Le protocole UDP

(Z:\Polys\Internet_transmission_donnees\03-UDP.fm- 9 décembre 2005 12:10)

PLAN

- Présentation
- Format des paquets UDP
- Multiplexage
- Détection des erreurs
- Conclusion

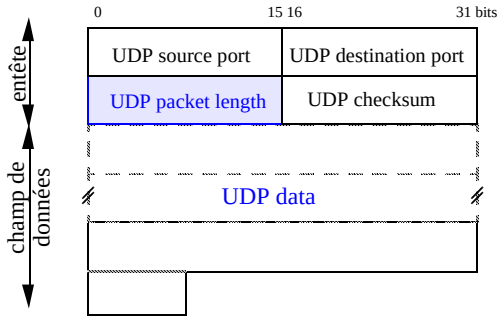
1. Présentation

"User Datagram protocol"
. RFC 768
. août 1980

Transmission :
. Par paquet de taille variable
. En mode non connecté (sans contexte !)
. Simple, sans ajout de mécanisme de contrôle
=> "datagram"
. Se contente des services offerts par la couche inférieure (IP)
=> peu de traitement
=> peu de délai

Multiplexage (sur-adressage) :
. une adresse IP <-> une station
. un port UDP ou TCP <-> un processus

2. Format des paquets UDP



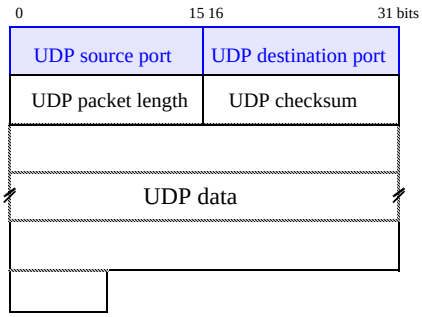
Format général :
. une entête de taille fixe.
. un champ de données de taille variable.

UDP packet length :
. longueur totale du paquet UDP (header+data)
. 8 <=packet length < 64 K octets.

Overhead minimum :
. + 8 octets !
. X25.3=3 octets, TP4(sans options)=5 octets!

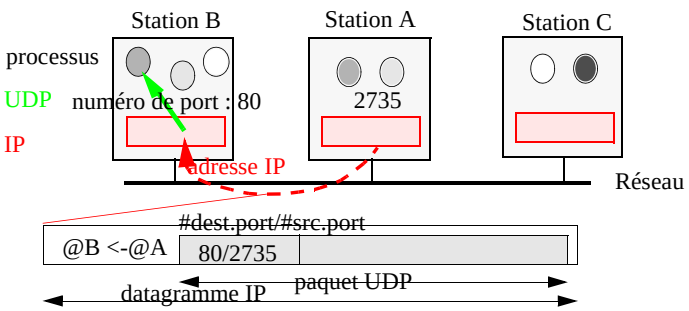
3. Multiplexage

3.1. Rôle



. accès à une station du réseau (par une des ses connexions)
=> adresse IP
. accès à un processus dans une des stations du réseau
=> numéro de port (sur-adressage)

UDP source port est optionnel :
. il spécifie le nº de port utilisé lors de la réponse
. 0 => il est inutilisé



3.2. Les numéros de port

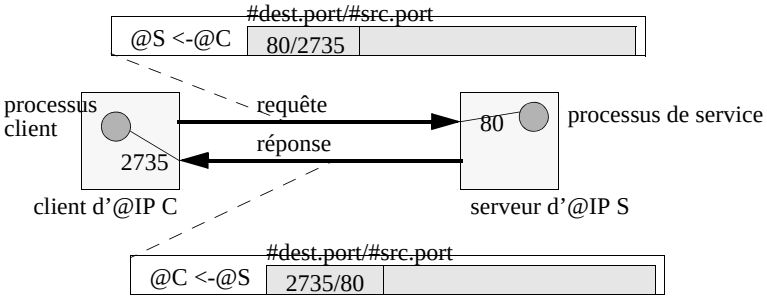
Indépendance vis-à-vis des applications :
. de leur durée de vie
. de la dénomination
. de leur spécificité

Numéro de port réservé (n°< 1024) :
. définit des services (/etc/services)
par exemple : 7 : echo
20/21 : ftp
111 : SUNRPC (remote procedure call)
513 : whod (who deamon)
520 : RIP (routing information protocol)
15 : netstat
80 : www
119 : nntp
. permanent
. fonctionnel

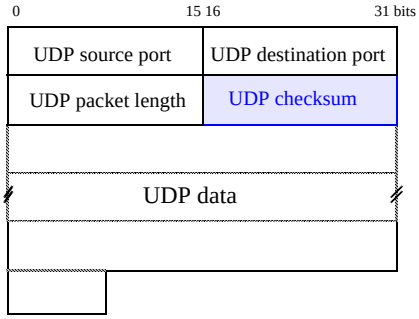
Attribution dynamique des autres numéros de ports :
. souplesse
. optimisation de l'utilisation de la ressource

Utilisation des numéros de port au sein du paradigme client/serveur :

- le processus serveur attend sur un port bien connu de tous
- le processus client envoie un message (une requête) sur ce port
- le processus serveur traite la requête
- et, si besoin, répond au client en utilisant le numéro de port indiqué dans la requête



4. Détection des erreurs



Détection de la corruption du contenu du paquet UDP par un champ de contrôle d'erreur (checksum).

. A l'émetteur : calcul sur le message transmis et transmission dans le champ "UDP checksum" du résultat.

. Au récepteur : calcul sur le message reçu et comparaison avec le contenu du champ "UDP checksum". S'ils sont identiques, aucune erreur n'est détectée, sinon le paquet est corrompu donc détruit (ignoré !).

. Complément à 1 de la somme en complément à 1 par demi-mots (processeur 16 bits !)

. Sur la totalité du paquet UDP (en supposant que le champ "UDP checksum" est nul)

+ le pseudo-entête IP suivant :

Source IP address		
Destination IP address		
0	Protocol	Total length

⇒ Ne respecte pas les couches !

Optionnel

. Pas de contrôle d'erreur : le champ "UDP checksum" = 0,

⇒ le vrai 0 est codé '1111111111111111_2'.

La même technique exactement est utilisée par TCP.

5. Conclusion

Protocole simple de transmission de données :
- surcôut minimal pour les paquets UDP.
- surcôut minimal pour le traitement du protocole :
  . pas de contexte,
  . très peu de contrôle :
    détection d'erreur optionnelle.

Sans (avec très peu d') augmentation de service :
- le service fourni est le service disponible.
- multiplexage (n° port).

Adapté au multicast

Attention : non-fiable =/= sans augmentation de la fiabilité !