Les couches Transport et Application

Communication d'un processus vers un autre processus

P. Laurençot

Rappel

- *Couche 3 (réseau)*
  - @IP -> permet d'identifier une machine unique au monde
  - @IP privée et publique
  - Classful et maintenant classless (CIDR)
  - **Objectif** : Trouver un chemin pour aller de la source à la destination en passant par plusieurs réseaux
    - Table de routage + ARP

- *Couche 2*
  - Permet de communiquer sur un même réseau
    - Utilisation @MAC
    - ARP

Plan du chapitre

Les couches hautes

1) Fonctionnalité

2) Les protocoles TCP/UDP
   - Généralité
   - Les ports

3) Les applicatifs ou la couche applicative

4) Sécurisation
   - VPN
   - Pare-feu (firewall)
   - IDS/IPS

Couche 4 - Fonctionnalité

Le rôle de la couche Transport est d'acheminer des octets entre **des processus** s'exécutant dans des systèmes appartenant à la même couche Réseau, sans qu'ils aient à se préoccuper des problèmes de médium, de routage, etc...

- Cahier des charges
  - transmission de données entre processus (de bout en bout)
  - données normales, données express
  - transparence
  - mode connecté, mode non connecté
  - Encapsulation des données de la couche supérieure
- Les couches Transport les plus utilisées
  - ISO
    - 1 Service en mode connecté, 5 classes de protocole
    - 1 Service en mode non connecté, 1 protocole
  - technologie TCP/IP
    - 1 Service en mode connecté (TCP), 1 protocole
    - 1 Service en mode non connecté (UDP), 1 protocole

Classe protocole transport

Objectif : fiabilité…

Classes (de qualité) de Service Réseau (niveau 3)	A	B	C
détecte les erreurs	oui	oui	non
taux d'erreurs détectées	fort	fort	nul
tente de corriger les erreurs détectées	oui	non	non
taux d'erreurs corrigées	fort	faible	nul
signale les erreurs non corrigées	oui	oui	non

	Transport ISO						technologie TCP/IP	
	mode connecté Classe :					mode non connecté	TCP	UDP
	0	1	2	3	4			
connexion	oui	oui	oui	oui	oui	non	oui	non
transfert de données	oui	oui	oui	oui	oui	oui	oui	oui
correction des erreurs signalées	non	oui	non	oui	oui	non	oui	non
contrôle de flux	non	non	oui	oui	oui	non	oui	non
données express	non	non	oui	oui	oui	non	oui	non
détection et correction des erreurs non signalées	non	non	non	non	oui	non	oui	non

TCP / UDP

TCP
Service en mode connecté

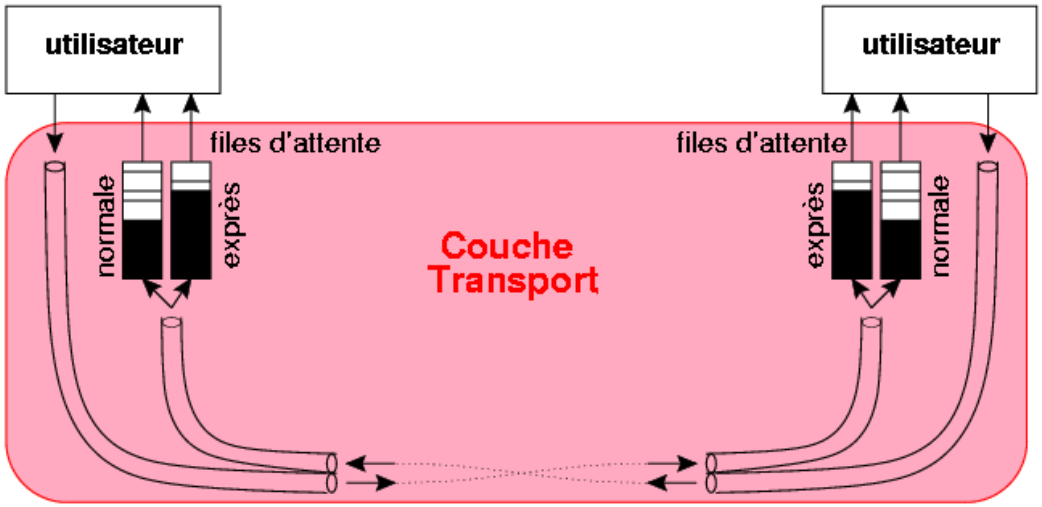
- jusqu'à 65534 points d'accès par système
- détection et tentative de correction des pertes et duplications
- signalement des erreurs
- livraison dans l'ordre d'émission
- contrôle de flux
- Service fiable

UDP
Service en mode non connecté

- jusqu'à 65534 points d'accès par système
- un ou plusieurs destinataires
- Service "non fiable"

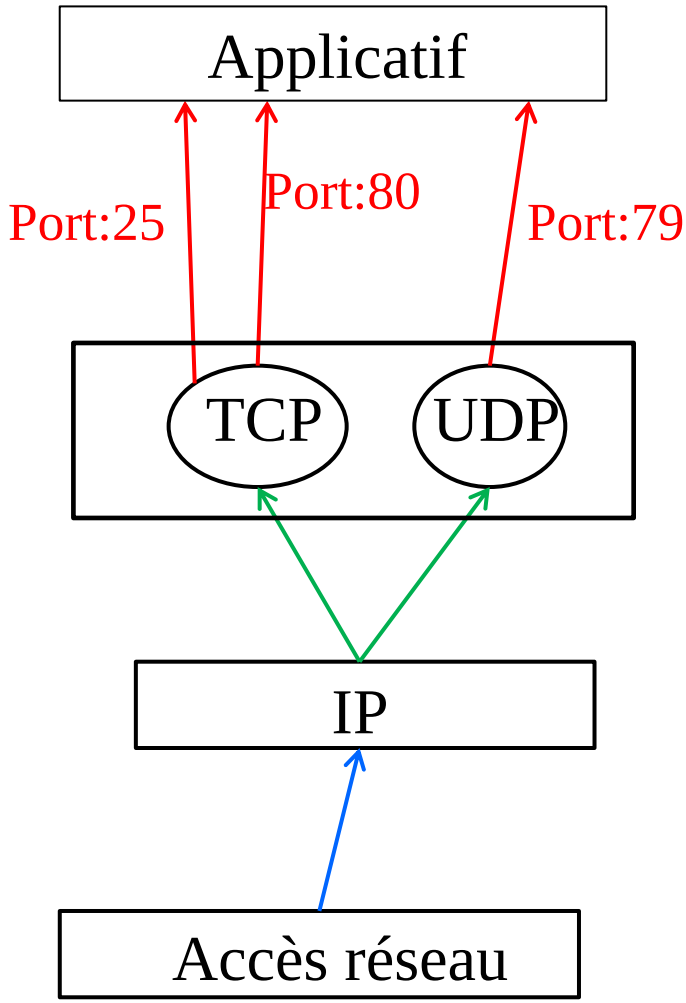
TCP

- mode connecté → canaux de transmission indépendants
- files d'attente des octets en attente de livraison
- établissement de connexion TCP = création des 2 canaux
- terminaison de connexion = destruction des canaux
- 3 phases (mode connecté)
  - Établissement de la communication *(triple poignée de main)*
  - Transmission des données
  - Terminaison de la communication



Port d'une connexion (1)

- Identifier un processus distant
  - *3 informations obligatoires :*
    - L'adresse IP
    - Le protocole utilisé
    - Le numéro de port
      - Privilégié/ connu
      - Enregistré
      - Privés /éphémères



Certaines informations sont implicites ou configurables pour certaines applications

Port d'une connexion (2)

- Port applicatif
  - Unicité d'un port par applicatif
  - Privilégié (<=1024)
    - 22 : ssh , 23 : telnet , 80 : http, 110 : POP3, 143: IMAP…
  - Enregistré > 1024
    - 8080 : alternate http, 6000 : serveur X, 3724 : warcraft,..
  - Plusieurs états d'un port en TCP:
    - Inexistant : aucune communication possible
    - LISTEN : en attente d'une connexion
    - ESTABLISHED : en communication
    - TIME_WAIT : fermeture de la connexion

Socket

Le concept d'accès au Service TCP (resp. UDP) est traduit par le concept de socket des systèmes d'exploitation

- création de *socket* selon les besoins des processus
  - par la fonction **`socket()`**
  - une *socket* appartient à un processus
  - une *socket* est détruite au plus tard à la mort du processus

- attribution d'une adresse de point d'accès (numéro de port)
  - par la fonction **`bind()`**
  - des restrictions sur le numéro de port (un seul processus par port)

- Socket en mode LISTEN = porte ouverte sur l'ordinateur…..

→ netstat -an

Transmission de données

Les données sont transmises dans des PDU-TCP (resp.PDU-UDP), transportées par le Service IP qui livre ... peut-être, une seule fois

Les risques :

- perte de PDU-IP → perte de la PDU-TCP ou PDU-UDP encapsulée et des données qu'elle contient
- livraisons multiples de la même PDU (duplication), donc de données
- livraison d'une suite d'octets dans un ordre différent de celui de la soumission

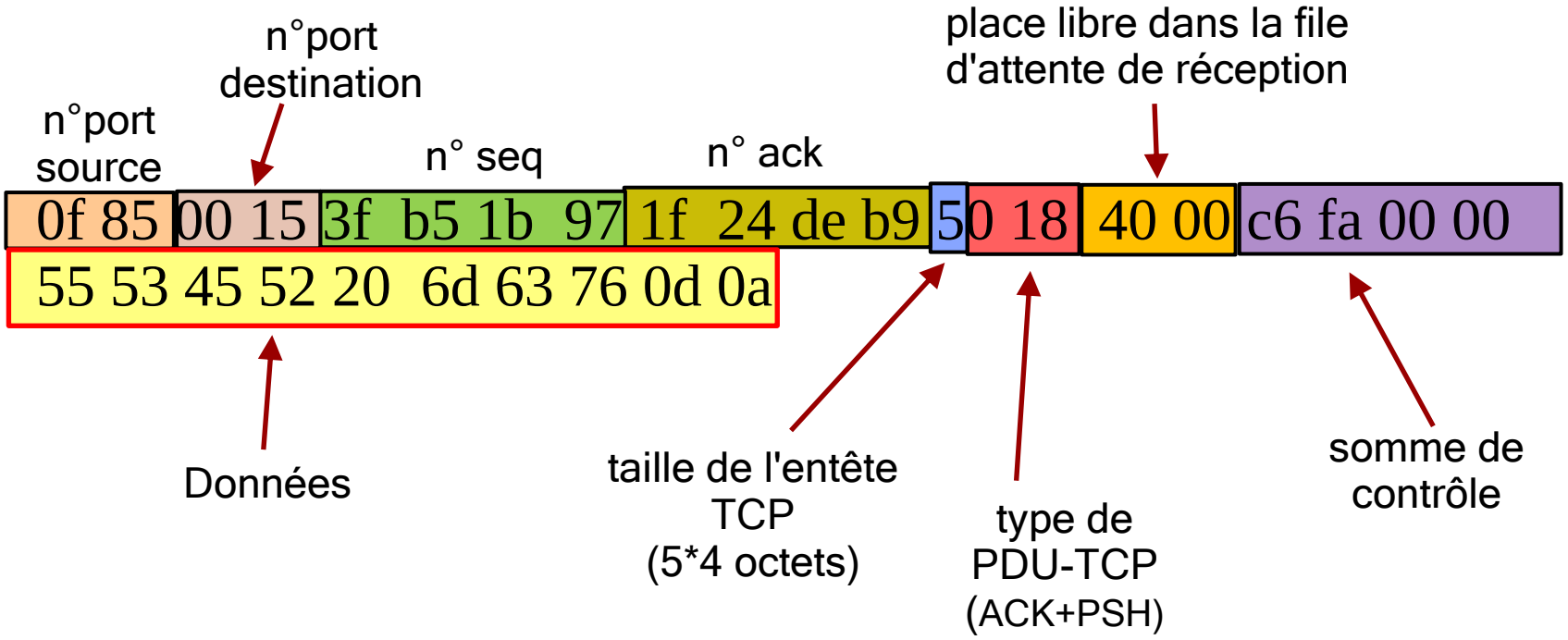
UDP ne fait ni détection, ni correction

TCP tente de détecter tous les problèmes et de les corriger

Fiabilité de TCP

- Numérotation lors de l'expédition
  - les octets de l'utilisateur sont transmis par bloc dans des PDU-TCP
  - l'entité-TCP calcule un numéro de séquence qui correspond "aux nombres d'octets envoyés" et le place dans la PDU envoyé, cela correspond au numéro **seq** : séquence
  - initialisation de seq : à l'établissement de connexion
  - Calcul lors de la phase d'initialisation d'un nombre correspondant "aux nombres d'octets reçus" (aussi envoyé dans la PDU), c'est le numéro **ack** : acquittement
- Vérification lors de la réception
  - n° reçu > n° attendu → une perte
  - n° reçu < n° attendu → une duplication
  - n° reçu = n° attendu → OK
- Correction
  - duplication : l'entité-TCP ignore la PDU-TCP en doublon
  - perte : retransmission de la PDU contenant le bloc
    - on renvoie la partie perdue

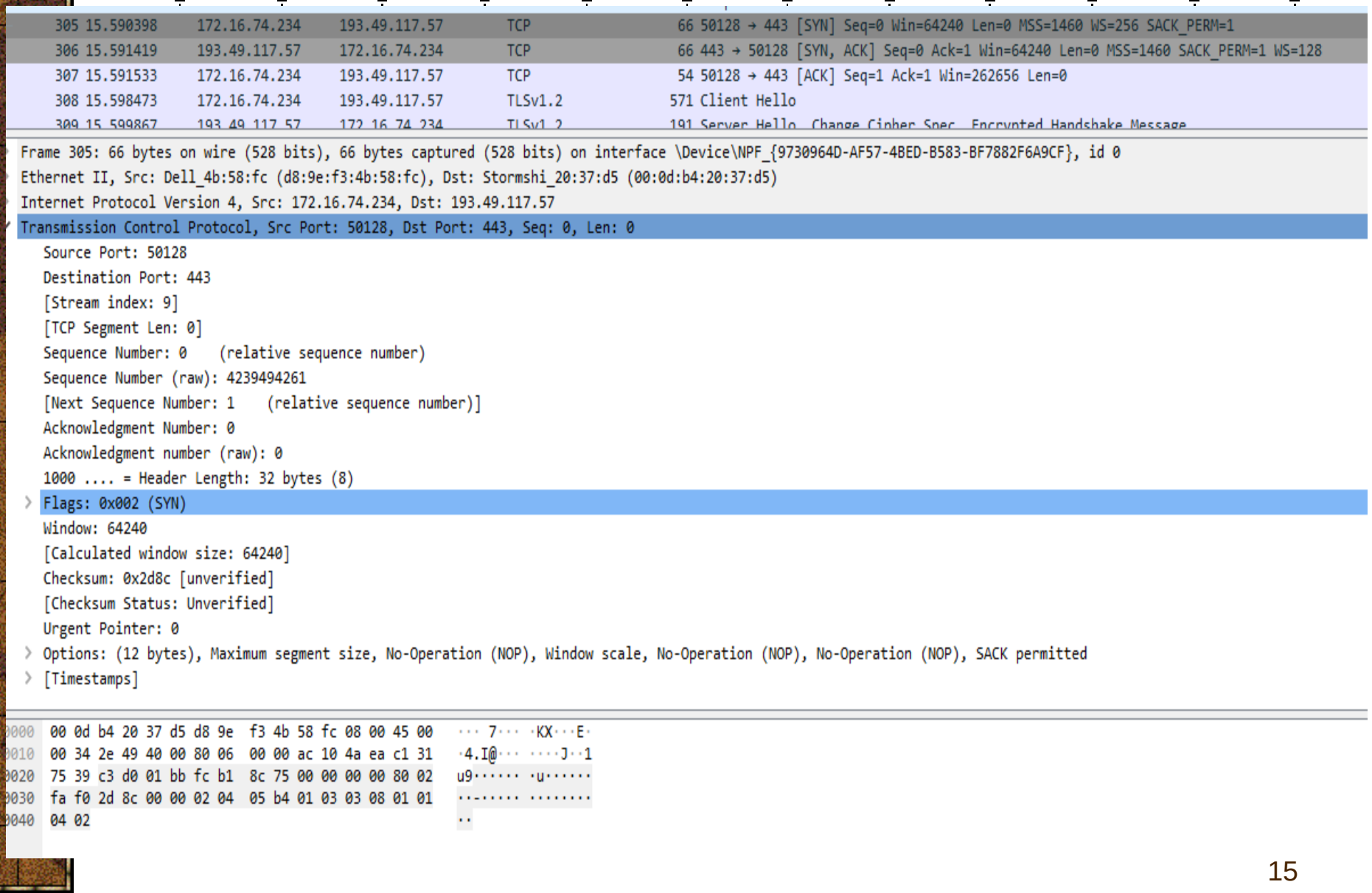
En-tête de la PDU-TCP



type de PDU-TCP

URG	ACK	PSH	RST	SYN	FIN
20	10	8	4	2	1

Wireshark


305 15.590398 172.16.74.234 193.49.117.57 TCP 66 50128 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
306 15.591419 193.49.117.57 172.16.74.234 TCP 66 443 → 50128 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
307 15.591533 172.16.74.234 193.49.117.57 TCP 54 50128 → 443 [ACK] Seq=1 Ack=1 Win=262656 Len=0
308 15.598473 172.16.74.234 193.49.117.57 TLSv1.2 571 Client Hello
309 15.599867 193.49.117.57 172.16.74.234 TLSv1.2 191 Server Hello, Change Cipher Spec, Encrypted Handshake Message
Frame 305: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{9730964D-AF57-4BED-B583-BF7882F6A9CF}, id 0
Ethernet II, Src: Dell_4b:58:fc (d8:9e:f3:4b:58:fc), Dst: Stormshi_20:37:d5 (00:0d:b4:20:37:d5)
Internet Protocol Version 4, Src: 172.16.74.234, Dst: 193.49.117.57
Transmission Control Protocol, Src Port: 50128, Dst Port: 443, Seq: 0, Len: 0
Source Port: 50128
Destination Port: 443
[Stream index: 9]
[TCP Segment Len: 0]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 4239494261
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
1000 .... = Header Length: 32 bytes (8)
Flags: 0x002 (SYN)
Window: 64240
[Calculated window size: 64240]
Checksum: 0x2d8c [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted
[Timestamps]
0000 00 0d b4 20 37 d5 d8 9e f3 4b 58 fc 08 00 45 00
0010 00 34 2e 49 40 00 80 06 00 00 ac 10 4a ea c1 31
0020 75 39 c3 d0 01 bb fc b1 8c 75 00 00 00 00 80 02
0030 fa f0 2d 8c 00 00 02 04 05 b4 01 03 03 08 01 01
0040 04 02

Applicatifs

Généralité

Notion client/serveur

- Serveur : **en attente de connexion → un port ouvert**
- *Client* : *veut une information détenue par le serveur*
  - **Connait le nom ou l'@IP du serveur,**
  - **Le numéro de port et le protocole utilisé (TCP/UDP)**
  - **La requête à effectuer et sa syntaxe**

En général, client et serveur sont sur 2 ordinateurs distincts

Client

- processus créés lorsque le besoin apparaît
  - par une personne
  - au démarrage du système
- l'entité-client doit savoir à quelle entité-serveur elle doit/peut s'adresser
  - Soit au lancement du processus : fichier de configuration, argument
  - Soit par saisie pendant l'exécution

Côté serveur

- Les entités serveur
  - fonctionnement permanent (démons)
    - processus créés au démarrage du système
  - processus lancés lorsqu'une demande arrive
    - par un processus "écouteur"
- Méthodes de prise en charge d'une demande de Service
  - L'entité serveur peut traiter elle-même la demande
    - Mise en attente des autres demandes jusqu'à la fin du traitement
  - L'entité serveur peut déléguer à un processus fils (ou un thread) le traitement de la demande
    - Contrôle du nombre de fils (ou de thread) ->nombre maximum donné, pré-création …
    - Risque de surcharge de la mémoire

ICMP (1)

- *ICMP : Internet Control Message Protocol*
  - Permet le contrôle des erreurs de transmission, basé sur IP
  - Composé principalement :
    - D'un entête IP
    - D'un type de message
  - Différents types de message :
    - Type 0 : echo réponse
      → réponse au type 8
    - Type 3 : destinataire inaccessible
    - Type 5 : redirection
    - Type 8 : demande echo
      → permet de savoir si une @IP répond (ping)

ICMP (2)

202	10.996855	172.16.74.234	8.8.8.8	ICMP	74	Echo (ping) request id=0x000
203	11.005729	8.8.8.8	172.16.74.234	ICMP	74	Echo (ping) reply id=0x000
232	12.012117	172.16.74.234	8.8.8.8	ICMP	74	Echo (ping) request id=0x000
233	12.020912	8.8.8.8	172.16.74.234	ICMP	74	Echo (ping) reply id=0x000
273	13.030993	172.16.74.234	8.8.8.8	ICMP	74	Echo (ping) request id=0x000
275	13.039917	8.8.8.8	172.16.74.234	ICMP	74	Echo (ping) reply id=0x000

```
Frame 232: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{9730964D-AF57-4BED-
Ethernet II, Src: Dell_4b:58:fc (d8:9e:f3:4b:58:fc), Dst: Stormshi_20:37:d5 (00:0d:b4:20:37:d5)
  Destination: Stormshi_20:37:d5 (00:0d:b4:20:37:d5)
  Source: Dell_4b:58:fc (d8:9e:f3:4b:58:fc)
  Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 172.16.74.234, Dst: 8.8.8.8
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 60
  Identification: 0xb413 (46099)
  Flags: 0x00
  Fragment Offset: 0
  Time to Live: 128
  Protocol: ICMP (1)
  Header Checksum: 0x0000 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.16.74.234
  Destination Address: 8.8.8.8
Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0x4d52 [correct]
  [Checksum Status: Good]
  Identifier (BE): 1 (0x0001)
  Identifier (LE): 256 (0x0100)
  Sequence Number (BE): 9 (0x0009)
  Sequence Number (LE): 2304 (0x0900)
  [Response frame: 233]
  Data (32 bytes)
    Data: 6162636465666768696a6b6c6d6e6f707172737475767761626364656667686 9
    [Length: 32]
```

```
0000  00 0d b4 20 37 d5 d8 9e  f3 4b 58 fc 08 00 45 00   ··· 7··· ·KX···E·
0010  00 3c b4 13 00 00 80 01  00 00 ac 10 4a ea 08 08   ·<·····  ····J···
0020  08 08 08 00 4d 52 00 01  00 09 61 62 63 64 65 66   ····MR··  ··abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                    wabcdefg hi
```

DHCP : définitions

DHCP : Dynamic Host Configuration Protocol -> RFC 2131

(successeur de BOOTP)

utilisation de la couche transport UDP : port 67 client ->serveur

port 68 serveur -> client

• Utilisation pour récupération automatique @IP, **etc**… sur un ordinateur

Serveur fournit (en général) :

- **Adresse IP**
- **Masque de réseau**
- **Adresse de la passerelle**
- **Adresse du DNS, nom du domaine**

Un serveur a un « pool » d'adresses qu'il peut distribuer

➡ utilisation d'un bail sur la durée de location d'une adresse

DHCP : fonctionnement

1. Emission d'un broadcast pour la demande d'une adresse IP
   ->DHCPDISCOVER
2. Réponse en unicast d'un serveur vers le client en lui spécifiant les différents paramètres à utiliser
   (auparavant le serveur fait 2 pings pour savoir si l'adresse qu'il propose n'est pas déjà utilisée…)
   ->DHCPOFFER
3. Client donne son accord sur cette adresse en faisant un broadcast
   -> DHCPREQUEST
4. Le serveur acquitte cette accord en répondant en unicast
   -> *DHCPACK*

Pour libérer une adresse IP, utilisation de *DHCPRELEASE*

Utilisable que sur le réseau LAN (par défaut)

Abandonné pour IPv6, et remplacé par ICMPv6

DNS (1)

- Domain Name System
- Pourquoi ?
  - les équipements communiquent grâce à leur adresse IP
    (table de routage,…)
  - Mais il est plus simple de retenir un nom qu'une suite de chiffre !!!
    - actuellement 4 octets, et 16 avec IPv6…

Un des rôles du DNS

Convertir les noms en adresse IP

Une @ IP est unique → nom unique

organisme : ICANN -> IANA

DNS (2)

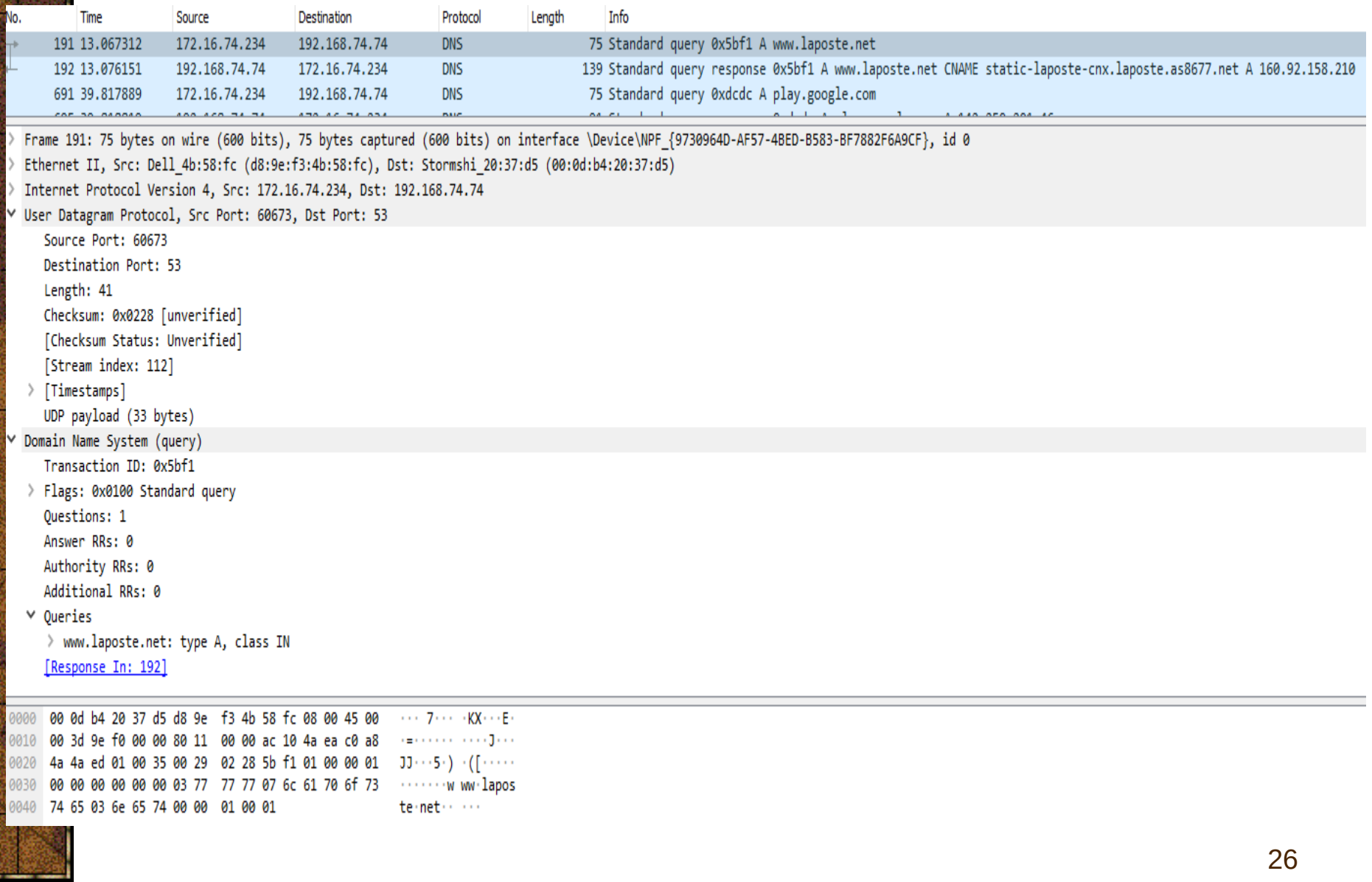
- Fonctionnement
  - Recherche en local dans le fichier

    /etc/hosts ou c:\windows\system32\drivers\etc\hosts
  - Requête DNS
    - Requête récursive
      - 1er étape vers son DNS local
        - Si réponse connue, fin
        - Sinon, envoie de la requête vers un DNS plus global
          - Etc…
    - 13 serveurs DNS mondiaux

      1375 instances

Source root-servers.org

DNS (3)



No.	Time	Source	Destination	Protocol	Length	Info
191	13.067312	172.16.74.234	192.168.74.74	DNS	75	Standard query 0x5bf1 A www.laposte.net
192	13.076151	192.168.74.74	172.16.74.234	DNS	139	Standard query response 0x5bf1 A www.laposte.net CNAME static-laposte-cnx.laposte.as8677.net A 160.92.158.210
691	39.817889	172.16.74.234	192.168.74.74	DNS	75	Standard query 0xdcdc A play.google.com

```
> Frame 191: 75 bytes on wire (600 bits), 75 bytes captured (600 bits) on interface \Device\NPF_{9730964D-AF57-4BED-B583-BF7882F6A9CF}, id 0
> Ethernet II, Src: Dell_4b:58:fc (d8:9e:f3:4b:58:fc), Dst: Stormshi_20:37:d5 (00:0d:b4:20:37:d5)
> Internet Protocol Version 4, Src: 172.16.74.234, Dst: 192.168.74.74
v User Datagram Protocol, Src Port: 60673, Dst Port: 53
    Source Port: 60673
    Destination Port: 53
    Length: 41
    Checksum: 0x0228 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 112]
  > [Timestamps]
    UDP payload (33 bytes)
v Domain Name System (query)
    Transaction ID: 0x5bf1
  > Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  v Queries
    > www.laposte.net: type A, class IN
    [Response In: 192]
```

```
0000  00 0d b4 20 37 d5 d8 9e  f3 4b 58 fc 08 00 45 00   ··· 7··· ·KX···E·
0010  00 3d 9e f0 00 00 80 11  00 00 ac 10 4a ea c0 a8   ·=·····  ····J···
0020  4a 4a ed 01 00 35 00 29  02 28 5b f1 01 00 00 01   JJ···5·)  ·([·····
0030  00 00 00 00 00 00 03 77  77 77 07 6c 61 70 6f 73   ·······w ww·lapos
0040  74 65 03 6e 65 74 00 00  01 00 01                 te·net··  ···
```

Protocole HTTP

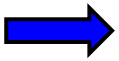
- HTTP : HyperText Transfert Protocol
  (RFC 1945 et 2616)
  - Protocole de rapatriement des documents
  - Protocole de soumission de formulaire
  - 2 versions un peu différente : HTTP 1.0 et 1.1
  - Fonctionnement au dessus d'une connexion en TCP
    - Utilisation d'une URI (Uniform Resource identifier)
    - http://<nom_machine>:<port>/<path>?<query>
  - Version plus récente : HTTP 2.0 -> RFC 7540
  - Nouvelle version HTTP 3.0 (2022/2023) RFC 9000
    ancien nom : http 2-over-quic (udp)

Méthode HTTP

- Quelques méthodes utilisées
  - **méthode GET**
    - récupération d'un document
  - **méthode POST**
    - envoi de données au programme situé à l'URL spécifié
  - **méthode HEAD**
    - récupération des informations sur le document (lignes d'en tête)

- A chaque requête
  - Envoie d'un GET ou d'un POST
  - Réponse de la part du serveur WEB

GET / HTTP/1.1\r\n
Host: www.isima.fr\r\n
Accept : text/html, application/xhtml+xml\r\n
Accept-Encoding : gzip, deflate\r\n
Connection : keep-alive\r\n
\r\n



HTTP/1.1 200 OK\r\n
Date: Tue, 22 Apr 2014 09:48:55 GMT\r\n
Server: Apache/2.4.4 (Win32) PHP/5.4.14\r\n
Last-Modified: Wed, Feb 2014 15:57:14 \r\n
Content-Length: 125\r\n
Keep-Alive: timeout=5, max=99\r\n
Connection: Keep-Alive\r\n etc…

Applicatifs

- De nombreux applicatifs utilisant des protocoles réseaux existent
  - SMTP (mail), RPC (connexion à distance), NTP (temps universel), NFS (partage de fichier), XMPP (messagerie instantanée), RIP (routage), BGP (routage),….
  - Chaque protocole est défini dans une RFC
  - A chaque protocole correspond un N° port

    Donc **N°port ⟷ applicatif**

Réseau

- A partir d'un PC éteint de l'ISIMA, on veut aller sur la page web de l'UCA.
  Combien de trames passeront sur le réseau ?

Réseau

- A partir d'un PC éteint de l'ISIMA, on veut aller sur la page web de l'UCA. Combien de trames passeront sur le réseau ?

➢ récupération d'une adresse IP, masque , etc… par DHCP

- 4 trames + 2 ping = *6 trames*

➢ requête DNS pour connaitre l'@IP du serveur web de l'UCA

- 2 trames pour le DNS + 2 trames pour ARP = *4 trames*

➢ Création de la trame pour aller vers l'UCA

- utilisation de la table de routage → @IP routeur en sortie ISIMA
- 2 trames pour ARP @MAC du routeur = *2 trames*

➢ Connexion TCP sur le serveur UCA -> *3 trames*

➢ Envoie de la requête http et réponse → *2 trames min*

- mais acquittement TCP -> *2 trames*

➢ Fin connexion TCP -> *4 trames* somme = *11 trames min*

Total final = 23 trames

Sécurisation

Pare-feu

- Un pare-feu (firewall en anglais), est un équipement ou un logiciel qui permet de protéger un réseau local des intrusions de personnes en provenance d'Internet, donc du réseau extérieur à l'entreprise.

C'est un système permettant de bloquer **des ports et des adresses IP**, c'est-à-dire en interdire l'accès aux personnes provenant de l'extérieur

- Fonctionnement

Filtrage des paquets:
-> analyse de l'entête des paquets TCP (UDP) et de l'entête du paquet IP

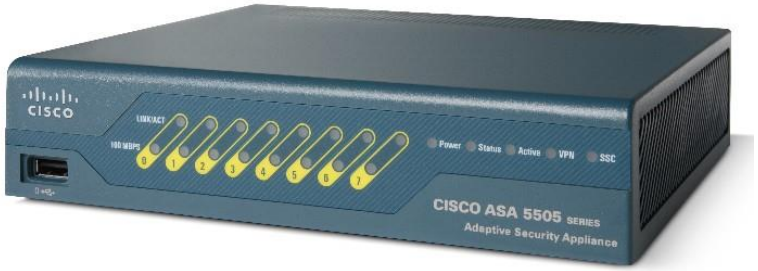
exemple filtre suivant :

L'adresse IP de la machine émettrice

L'adresse IP de la machine réceptrice

Le protocole (TCP, UDP, ICMP, ARP, ...)

Le numéro de port, …



Proxy

- Un proxy est une machine intermédiaire entre 2 stations qui peut modifier les données protocolaires (entêtes)
- Utilisation actuelle:
  - permettre aux ordinateurs du LAN d'accéder à internet par son intermédiaire (fonction NAT)
  - servir de cache, c'est-à-dire garder en mémoire les pages les plus souvent visitées pour pouvoir les fournir plus rapidement, on l'appelle alors serveur proxy-cache. Lorsque l'on demande une page html a un navigateur, le navigateur interroge d'abord le proxy
    Un proxy peut automatiquement mettre à jour son cache
    Il peut aussi interdire l'accès à certains site (~firewall niveau 7)
  - jouer le rôle de pare-feu

Sonde

- IPS (Intrusion Prevention System)
- IDS (Intrusion Detection System)

  - Observation de tous les paquets passant sur le réseau
  - Objectif : détection des menaces
    - Virus, paquet mal formé, scan de ports, etc…
    - Utilisation d'une base de données des menaces….

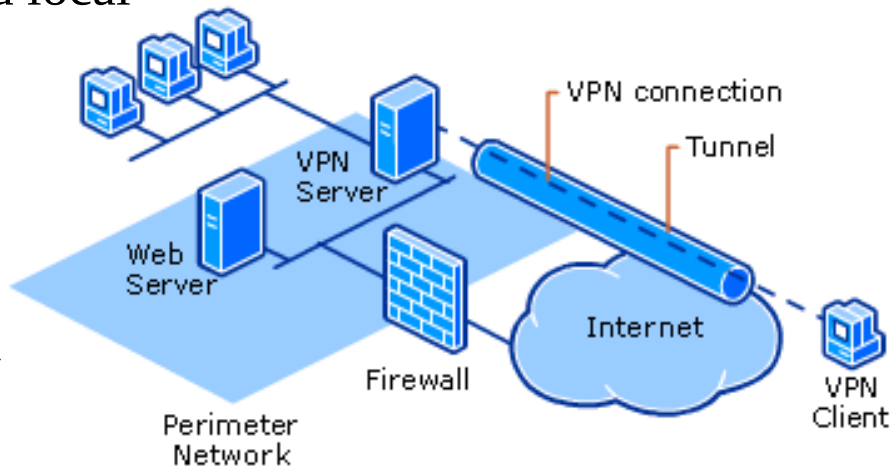
  - IDS : détecte les problèmes et avertit
  - IPS : détecte le problème et coupe la liaison

VPN

- VPN (Virtual Private Network)
- Objectif :
  - **UN VPN permet à des ordinateurs distants d'être considéré comme faisant partie du réseau local.**
- Besoin:
  - Un client VPN (sur la machine cliente)
  - Un serveur VPN dans le réseau local

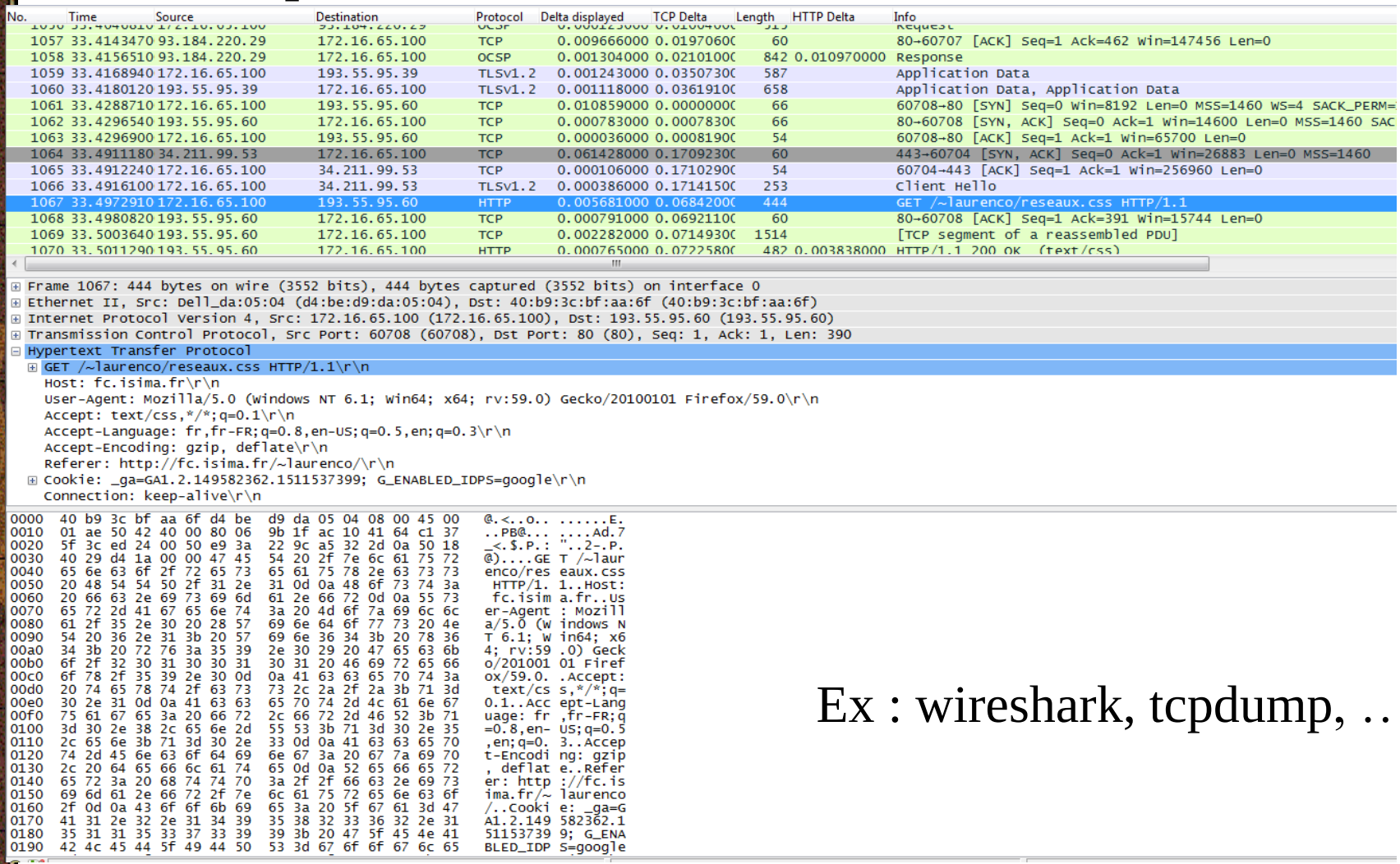
Ordinateur distant récupère
une adresse IP du réseau local

Chiffrement des données
entre client VPN et serveur VPN



Sniffer

- Récupération brute des données



No.	Time	Source	Destination	Protocol	Delta displayed	TCP Delta	Length	HTTP Delta	Info
1057	33.4143470	93.184.220.29	172.16.65.100	TCP	0.009666000	0.0197060(	60		80→60707 [ACK] Seq=1 Ack=462 Win=147456 Len=0
1058	33.4156510	93.184.220.29	172.16.65.100	OCSP	0.001304000	0.0210100(	842	0.010970000	Response
1059	33.4168940	172.16.65.100	193.55.95.39	TLSv1.2	0.001243000	0.0350730(	587		Application Data
1060	33.4180120	193.55.95.39	172.16.65.100	TLSv1.2	0.001118000	0.0361910(	658		Application Data, Application Data
1061	33.4288710	172.16.65.100	193.55.95.60	TCP	0.010859000	0.0000000(	66		60708→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=
1062	33.4296540	193.55.95.60	172.16.65.100	TCP	0.000783000	0.0007830(	66		80→60708 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460 SAC
1063	33.4296900	172.16.65.100	193.55.95.60	TCP	0.000036000	0.0008190(	54		60708→80 [ACK] Seq=1 Ack=1 Win=65700 Len=0
1064	33.4911180	34.211.99.53	172.16.65.100	TCP	0.061428000	0.1709230(	60		443→60704 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=1460
1065	33.4912240	172.16.65.100	34.211.99.53	TCP	0.000106000	0.1710290(	54		60704→443 [ACK] Seq=1 Ack=1 Win=256960 Len=0
1066	33.4916100	172.16.65.100	34.211.99.53	TLSv1.2	0.000386000	0.1714150(	253		Client Hello
1067	33.4972910	172.16.65.100	193.55.95.60	HTTP	0.005681000	0.0684200(	444		GET /~laurenco/reseaux.css HTTP/1.1
1068	33.4980820	193.55.95.60	172.16.65.100	TCP	0.000791000	0.0692110(	60		80→60708 [ACK] Seq=1 Ack=391 Win=15744 Len=0
1069	33.5003640	193.55.95.60	172.16.65.100	TCP	0.002282000	0.0714930(	1514		[TCP segment of a reassembled PDU]
1070	33.5011290	193.55.95.60	172.16.65.100	HTTP	0.000765000	0.0722580(	482	0.003838000	HTTP/1.1 200 OK (text/css)

```
⊞ Frame 1067: 444 bytes on wire (3552 bits), 444 bytes captured (3552 bits) on interface 0
⊞ Ethernet II, Src: Dell_da:05:04 (d4:be:d9:da:05:04), Dst: 40:b9:3c:bf:aa:6f (40:b9:3c:bf:aa:6f)
⊞ Internet Protocol Version 4, Src: 172.16.65.100 (172.16.65.100), Dst: 193.55.95.60 (193.55.95.60)
⊞ Transmission Control Protocol, Src Port: 60708 (60708), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 390
⊟ Hypertext Transfer Protocol
  ⊞ GET /~laurenco/reseaux.css HTTP/1.1\r\n
    Host: fc.isima.fr\r\n
    User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:59.0) Gecko/20100101 Firefox/59.0\r\n
    Accept: text/css,*/*;q=0.1\r\n
    Accept-Language: fr,fr-FR;q=0.8,en-US;q=0.5,en;q=0.3\r\n
    Accept-Encoding: gzip, deflate\r\n
    Referer: http://fc.isima.fr/~laurenco/\r\n
  ⊞ Cookie: _ga=GA1.2.149582362.1511537399; G_ENABLED_IDPS=google\r\n
    Connection: keep-alive\r\n
```

```
0000  40 b9 3c bf aa 6f d4 be  d9 da 05 04 08 00 45 00   @.<..o.. ......E.
0010  01 ae 50 42 40 00 80 06  9b 1f ac 10 41 64 c1 37   ..PB@... ....Ad.7
0020  5f 3c ed 24 00 50 e9 3a  22 9c a5 32 2d 0a 50 18   _<.$.P.: "..2-.P.
0030  40 29 d4 1a 00 00 47 45  54 20 2f 7e 6c 61 75 72   @)....GE T /~laur
0040  65 6e 63 6f 2f 72 65 73  65 61 75 78 2e 63 73 73   enco/res eaux.css
0050  20 48 54 54 50 2f 31 2e  31 0d 0a 48 6f 73 74 3a    HTTP/1. 1..Host:
0060  20 66 63 2e 69 73 69 6d  61 2e 66 72 0d 0a 55 73    fc.isim a.fr..Us
0070  65 72 2d 41 67 65 6e 74  3a 20 4d 6f 7a 69 6c 6c   er-Agent : Mozill
0080  61 2f 35 2e 30 20 28 57  69 6e 64 6f 77 73 20 4e   a/5.0 (W indows N
0090  54 20 36 2e 31 3b 20 57  69 6e 36 34 3b 20 78 36   T 6.1; W in64; x6
00a0  34 3b 20 72 76 3a 35 39  2e 30 29 20 47 65 63 6b   4; rv:59 .0) Geck
00b0  6f 2f 32 30 31 30 30 31  30 31 20 46 69 72 65 66   o/201001 01 Firef
00c0  6f 78 2f 35 39 2e 30 0d  0a 41 63 63 65 70 74 3a   ox/59.0. .Accept:
00d0  20 74 65 78 74 2f 63 73  73 2c 2a 2f 2a 3b 71 3d    text/cs s,*/*;q=
00e0  30 2e 31 0d 0a 41 63 63  65 70 74 2d 4c 61 6e 67   0.1..Acc ept-Lang
00f0  75 61 67 65 3a 20 66 72  2c 66 72 2d 46 52 3b 71   uage: fr ,fr-FR;q
0100  3d 30 2e 38 2c 65 6e 2d  55 53 3b 71 3d 30 2e 35   =0.8,en- US;q=0.5
0110  2c 65 6e 3b 71 3d 30 2e  33 0d 0a 41 63 63 65 70   ,en;q=0. 3..Accep
0120  74 2d 45 6e 63 6f 64 69  6e 67 3a 20 67 7a 69 70   t-Encodi ng: gzip
0130  2c 20 64 65 66 6c 61 74  65 0d 0a 52 65 66 65 72   , deflat e..Refer
0140  65 72 3a 20 68 74 74 70  3a 2f 2f 66 63 2e 69 73   er: http ://fc.is
0150  69 6d 61 2e 66 72 2f 7e  6c 61 75 72 65 6e 63 6f   ima.fr/~ laurenco
0160  2f 0d 0a 43 6f 6f 6b 69  65 3a 20 5f 67 61 3d 47   /..Cooki e: _ga=G
0170  41 31 2e 32 2e 31 34 39  35 38 32 33 36 32 2e 31   A1.2.149 582362.1
0180  35 31 31 35 33 37 33 39  39 3b 20 47 5f 45 4e 41   51153739 9; G_ENA
0190  42 4c 45 44 5f 49 44 50  53 3d 67 6f 6f 67 6c 65   BLED_IDP S=google
```

Ex : wireshark, tcpdump, …