Cours de Réseaux

6 – La couche transport

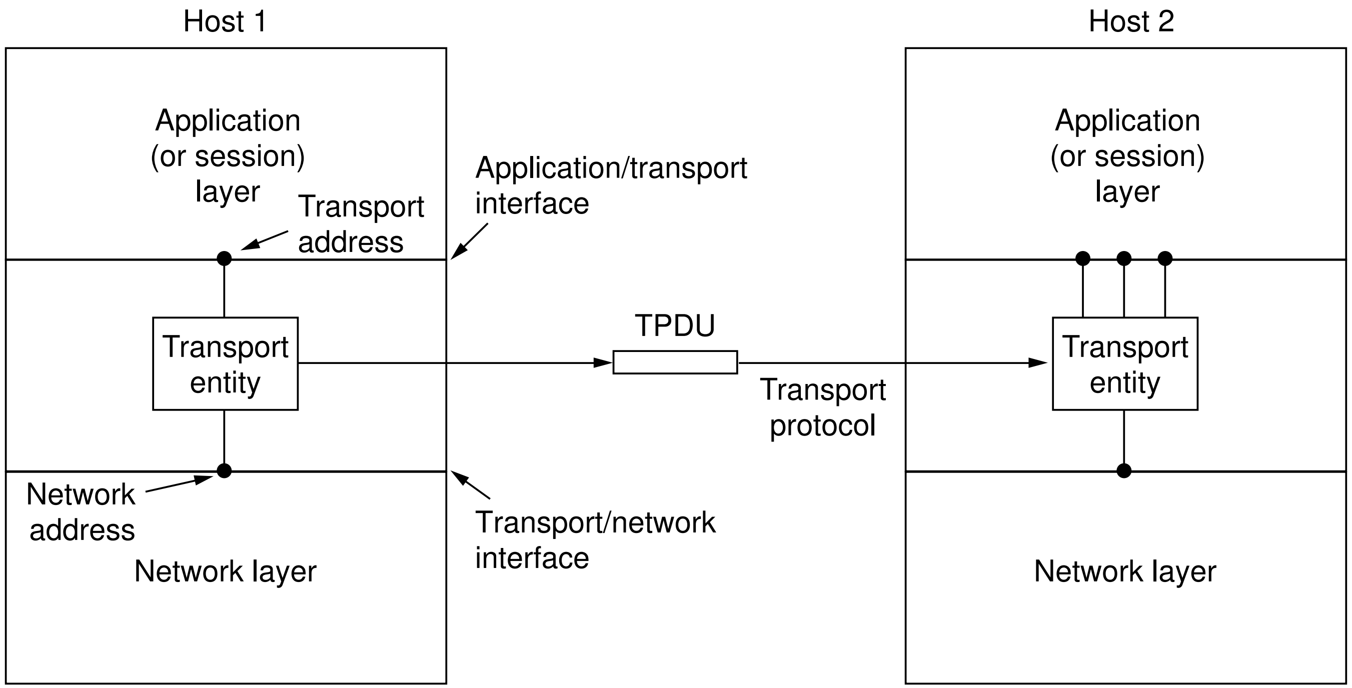
Rappel : le modèle hybride

5	Couche application
4	Couche transport
3	Couche réseau
2	Couche liaison de données
	(Sous-couche contrôle d'accès au canal)
1	Couche physique

La couche transport

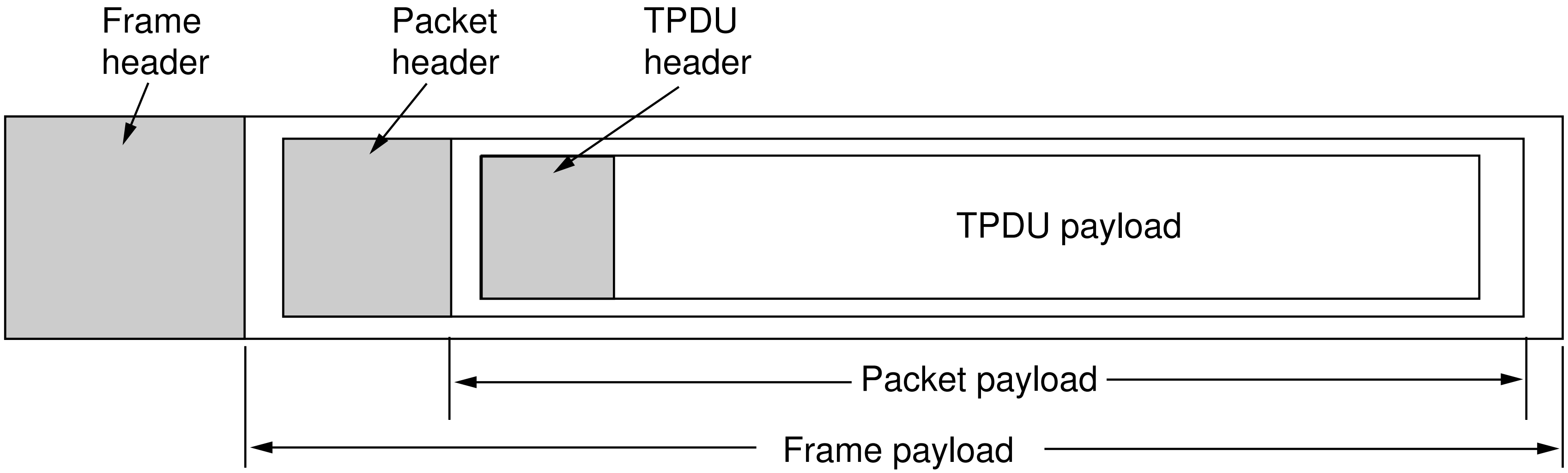
- Gère les communications de bout à bout entre processus.
- Abstraction des primitives de la couche réseau.
- Gestion des *ports*.
- Service : transmission fiable d'un flot, mais aussi autres services possibles.
- Contrôle d'erreurs et de congestion pour la connexion *bout-à-bout*.
- Tourne entièrement sur les machines utilisateurs, pas sur les routeurs (à quelques exceptions près).

Couches réseau, transport, et application



TPDU = *Transport Protocol Data Unit*

Imbrication de TPDU, paquets, et trames



Adressage dans la couche transport

Une adresse dans la couche transport est une paire de
- adresse IP
- numéro de *port* qui donne un point d'accès à la couche transport

Comment connaître le bon numéro de port ?
- Convention pour des services standard.
- Serveur d'annuaire (*directory server*) qui donne des numéros de port sur demande.
- Serveur de processus qui crée un nouveau processus sur demande et renvoie le numéro de port.

Les ports dans internet

Les numéros < 1024 : *port réservés*. Liste gérée par l'*IANA* : *Internet Assigned Numbes Authority*. Quelques exemples :

ftp	21
ssh (secure shell)	22
telnet (*insécure !*)	23
SMTP (email)	25
HTTP (web)	80
HTTPS (web sécurisé)	443

Les ports réservés sous UNIX

Extrait du fichier `/etc/services` sur une machine UNIX :

```
# Network services, Internet style
ftp          21/tcp
daytime      13/tcp
daytime      13/udp
www          80/tcp   http         # WorldWideWeb HTTP
domain       53/tcp   nameserver   # name-domain server
domain       53/udp   nameserver
```

Les services Internet

Les services suivent chacuns son *protocole d'application* (voir le chapitre suivant).

Par convention, *si* on veut fournir un certain service standard on l'attache au numéro de port selon `/etc/services`.

Le service peut être réalisé par un processus UNIX dédié, ou par le *superserveur* `inetd` (voir le fichier `/etc/inetd.conf`).

Problèmes particuliers dans la couche transport

- Le délai « normal » dépend de la topologie du réseaux, de l'émetteur et du destinataire.
  Dans la couche liaison de données, par contre, le délai maximal est normalement connu d'avance.
- Les TPDU peuvent être beaucoup retardés à cause des congestion et à cause du fait que les routeurs utilisent des files d'attente.
  Temps de vie maximal d'un paquet sur internet : 120 secondes (?)
  La couche réseau, par contre, ne peut pas « mémoriser » des trames.

Protocoles internet de la couche Transport : TCP et UDP

UDP (*User Datagram Protocol*) utilisé pour envoyer des petits blocs de données (service sans connexion).

Applications :

– DNS : domain name service

– RPC : remote procedure call

TCP (*Transmission Control Protocol*) : service fiable avec connexion.

Utilisé pour transfert de fichier, web, login à distance, email, etc.

Primitives de transport TCP (1)

`socket` créer un socket (point de connexion entre processus en UNIX) pour le point terminal d'une communication.

`bind` attacher une adresse (numéro de port) à un socket (UNIX : seul *root* peut attacher un numéro de port < 1024)

`listen` annoncer la volonté d'accepter des connexions

`accept` renvoie la tentative de connexion suivante (bloque jusqu'à l'arrivée d'une telle tentative)

`connect` tente d'établir une connexion (sur un socket existant)

`close` libère la connexion.

Primitives de transport TCP (2)

`send` envoyer des données via une connexion

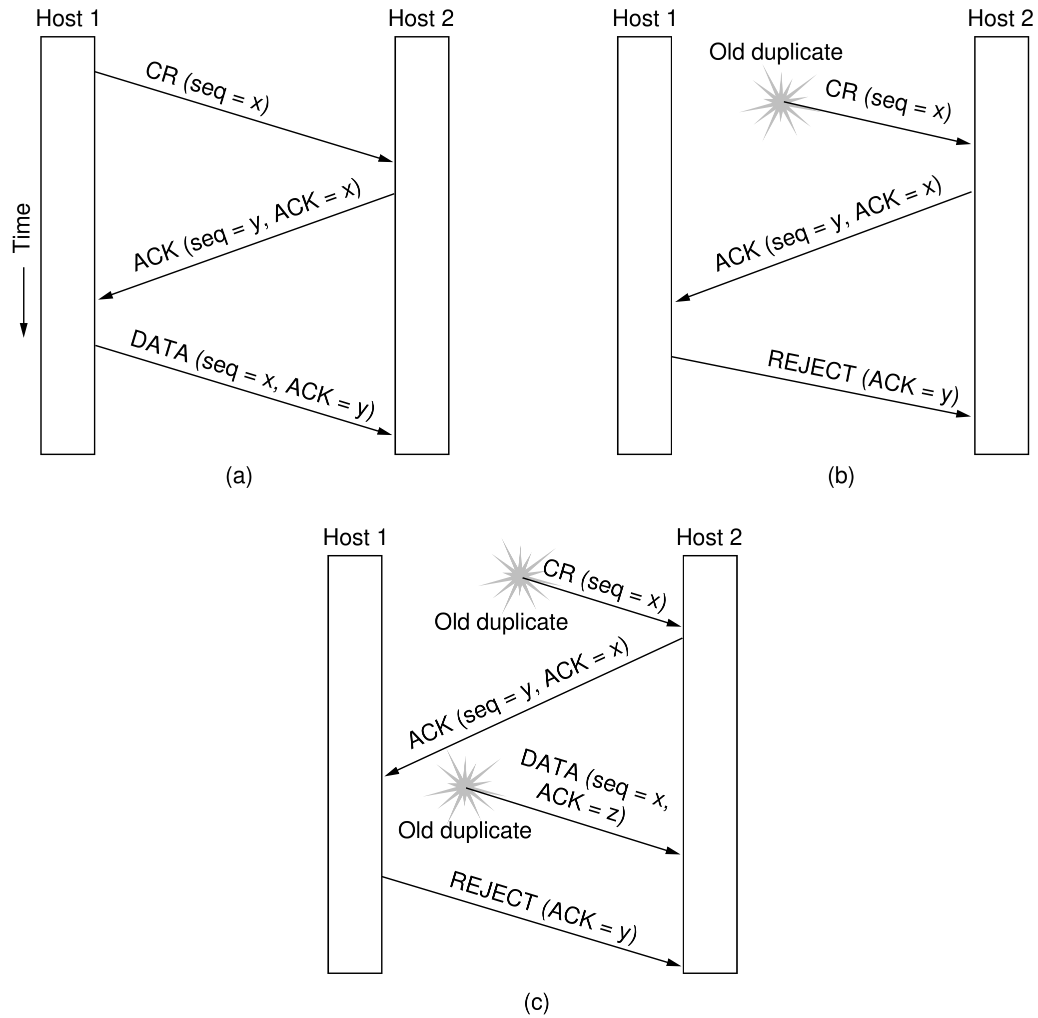
`receive` recevoir des données d'une connexion

Établissement d'une connexion TCP (1)

Procédure en trois étapes (*three-way handshake*).

– Chacun des deux hôtes maintient un compteur pour numéroter les connexions.
– L'émetteur demande une connexion (CR - *connection request*) et donne son numéro de connexion.
– Le récepteur envoie un acquittement avec le numéro reçu et son propre numéro de connexion)
– l'émetteur renvoie un acquittement.

Établissement d'une connexion TCP (2)



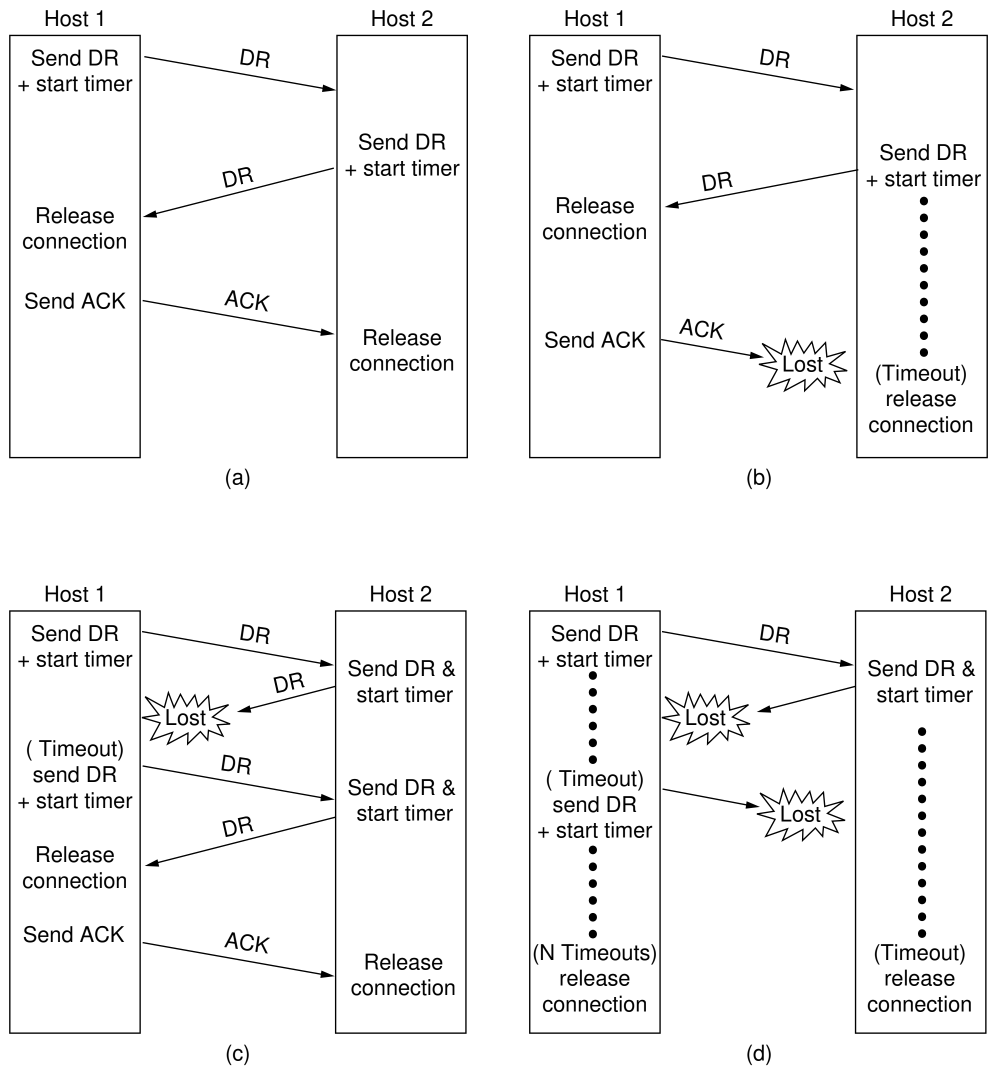
Déconnexion TCP (1)

On essaye d'établir un accord des deux côtés sur le fait que la connexion peut être libérée

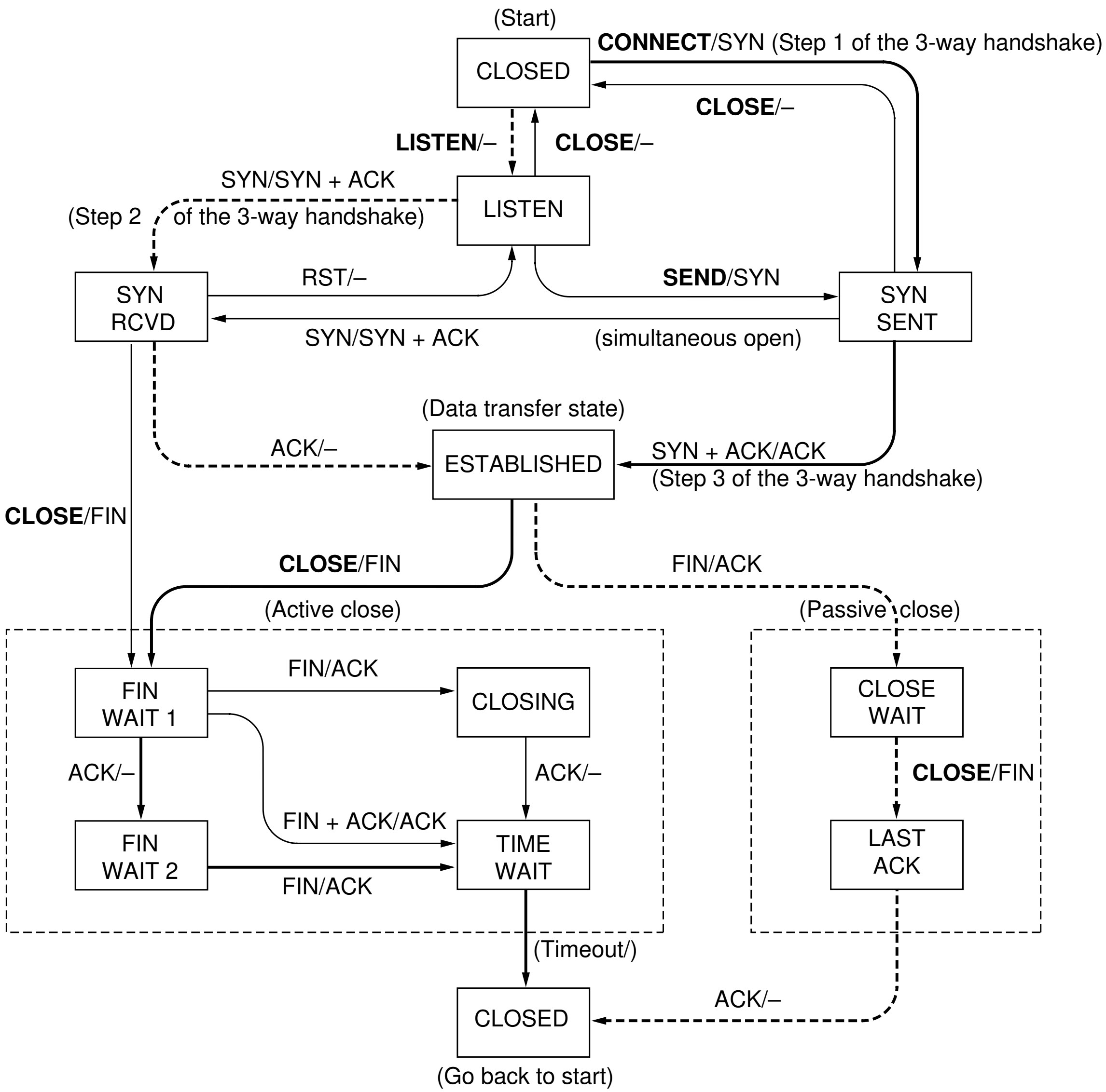
Il n'est pas possible que les deux hôtes soient sûr que leur dernier message a été reçu.

$\Rightarrow$ Protocole pratique utilise des horloges. Si pas d'acquittement reçu à temps on coupe la connexion. De plus, couper les connexions qui sont inactives pour un certain temps.

Déconnexion TCP (2)



Les états d'une connexion TCP



Les états des connexions sous UNIX

```
% netstat --inet
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Ad Foreign Address         State
tcp        0      0 []:46742 dellpe02.lsv.ens-ca:ssh TIME_WAIT
tcp     3253      0 []:39145 news-4.proxad.net:nntp  ESTABLISHED
```

Dans cet exemple : `[]` remplace `bdv75-4-82-227-67`.

Transmission de données en TCP

- Protocole avec *fenêtre d'anticipation* similaire aux protocoles vus dans la couche *liaison de données*.
- Gestion dynamique des tailles des fenêtres (aussi pour le contrôle de congestion)
- Une machine doît gérer plusieurs fenêtres (une par connexion TCP)

Firewalls (1)

(fr : *gardes-barrières* ou *mur pare-feu*) : filtrer les paquets selon

– adresse IP émetteur (attention : peut être fausse)

– port d'émission

– adresse IP destinataire

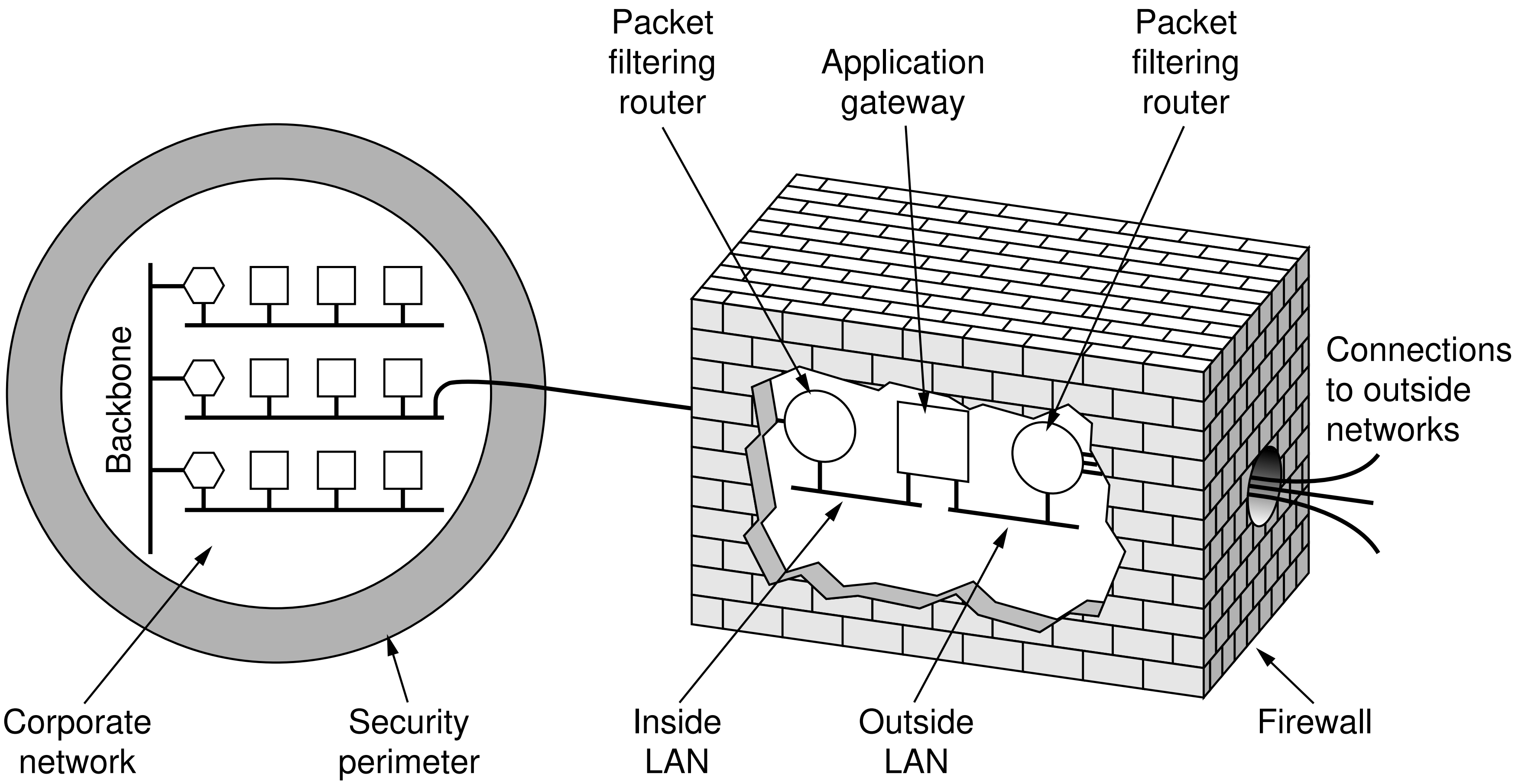
– port de destination

De préférence sur une machine dédiée.

Souvent en combinaison avec *network address translation* (NAT).

Des *firewalls* modernes peuvent aussi filtrer selon l'application.

Firewalls (2)


Packet
filtering
router
Application
gateway
Packet
filtering
router
Backbone
Connections
to outside
networks
Corporate
network
Security
perimeter
Inside
LAN
Outside
LAN
Firewall