Réseau

Couches :

PDU = Protocol Data Unit

1. Physique PDU = Bit
2. Liaison de données
   1. MAC : medium Access Control
   2. LLC : Logical Link control (Ethernet ne l'utilise pas, Wifi si)

   PDU = trame
3. Réseau PDU = paquet
4. Transport Transferts de segments
5. Session 6. Présentation 7. Application

Liaison de données :

* Logiciel qui permet de récupérer des trames sur le réseau = sniffer
* Ethernet 802.3, Ethernet II, Wifi : si ≥ 05DC ⇒ II ⇒ type ; si < 05DC ⇒ 802.3 ⇒ Lg
* Analyse trame MAC :

dest						src							
00	60	97	51	96	9f	00	60	b0	3c	7a	87	08	00
@ MAC test						@ MAC src						type IP	
version IP													
45	00	00	32	0c	fc	00	00	39	06	5c	65		
lg									protocol				
c1	37	5f	01	01	96	33	01	0f	85	00	25	...	

[Ethernet] [IP] [TCP] — [IP] car 0800 ; [TCP] car 06 = protocol ; protocole

* Possibilités d'émission d'Ethernet : unicast, multicast, broadcast } groupes
* Equipement : Half Duplex : HUB ; Full Duplex : Switch

Réseau :

* mode connecté (IPv6)
* mode non connecté (IPv4) : chaque paquet est préfixé par une entête contenant une @ dest, suffisante pour permettre la livraison externe du paquet

IPv4 : Dest unique

@IP : 193.55.96.26 = Partie réseau + Hôte

@IP réseau = bits hôte à 0 ; @IP broadcast = bits hôte à 1

Classe A : 0 → 127 . x . x . x masque = 255.0.0.0

Classe B : 128 → 191 . x . x . x masque = 255.255.0.0

Classe C : 192 → 223 . x . x . x masque = 255.255.255.0

masque : donné avec @IP : @IP & masque = réseau

@IP privées : 8 (2^24) bits à 1

classe A : @réseau : 10.0.0.0 → 10.255.255.255 masque : /8

classe B : 172.16.0.0 → 172.31.0.0 /16

classe C : 192.168.0.0 → 192.168.255.0 /24

Calculer un sous-réseau : ex : 30 ss réseaux de 136.25.0.0

* 30 → 32 = 2^5 ⇒ 5 bits de ss-réseau

	@ réseau	ss-r	Hôte
	136.25.	0000	0 : 000 . 0000 0000
	136.25.	0001	0 : 000 . 0000 0000
	136.25.	0010	0 : 000 . 0000 0000
	136.25.	0011	0 : 000 . 0000 0000

etc...

@ réseau = 136.25.8.0

@ broadcast = 136.25.0000 1111.1.1 = 136.25.15.255

masque ss réseau = 255.255.1111 1000.0 = 255.255.248.0

IPv6 : @ sur 16 octets, en hexa, + sécurisé, + de machines

Unicast, multicast, anycast, broadcast (en plus)

Routage :

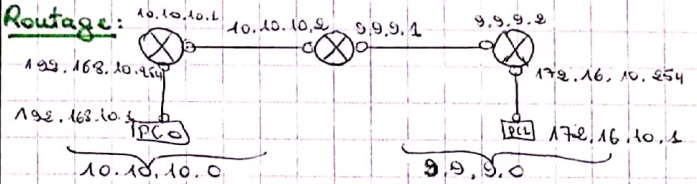


Table de routage : netstat -rn

ARP : Address Resolution Protocol : correspondance @MAC ↔ @IP

@IP = ville @MAC : quartier (prend du temps)

A —Broadcast Ethernet ARP qui contient IPB→ B ⇒ stockage dans la table ARP_A de IP_B ⇔ MAC_B

A ←répond et insère son @MAC— B

Routage → statique : renseigné : @réseau dest, masque réseau dest, @IP prochain routeur vers dest

Routage → dynamique : mise à jour dynamiquement des tables de routage (vecteurs distants / état de liens)

Transport :

TCP : Transmission control protocol, mode connecté, tente de détecter les erreurs et de les corriger, fiable

UDP : non connecté, non "fiable"

Sockets = point d'accès au service TCP/UDP, lien entre l'applicatif et la couche transport.

socket() : création ; bind() : attribution @ de point d'accès

3 infos pour identifier un processus : @IP, protocol, n° port

Timers : TCP peut envoyer tous segments d'un coup, timers pour temporisation : A chaque émission, création d'un timer, si ack avant fin timer ⇒ OK, sinon segment perdu.

Win = nb d'octets libres dans la file d'attente

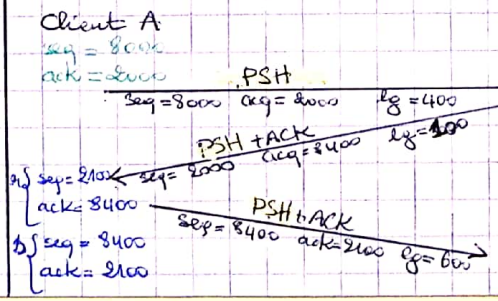
Segment TCP :

port src	port dest	n° seq	n° acq	taille entête	Réservé	Flags	...

Flags :

ACK = accusé de réception SYN = synchro FIN = demande fin de connexion

PSH = push RST = Reset URG = urgent.



Algo TCP :

si type = SYN ou SYN+ACK alors
  seq = seq + 1
sinon
  seq = seq + lg
fsi

} envoi

si type = SYN ou SYN+ACK alors
  ack = seq reçu + 1
Sinon
  ack = ack + lg
fsi

} réception

Application :

Applicatif : → rend accessible les informations mémorisées sur un système distant
→ rend accessible une ressource à d'un sgt distant

NFS : network file system

- partage de répertoires
- utilise le protocole RPC
- config :
  - → construit la liste des répertoires exportables
  - → vérifie que le RPC est lancé
  - → lance les serveurs
  - → vérifie l'exportation

  } serveur
  - → montage
  - → opérations sur les fichiers

  } client

NIS : network information service

- centralisation d'informations sur un réseau UNIX
- distribution des infos contenues dans des fichiers de config
- comporte serveur (maître + esclave), une bibliothèque d'accès client et des commandes d'administration
- serveur NIS gère des cartes (maps) stockées dans des fichiers de BDD à partir des fichiers de config. Le client récupère les infos en ? le serveur à partir d'appels RPC.

DHCP : Dynamic Host Configuration Protocol

- assure la config auto des @IP et masque d'une machine. Utilise couche transport UDP port 67/68
- → Machine envoi un broadcast "coucou je veux une @IP"
  - → tout serveur DHCP le reçoit, puis répond en unicast en lui proposant une @IP (ping 1 : émet à @IP proposée ; ping 2 : si pas de réponse @IP libre)
  - → Client donne accord en broadcast
  - → serveur acquitte l'accord en unicast

MSN → MSNP (Microsoft notification protocol)

ICMP : Internet Control Message Protocol.

- véhicule des messages de contrôle et d'erreur :

type 0 (Echo) ; type 3 (dest inaccessible) ; 5 (redirection) ; 8 (demande écho) = ping.

- Protocole transport msg mail (SMTP) : simple mail transfert Protocol
  - → ds TCP port 25
  - → serveur renvoi code 220 (suite ok/ko) + nom
  - → envoi requête → réponse code 250 (OK) + nom

POP = post office Protocol pop3 : port 110
- permet l'authentification
- permet de relever le courrier sur un serveur
- besoin de télécharger les mails ⇒ pas de manip sur le serveur

IMAP = internet message access protocol port 143
- permet de relever le courrier sur un serveur
- permet l'authentification si nécessaire de façon chiffrée
- gère les mails sur le serveur ⇒ pas besoin de les télécharger

SPAM = courrier non désiré

Format MIME.

Le web :

HTTP = HyperText Transfert Protocol
- rapatriement des documents / soumission de formulaire
- Fonctionnement au dessus d'une connexion TCP
- méthodes HTTP : get = récupérer doc, Post = envoi de données, Head = récup infos du doc, Put, delete, ...
- Code de réponse : 100→199 : informat° 200→299 : succès 300→399 : redirection 400→499 : erreur dû au clt 500→599 : erreur serveur
- caches HTTP : sur le navigateur / sur les proxys
- Implanter la notion de session sur plusieurs requêtes HTTP :
  → input HIDDEN dans les formulaires
  → ré-écriture dans chaque URL
  → utilisation d'un cookie
- HTTP secured = HTTP + SSL
  → utilisat° chiffrement asym puis sym
  → certificat X509, port 443
  → Failles : MAN in the Middle, sécu interne à l'entp
- Défaut Http : latence d'une page → amélioré° HTTP/2 protocol SPDY

Les sockets :

- mécanisme d'interface de programmation
- Connexion définie par : type protocol / @IP / n° port processus
- Fichier virtuel sous Unix avec les opérations d'ouverture, fermeture, écriture, lecture
- Types : stream socket : TCP, Datagram socket : UDP, Raw-Socket : IP, ICMP

En IPv6 :
- API identiques qu'en IPV4
- plusieurs clients : thread, nvx processus
  Recv bloquante, entrée clavier aussi
- SSL : secure socket layer entre TCP et applicative ⇔ authentification du serveur, confidentialité des données, intégrité

- Etapes SSL :
  → Handshake → SSL Record
  pré-requis : certificat + clé publique / privée

Sécurisation d'un réseau :

- Authentification, confidentialité, intégrité, disponibilité non-répudiation
  ⇒ TCP/IP ne respecte pas ces critères.
- Techniques de sécurisation : chiffrement, protocoles sécurisés, restreindre les accès, anti-virus, ...
- 2 types de sécu ⇔ d'un système isolé / d'un système & réseau
- Pare-feu : permet de protéger le réseau interne de l'extérieur ⇒ filtrage
- Sécurité d'un réseau :
  → équipement actif (routeur, pare-feu...)
  → NAT statique = associat° n @ avec n @.
  = remplace l'@ de la station du réseau par une @ externe publique
  → @ retour : proxy-arp (pare-feu et routeurs)
  → NAT dynamique : toutes stations & un réseau local peuvent envoyer des PDU IP vers Ethernet, différenciés
  → PAT = traduction automatique de l'@ IP de la station émettrice avec l'@ IP de la passerelle et translation du port (& fichier stations avec ñ @IP passerelle)
  → Sonde IPS / IDS
  - IPS = intrusion detect° system = bloquent flux réseau, doit être en coupure de flux
  - IDS = in. detect° system = alertent les admin, coupure de flux ou en écoute.
- sécurité & (réseau) : (virtual private network)
  → VPN, protocole L2TP
- proxy : composant logiciel servant d'intermédiaire entre la source et la destination.
- chiffrement : symétrique (DES, IDEA...) / asym (DH, RSA) / Hachage ou signature (MD5, SHA-256)
- sécurité Wifi : par défaut : WEP, WPA ou WPA2 : +++
- Protocole Kerberos : protocole d'authentification
  → récupère clé de session puis auth vers serveur et connexion

ping : → s'appuie sur un protocole ICMP
→ permet de connaître : @IP, n° séquence ICMP, durée de vie du paquet (TTL) ...
→ fail au début car on ne connaît pas @mac dest ⇒ d'où une requête ARP qui permet de faire la correspondance @mac ↔ @IP

DNS : Domain Name System : traduit les noms de domaine Internet en @ IP, permet à un PC de récupérer l'@ IP d'un serveur que le PC veut joindre

Hub : travaille sur la couche 1 du modèle OSI, envoi tous les messages qu'il reçoit sur le réseau.

Switch : chargé de diriger le trafic dans la bonne direction.