmes distribués ou centralisés ;

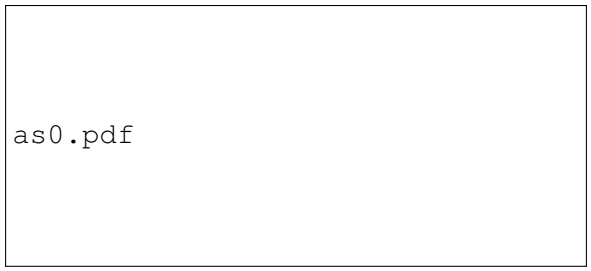
— convergence, stabilité et cohérence des algorithmes.

Pourquoi plusieurs classes d'algorithmes ? Parce qu'on ne peut plus se contenter d'une organisation centralisée.

Organisation en Systèmes Autonomes de l'Internet

Aujourd'hui, l'Internet est organisé en *Systèmes Autonomes* (AS ci-après), vastes réseaux (en général), administrés chacun par une entité unique.

Chaque AS possède des routeurs *intérieurs* reliant les sous-réseaux entre eux, et des routeurs *extérieurs* reliés à des routeurs extérieurs d'autres AS.



Organisation du Routage dans l'Internet

Chaque AS organise son propre routage interne librement. Plusieurs algorithmes sont connus : RIP, OSPF.

Pour la partie externe, il faut un protocole commun.

Les AS communiquent entre eux par un seul algorithme lié à un seul protocole : aujourd'hui, BGP.

Les routeurs *extérieurs* d'un AS doivent connaître **toutes** les adresses des autres AS afin de constituer un routage cohérent. On insiste sur **toutes**, pas seulement celles des AS et routeurs adjacents.

On pourra voir qu'il est difficile de concilier le fonctionnement extérieur, visible, avec les choix politiques internes. C'est un sujet de recherches actuellement.

4.2 Algorithmes à Vecteurs de Distances

Algorithme à Vecteur de Distance RIP

Principes :

- Diffusion d'informations sur le routage à base de la **distance** ; quelle métrique pour exprimer une distance ? le plus fréquent : *nombre de sauts* ;
- Chaque routeur dispose d'une table contenant des triplets $(destination, numéro\_de\_liaison, coût)$ c'est-à-dire pour telle destination, envoyer sur telle liaison pour tel coût ;
- La table est mise à jour dynamiquement ; il y a diffusion périodique d'informations $(destination, coût)$
- Chaque routeur qui reçoit une information la compare au contenu courant de sa table ;
- Si l'information est *meilleure* il la prend ;
- Sinon, il y a des cas où l'on est obligé d'accepter une information fut-elle *moins bonne*, d'autres où on la rejettera.

Algorithme RIP

Données : table de routage ; des doublets $(dest, cout)$ reçus sur une liaison l_{recue}
Résultat : table de routage
pour *toutes les données arrivant* **faire**
  si $dest_{recue}$ *trouvée dans table* **alors**
    si $l_{recue} == l_{table}$ **alors**
      $c_{table} = c_{recue} + 1$;
    sinon
      si $c_{table} > (c_{recue} + 1)$ **alors**
        l_{table} = l_{recue} ;
        c_{table} = c_{recue} + 1 ;
  sinon
    ajouter $(d_{recue}, l_{recue}, c_{recue} + 1)$ dans table

Vecteurs de Distances - Exemple

vecDist1.pdf

Extrait de *Le routage dans l'Internet.*

Initialisation ;
A diffuse $(A, 0)$ sur les liaisons 1 et 3 ;
B diffuse $(A, 1), (B, 0)$ sur 1, 2 et 4 ;
information perdue sur 4 ;
C diffuse $(C, 0), (A, 2), (B, 1)$ sur 2 et 5 ;
D diffuse $(D, 0), (A, 1)$ sur 3 et 6.

A			B			C			D			E		
→	l	c	→	l	c	→	l	c	→	l	c	→	l	c
A	loc	0	B	loc	0	C	loc	0	D	loc	0	E	loc	0
			A	1	1				A	3	1			
B	1	1				A	2	2						
						B	2	1						
			C	2	1							C	5	1
												A	5	3
												A	6	2

Justifications

Traitement global : Chaque routeur diffuse périodiquement sa table, ensemble de couples $(destination, coût)$ vers ses voisins adjacents.

Important : Si une nouvelle information sur une destination arrive par la **même** liaison que celle par laquelle on route, alors il faut la prendre, qu'elle soit bonne ou mauvaise.

Pourquoi ? Considérer le point de vue d'un routeur.

vecDist2.pdf

Si A annonce à R qu'une destination X est atteinte pour un coût 10 et que le même, A annonce ensuite un coût différent (envisager 7 puis 12) pour la même destination, R doit accepter ce nouveau coût, sauf si entre temps il a un meilleur chemin.

Remarques

Attention : Nous avons vu **une** simulation possible de la diffusion. L'ordre de diffusion peut être totalement différent, avec des résultats différents sur les tables.

On peut démontrer la convergence de cet algorithme si aucune modification (panne, apparition de nouvelles liaisons) n'arrive. Mais les pannes et modifications sont **fréquentes**. On peut montrer que pour toute panne ou apparition de liaison, l'algorithme converge si un nouvel incident n'a pas lieu avant l'aboutissement de la convergence.

Le défaut de fonctionnement reproché à RIP est résumé ainsi :
Les bonnes nouvelles se propagent vite, les mauvaises se propagent doucement.

Traitement d'une Panne

Principe du traitement d'une panne : annoncer un coût ∞ pour chaque voisin en panne. Cette détection peut se faire par un outil comme `ping` permettant de tester l'existence d'un hôte.

Exemple : Supposons que la liaison 1 tombe en panne. B corrige sa table avec le triplet $(A, 1, \infty)$ et la diffuse.

Selon l'ordre de propagation de l'information, on peut constater une convergence rapide, ou des phénomènes connus sous le nom de *comtage à l'infini* :
Supposons que C diffuse sa table, **avant** que B ne diffuse la sienne. C diffuse entre autres informations $(A, 2)$, qui lors du traitement sur B va engendrer le triplet $(A, 2, 3)$! ! !

D'autres défauts de fonctionnement de cet algorithme ont été répertoriés (voir bibliographie), accompagnés de solutions plus ou moins heureuses. Elles font l'objet de cours ultérieurs.

4.3 Algorithmes à états de liens

Algorithmes à États de Liens - Principes

- Chaque routeur possède la topologie complète du réseau ;
- Deux tâches sont accomplies pour arriver à connaître cette topologie :
  - test d'activité de tous les routeurs adjacents : échanges courts (type `ping`)
  - diffusion périodique de l'état des liens : c'est un compte-rendu des communications possibles. L'état est rediffusée *autant que nécessaire* à tous les routeurs participants, avec horodatage (numéro de séquence) des messages.
- Chaque routeur calcule un plus court chemin vers chacun des autres routeurs.

Exemple : *Open Shortest Path First* (OSPF), est actuellement un algorithme à état de liaisons très utilisé à l'intérieur des systèmes autonomes de l'Internet.

Algorithmes à états de liens - Algo

Données : ensembles $(voisins, couts, séquence)$; N : ensemble de nœuds dont le pcc est connu ;
$D(v)$: coût parcours vers v
Résultat : nouvelle topologie et arbre des plus courts chemins
//Initialisation N={A} ; D(v) infini (tous);
pour *tous les nœuds v* **faire**
    si *v adjacent* **alors**
        D(v)=coût(A,v);
//mise à jour
répéter
    trouver w extérieur à N tel que D(w) min;
    ajouter w à N;
    pour *tout v adjacent à w* **faire**
        D(v)=min(D(v),D(w)+c(W,v));
jusqu'à *tous les nœuds v explorés*;

4.4 Autres algorithmes

Autres algorithmes

Ces quelques lignes juste pour dire qu'il existe d'autres types d'algorithmes et d'autres raffinements. Par exemple un algorithme à *état de chemins* (BGP), est utilisé entre systèmes autonomes.

Pour d'autres développements voir la bibliographie et la suite de ce cours.

5 Chapitre 9 Grandes Applications – Serveurs de Noms

Serveurs de Noms - Petit Retour

L'application *Serveurs de Noms* est aujourd'hui une base fondamentale de l'Internet.

Elle n'est pas seulement utilisée pour la correspondance nom↔adresse des hôtes, mais plus généralement, pour enregistrer des informations d'administration.

RFC1034 décrit les concepts de base. Suivent un tas de compléments et mises à jour.

Exemple : La messagerie électronique utilise les serveurs de noms pour trouver l'hôte à contacter pour l'acheminement des courriels sur un site.

En effet, que l'adresse électronique d'une personne comporte ou non un nom d'hôte, il y a souvent un (ou quelques) hôte(s) fixé(s) pour l'acheminement dans le domaine de destination.

L'application *serveur de noms* ou DNS, permet de déterminer en fonction d'un nom d'hôte ou du nom de domaine de l'adresse électronique, le serveur à contacter pour la messagerie.

Principes de Fonctionnement

Déjà vu :

— Les conventions de nommage.

— Le fonctionnement de base, consistant à avoir un serveur de noms dans le domaine local, contenant la correspondance noms↔adresse des hôtes locaux.

— La requête est acheminée vers un autre serveur (souvent un serveur racine), qui est capable de la refaire suivre.

— Le serveur qui connaît la correspondance répond directement au demandeur.

La partie du système d'exploitation prenant en charge la résolution s'appelle *resolver*. Cet outil existe forcément sur chaque hôte, consulte un fichier de base ($/etc/resolv.conf$), contenant au moins :

— le domaine dans lequel il faut chercher les noms simples (ceux donnés sans le caractère point) ;

— l'adresse IP d'au moins un serveur de noms, en général le serveur de noms local.

Exemple

```
search info.rmatique.fr nameserver 123.231.111.12
```

•Veut dire que tout nom simple, par exemple *hotel* sera recherché (complété) en tant que *hotel.info.rmatique.fr*. Ensuite, ayant le nom complet, la requête sera expédiée vers 123.231.111.12.

•Le serveur de noms reçoit alors un nom interne ou externe et réagit en fonction de ce qu'il trouve dans sa base.

•S'il ne trouve rien, il doit posséder l'adresse d'au moins un autre serveur de noms (souvent un serveur racine) afin de faire suivre la requête.

•Noter que les requêtes aux serveurs de noms utilisent udp.

•Noter aussi qu'il est indispensable d'avoir l'adresse IP du serveur de noms !

Remarques

— Il est possible de gérer des sous-domaines séparémment, c'est-à-dire en fait avoir dans un même domaine, géré par une même autorité administrative, plusieurs serveurs de noms relatifs à plusieurs domaines. Plus généralement, on parlera de *zone d'autorité* pour l'unité gérée par un serveur donné. Mais, un serveur de noms peut gérer plusieurs zones...

— Il est possible aussi d'avoir plusieurs serveurs pour une même zone : penser aux pannes en particulier. Dans ce cas, il y a un serveur dit *primaire* disposant de l'information **origine** et des serveurs *secondaires* disposant d'une **copie**.

— Les requêtes sont vulnérables et DNSSEC décrit dans la RFC2535 permet de construire une *chaine de confiance*.

— La mise à jour dynamique (RFC2136) devient indispensable avec la distribution dynamique des adresses avec des logiciels comme DHCP par exemple.

Plus loin

•Chaque serveur gère des informations comme la correspondance *inverse* (obtention d'un nom à partir d'une adresse), le contact messagerie, la durée de vie des informations, les noms multiples d'un même hôte, etc.

•L'application *serveur* s'appelle *named*, qui lit un fichier de configuration de démarrage (*named.conf*).

Plusieurs bases contiennent ensuite toutes les autres informations. La localisation de ces bases est donnée dans le fichier de démarrage.

De même, les serveurs secondaires ou primaires sont nommés dans ce fichier de démarrage.

Conclusion : le point d'entrée pour aller plus loin est : *named* en plus de DNS.