



R2.05 – Les protocoles reseaux

RESPONSABLE : CRISTINA ONETE

MATERIEL : [HTTPS://WWW.ONETE.NET/TEACHING.HTML](https://www.onete.net/teaching.html)

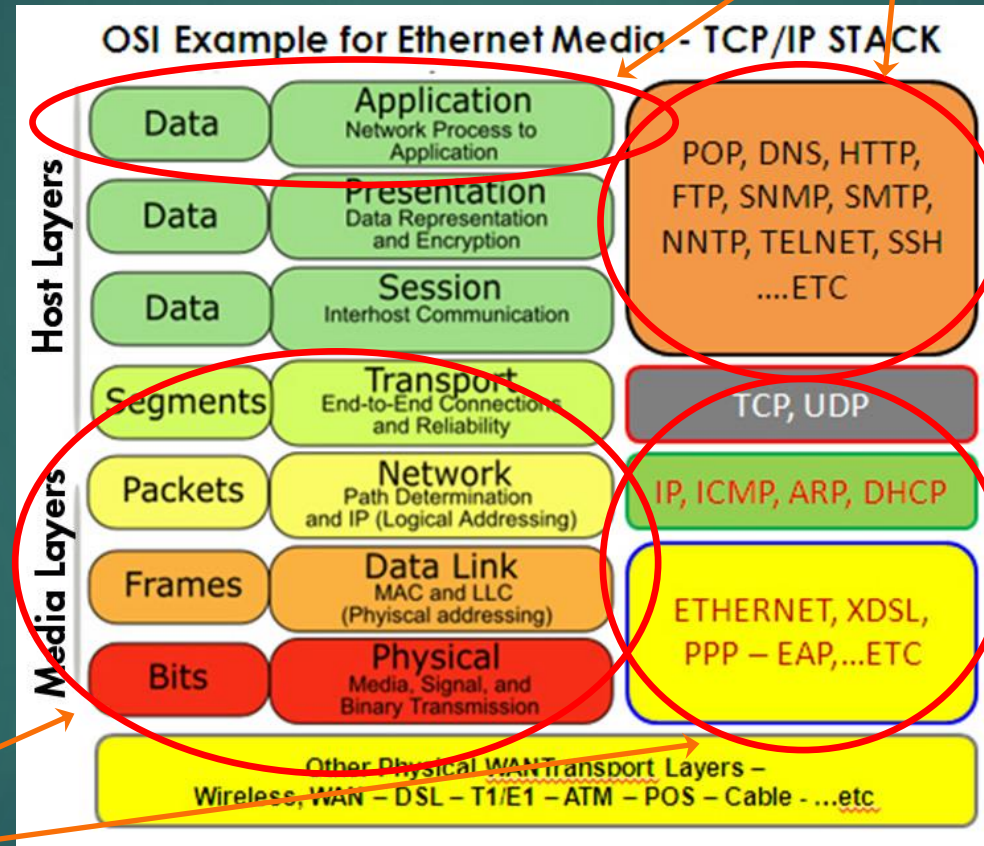
EMAIL : CRISTINA.ONETE@GMAIL.COM

R2.04, R2.05

2

R2.05

Les couches architecturales des réseaux



Les protocoles des réseaux

R2.04

Source : ipv6.com

Aujourd'hui : des applications

4

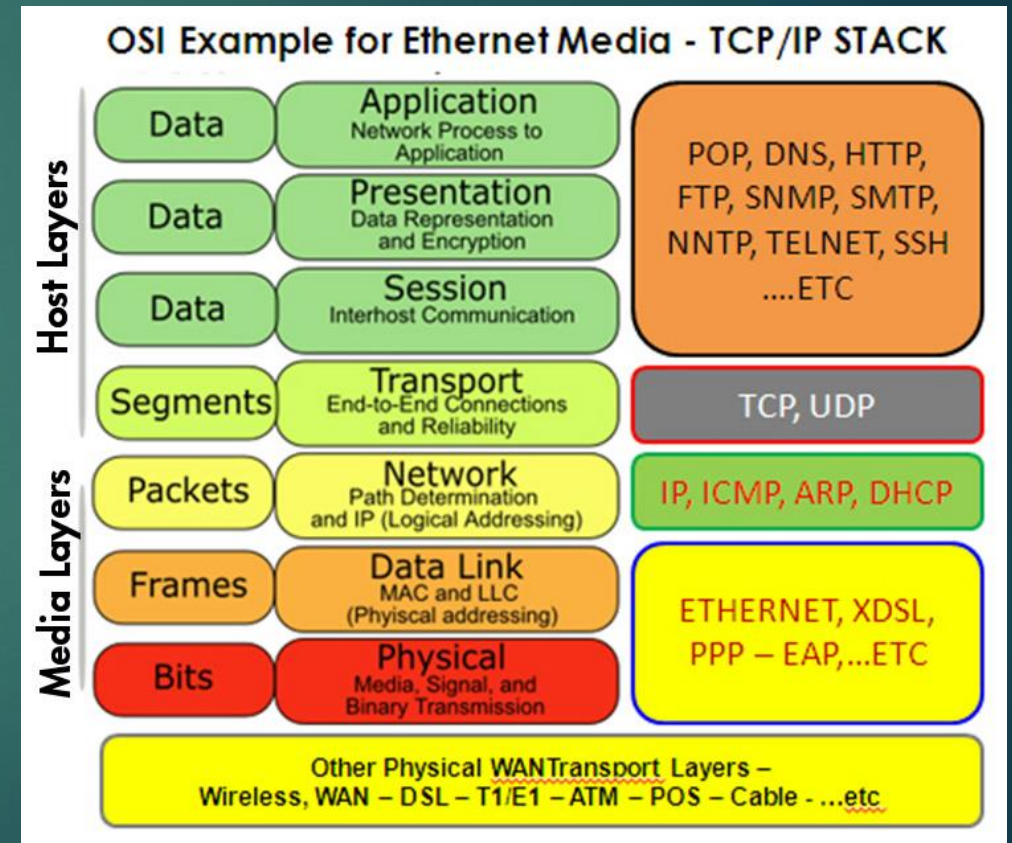
données des applications



messages pour plusieurs applications

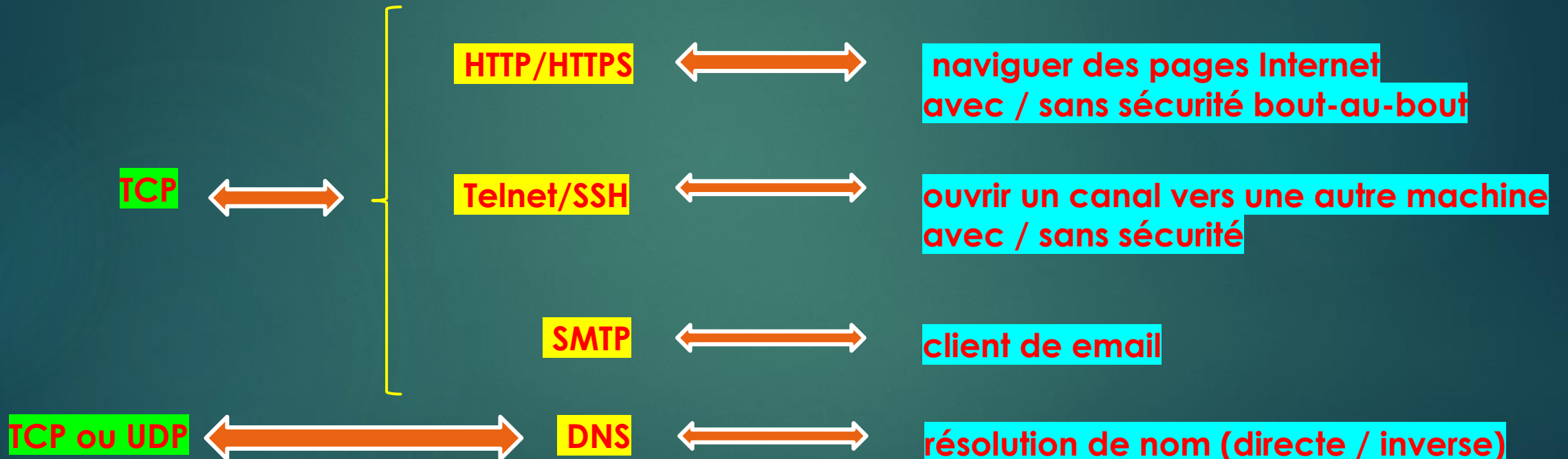


séparation : chaque application $\leftrightarrow$ port



Quelques protocoles typiques

5



HTTP(S) : la navigation Internet

6

► Une connexion Internet :



Modèle client-serveur

Client :

- application envoie des requêtes
- initialise le contact
- actif

Serveur :

- application répond aux requêtes
- à l'attente
- sur Apache, Windows Server

Côté client

7

Processus

```
root      2094      1  0  10:03  ?           Ss      0:00  /usr/sbin/apache2 -k
www-data  2162     2094  0  10:03  ?           S       0:00  \_ /usr/sbin/apache2
www-data  2163     2094  0  10:03  ?           S       0:00  \_ /usr/sbin/apache2
www-data  2164     2094  0  10:03  ?           S       0:00  \_ /usr/sbin/apache2
www-data  2165     2094  0  10:03  ?           S       0:00  \_ /usr/sbin/apache2
www-data  2166     2094  0  10:03  ?           S       0:00  \_ /usr/sbin/apache2
```

6 processus pour traiter
des requêtes

commande ss, options Intp
(remplace netstat)

Réseau

```
root@debian7-cli:~# netstat -Intp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 0.0.0.0:80               0.0.0.0:*               LISTEN
3389/apache2
```

Connexion se fait sur TCP

Requête attendue, port 80

État d'écoute : LISTEN

HTTP sur Wireshark

8

No.	Time	Source	Destination	Protocol	Info	
1	12:27:44.481737	192.168.56.254	192.168.56.101	TCP	55986 → 80 [SYN] Seq=0 Win=29200 L	initialization connexion TCP
2	12:27:44.482070	192.168.56.101	192.168.56.254	TCP	80 → 55986 [SYN, ACK] Seq=0 Ack=1	
3	12:27:44.482117	192.168.56.254	192.168.56.101	TCP	55986 → 80 [ACK] Seq=1 Ack=1 Win=2	
4	12:27:44.482288	192.168.56.254	192.168.56.101	HTTP	GET / HTTP/1.1	requête HTTP
5	12:27:44.482384	192.168.56.101	192.168.56.254	TCP	80 → 55986 [ACK] Seq=1 Ack=369 win	
6	12:27:44.483092	192.168.56.101	192.168.56.254	HTTP	HTTP/1.1 200 OK (text/html)	réponse HTTP (données)
7	12:27:44.483111	192.168.56.254	192.168.56.101	TCP	55986 → 80 [ACK] Seq=369 Ack=370 w	
8	12:27:44.530072	192.168.56.254	192.168.56.101	HTTP	GET /favicon.ico HTTP/1.1	
9	12:27:44.531124	192.168.56.101	192.168.56.254	HTTP	HTTP/1.1 404 Not Found (text/html	
10	12:27:44.531180	192.168.56.254	192.168.56.101	TCP	55986 → 80 [ACK] Seq=680 Ack=873 w	
11	12:27:49.536687	192.168.56.101	192.168.56.254	TCP	80 → 55986 [FIN, ACK] Seq=873 Ack=	fin de la connexion TCP
12	12:27:49.573712	192.168.56.254	192.168.56.101	TCP	55986 → 80 [ACK] Seq=680 Ack=874 w	
13	12:28:16.122995	192.168.56.254	192.168.56.101	TCP	55986 → 80 [FIN, ACK] Seq=680 Ack=	
14	12:28:16.123288	192.168.56.101	192.168.56.254	TCP	80 → 55986 [ACK] Seq=874 Ack=681 w	

client

port : TCP/UDP

port : HTTP

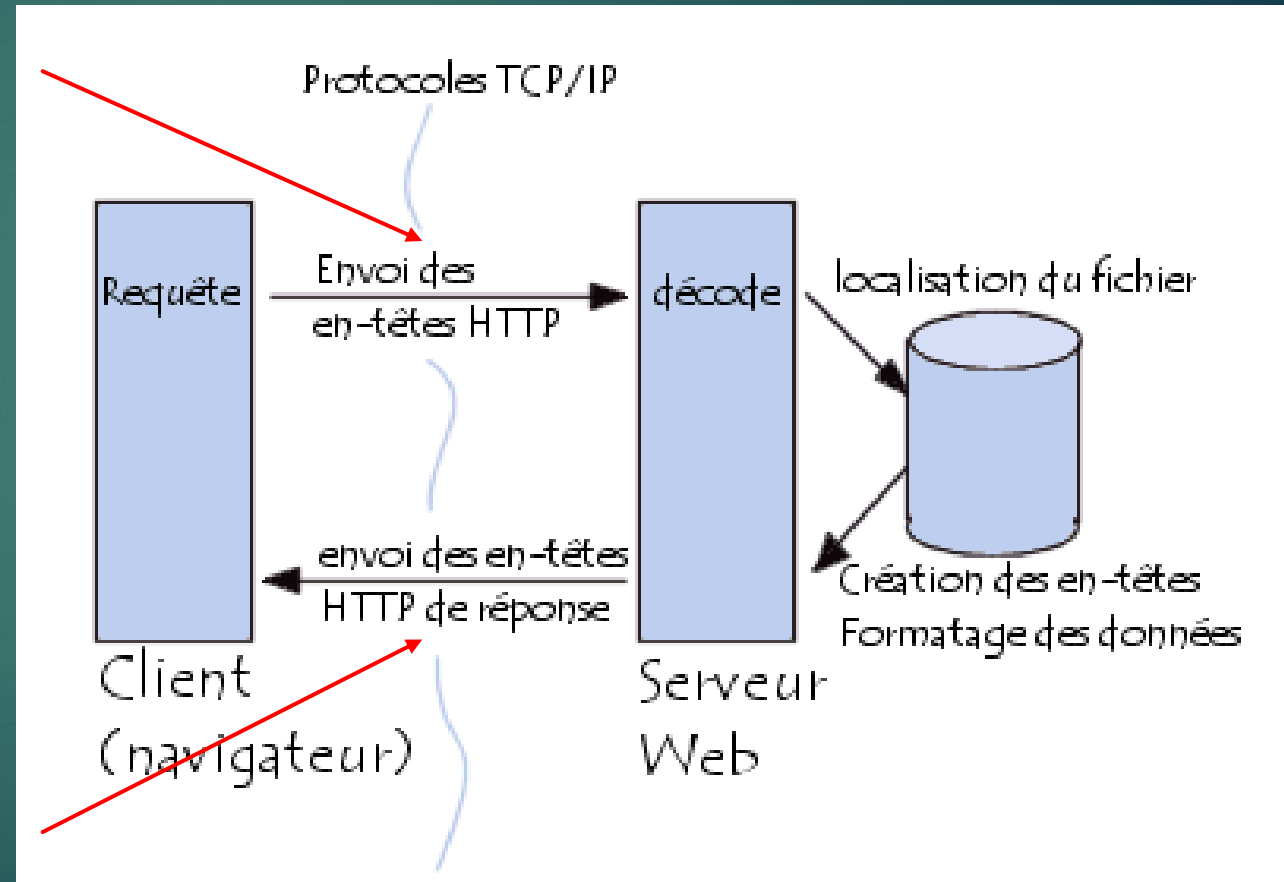
ports

Le protocole HTTP

9

GET /fichier.html HTTP 1.1
Host: www.monsite.fr
User-Agent : Mozilla...

HTTP 1.1 200 OK
...
Server : Apache/2.2.3
Content-Length: <taille>
Content-type: text/html



Avec la commande ss

10

Client

```
manu@tinyManu ~ $ netstat -ntp
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.56.254:55986    192.168.56.101:80      ESTABLISHED
```

Serveur

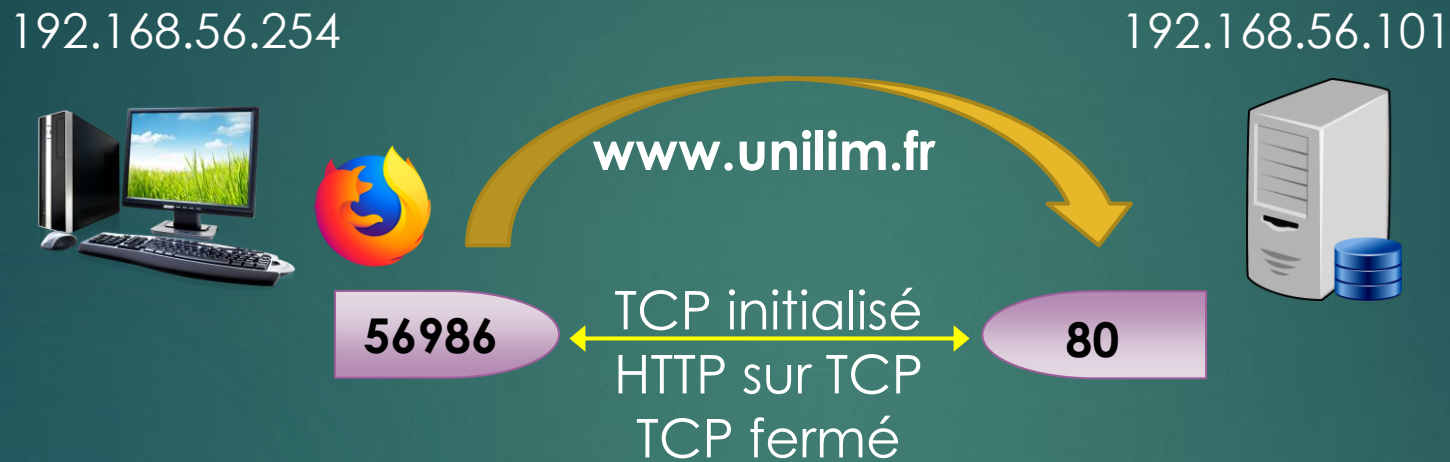
```
root@debian7-cli:/var/www# netstat -ntpa
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      3389/apache2
tcp        0      0 192.168.56.101:80       192.168.56.254:55986    ESTABLISHED 3398/apache2
```

actif

à l'attente

Pour résumer

11



- ▶ 1. Le client cherche la machine ciblée par adresse IP
- ▶ 2. Le service Web est à trouver sur le port 80 (HTTP)
- ▶ 3. La communication HTTP se fait dans l'encapsulation HTTP

HTTP vs HTTPS

12

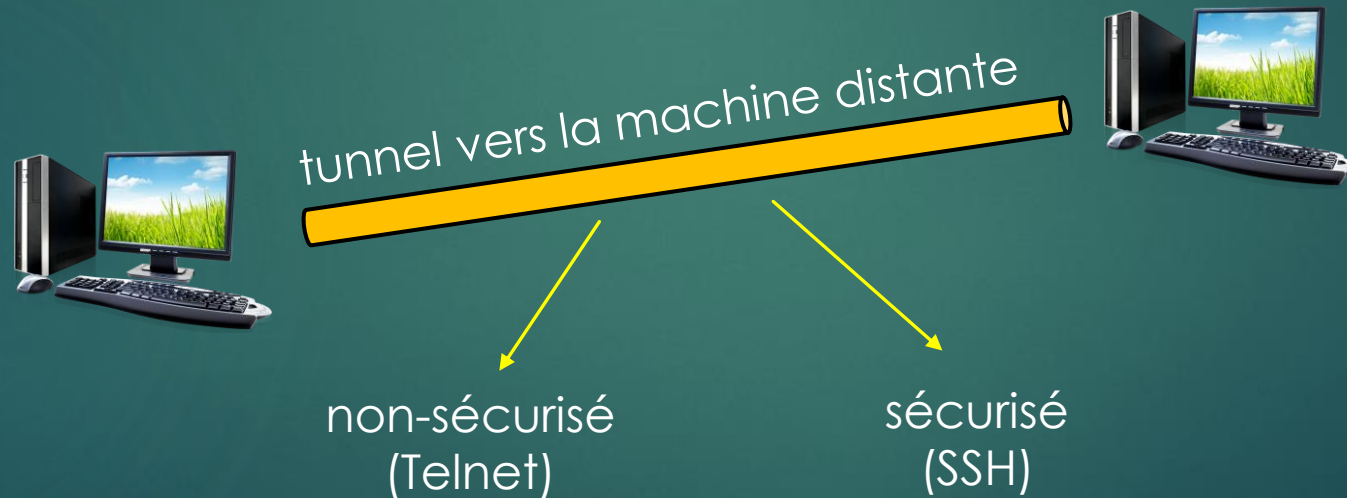
- ▶ Port HTTP : 80 vs. port HTTPS : 443
- ▶ Les communications sur HTTP sont non-sécurisées
 - ▶ Par contre HTTPS sert à donner une sécurité de bout au bout
 - ▶ De plus en plus, les connexions aujourd'hui sont sécurisées
- ▶ HTTPS = HTTP sur TLS (sur TCP)
 - ▶ Une première étape : un échange de messages, standard RFC 5246
 - ▶ A la fin : le client + le serveur partagent des clés de chiffrement /authentification
 - ▶ Deuxième étape : communication HTTP chiffrée + authentifiée

Telnet & SSH

Problématique

14

- ▶ Utilisateur d'une machine veut manipuler une machine à distance
 - ▶ Pour modifier ses fichiers, par exemple
 - ▶ Ou pour contrôler l'état de cette machine (pour un serveur, par exemple)



Telnet

15

- ▶ Protocole qui permet d'émuler un terminal vers une machine ciblée
- ▶ Les commandes tapées dans le terminal s'exécuteront sur l'autre machine
- ▶ Environnement client-serveur
 - ▶ Machine cible = serveur
 - ▶ Machine sur laquelle on se connecte = client
- ▶ Canal non-sécurisé (information susceptible à être interceptée)
 - ▶ Mais accès limité aux personnes fournissant un nom d'utilisateur & un mot de passe

Quelques commandes

16

- Pour se connecter : `telnet <adresse IP de la machine ciblée>`

Comma nde	Description
?	Affiche l'aide
close	Termine la session Telnet
display	Affiche à l'écran les paramètres de la connexion (type de terminal, port)
environ	Permet de définir les variables d'environnement du système d'exploitation
logout	Permet de se déconnecter
mode	Bascule entre les modes de transfert ASCII (transfert d'un fichier en mode texte) et BINARY (transfert d'un fichier en binaire)
open	Permet de lancer une autre connexion à partir de la connexion en cours
quit	Quitte l'application Telnet
set	Modifie les paramètres de la connexion
unset	Charge les paramètres de connexion par défaut

Source : commentcamarche.com

Se connecter à Telnet

17

- Port 23 sur TCP

adresse IP cible

```
manu@tinyManu ~ $ telnet 192.168.56.101
Trying 192.168.56.101...
Connected to 192.168.56.101.
Escape character is '^]'.
Debian GNU/Linux 7
debian7-cli login: user
Password:
Last login: Sun May  8 14:21:15 BST 2016 from 192.168.56.254 on pts/0
Linux debian7-cli 3.2.0-4-amd64 #1 SMP Debian 3.2.68-1+deb7u4 x86_64
user@debian7-cli:~$
```

nom d'utilisateur & mot de passe

Telnet sur netstat

18

Client

```
manu@tinyManu ~ $ netstat -tn | grep 192.168.56.101
tcp        0      0 192.168.56.254:59086 192.168.56.101:23 ESTABLISHED
```

machine ciblée

Serveur

source (client) : port 59086

cible (serveur) : port 23

```
root@debian7-cli:/var/www# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:23               0.0.0.0:*               LISTEN
tcp        0      0 192.168.56.101:23       192.168.56.254:59086   ESTABLISHED
```

Sur Wireshark

19

No.	Time	Source	Destination	Protocol	Info
1	13:22:40.957213851	192.168.56.254	192.168.56.101	TCP	59086 → 23 [SYN] Seq=0
2	13:22:40.957595713	192.168.56.101	192.168.56.254	TCP	23 → 59086 [SYN, ACK] Seq=1
3	13:22:40.957638728	192.168.56.254	192.168.56.101	TCP	59086 → 23 [ACK] Seq=1
4	13:22:40.957729982	192.168.56.254	192.168.56.101	TELNET	Telnet Data ..
5	13:22:40.957848251	192.168.56.101	192.168.56.254	TCP	23 → 59086 [ACK] Seq=1
6	13:22:40.964333575	192.168.56.101	192.168.56.254	TELNET	Telnet Data ..
7	13:22:40.964369751	192.168.56.254	192.168.56.101	TCP	59086 → 23 [ACK] Seq=28
8	13:22:40.964509206	192.168.56.101	192.168.56.254	TELNET	Telnet Data ..
9	13:22:40.964523550	192.168.56.254	192.168.56.101	TCP	59086 → 23 [ACK] Seq=28
10	13:22:40.964903913	192.168.56.254	192.168.56.101	TELNET	Telnet Data ..
11	13:22:40.965355289	192.168.56.101	192.168.56.254	TELNET	Telnet Data ..
12	13:22:40.965414791	192.168.56.254	192.168.56.101	TELNET	Telnet Data ..
13	13:22:40.965592504	192.168.56.101	192.168.56.254	TELNET	Telnet Data ..
14	13:22:40.965653180	192.168.56.254	192.168.56.101	TELNET	Telnet Data ..
15	13:22:40.965739819	192.168.56.101	192.168.56.254	TELNET	Telnet Data ..

TCP initialisé

ACK TCP à part

données envoyées par Telnet

ACK dans message Telnet

données non-chiffrées

Telnet vs. SSH

20

- ▶ La communication sur Telnet n'est pas sécurisée
 - ▶ Si on veut avoir une sécurité de bout-au-bout, il faut utiliser SSH
- ▶ SSH sert à authentifier les deux parties et à établir des clés partagées pour faire du chiffrement et l'authentification des messages
 - ▶ Même rôle que TLS, mais un protocole différent
 - ▶ Port 22 sur TCP

```
manu@tinyManu ~ $ ssh user@192.168.56.101
The authenticity of host '192.168.56.101 (192.168.56.101)' can't be established.
ECDSA key fingerprint is SHA256:NJEV9ZQ1xJNMF/ut67o059+nV1ZxaaNNv5lqrkJE9wY.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.101' (ECDSA) to the list of known hosts.
user@192.168.56.101's password:
Linux debian7-cli 3.2.0-4-amd64 #1 SMP Debian 3.2.68-1+deb7u4 x86_64
Last login: Sun May  8 14:38:49 2016 from 192.168.56.254
user@debian7-cli:~$ pwd
/home/user
user@debian7-cli:~$ exit
logout
Connection to 192.168.56.101 closed.
```


SSH sur Wireshark

21

No.	Time	Source	Destination	Protocol	Info
1	13:41:34.714078811	192.168.56.254	192.168.56.101	TCP	50042 → 22 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK
2	13:41:34.714455738	192.168.56.101	192.168.56.254	TCP	22 → 50042 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MS
3	13:41:34.714494366	192.168.56.254	192.168.56.101	TCP	50042 → 22 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=2
4	13:41:34.714781157	192.168.56.254	192.168.56.101	SSHv2	Client: Protocol (SSH-2.0-OpenSSH_7.1)
5	13:41:34.714893089	192.168.56.101	192.168.56.254	TCP	22 → 50042 [ACK] Seq=1 Ack=22 Win=14480 Len=0 TSval=
6	13:41:34.723801457	192.168.56.101	192.168.56.254	SSHv2	Server: Protocol (SSH-2.0-OpenSSH_6.0p1 Debian-4+deb
7	13:41:34.723895591	192.168.56.254	192.168.56.101	TCP	50042 → 22 [ACK] Seq=22 Ack=40 Win=29312 Len=0 TSval=
8	13:41:34.724723011	192.168.56.254	192.168.56.101	TCP	[TCP segment of a reassembled PDU]
9	13:41:34.724743892	192.168.56.254	192.168.56.101	SSHv2	Client: Key Exchange Init
10	13:41:34.724930390	192.168.56.101	192.168.56.254	TCP	22 → 50042 [ACK] Seq=40 Ack=1870 Win=20272 Len=0 TSv
11	13:41:34.725333963	192.168.56.101	192.168.56.254	SSHv2	Server: Key Exchange Init
12	13:41:34.725646566	192.168.56.254	192.168.56.101	SSHv2	Client: Diffie-Hellman Key Exchange Init
13	13:41:34.727433910	192.168.56.101	192.168.56.254	SSHv2	Server: Diffie-Hellman Key Exchange Reply, New Keys
14	13:41:34.729709417	192.168.56.254	192.168.56.101	SSHv2	Client: New Keys
15	13:41:34.769313075	192.168.56.101	192.168.56.254	TCP	22 → 50042 [ACK] Seq=1336 Ack=1966 Win=20272 Len=0 T
16	13:41:34.769338430	192.168.56.254	192.168.56.101	SSHv2	Client: Encrypted packet (len=40)
17	13:41:34.769416088	192.168.56.101	192.168.56.254	TCP	22 → 50042 [ACK] Seq=1336 Ack=2006 Win=20272 Len=0 T
18	13:41:34.769503984	192.168.56.101	192.168.56.254	SSHv2	Server: Encrypted packet (len=40)
19	13:41:34.769769353	192.168.56.254	192.168.56.101	SSHv2	Client: Encrypted packet (len=56)

TCP initialisé

negotiation
paramètres SSH

établissement
de clés SSH

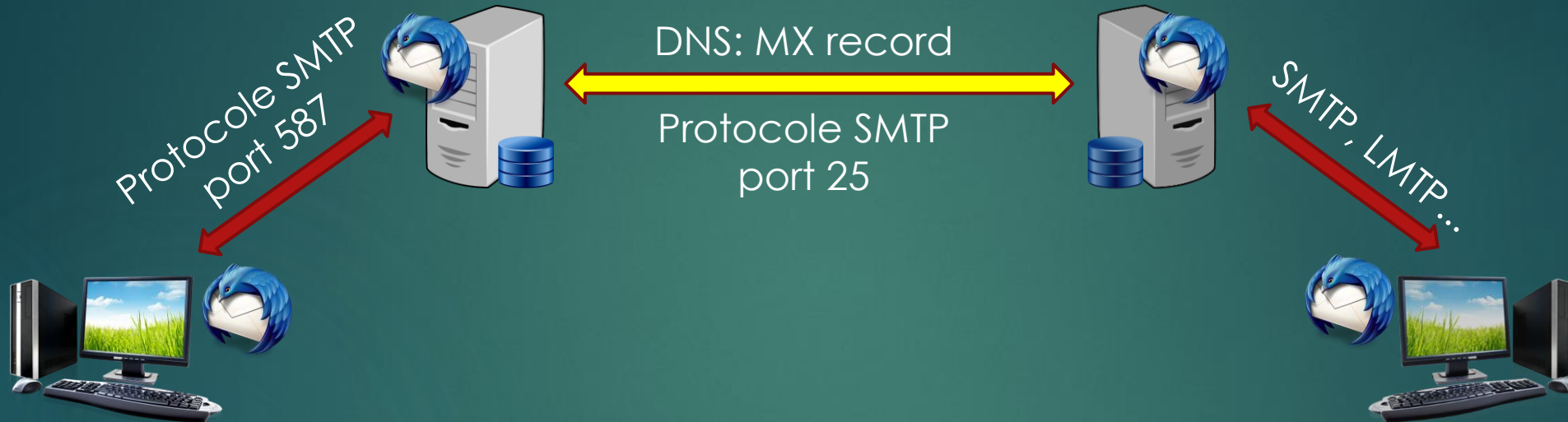
les messages sont chiffrés



SMTP: Simple mail transfer protocol

Les serveurs mail

23



- ▶ Un utilisateur (client mail) consulte ses mails sur un serveur mail
- ▶ Les emails écrites par le client sont stockés par le serveur, et livrés à un autre serveur mail
- ▶ Le recepateur consulte son propre serveur pour retrouver ses messages

SMTP sur Wireshark

24

port SMTP

No.	Time	Source	Destination	Protocol	Info
3	15:05:13.515236244	164.81.6.183	164.81.1.45	TCP	37742 → 25 [SYN] Seq= Win=27320 Len=0
4	15:05:13.578778483	164.81.1.45	164.81.6.183	TCP	25 → 37742 [SYN, ACK] Seq=0 Ack=1 Win=
5	15:05:13.578831129	164.81.6.183	164.81.1.45	TCP	37742 → 25 [ACK] Seq= Ack=1 Win=27392
7	15:05:14.647755281	164.81.1.45	164.81.6.183	SMTP	S: 220 smtp.unilim.fr ESMTP Sendmail 8
8	15:05:14.647807114	164.81.6.183	164.81.1.45	TCP	37742 → 25 [ACK] Seq=1 Ack=82 Win=2739
13	15:05:26.020946763	164.81.6.183	164.81.1.45	SMTP	C: EHLO pc.unilim.fr
14	15:05:26.084085213	164.81.1.45	164.81.6.183	TCP	25 → 37742 [ACK] Seq=82 Ack=19 Win=579
15	15:05:26.086484394	164.81.1.45	164.81.6.183	SMTP	S: 250 smtp.unilim.fr Hello vpn183-per
16	15:05:26.086513591	164.81.6.183	164.81.1.45	TCP	37742 → 25 [ACK] Seq=19 Ack=313 Win=28
17	15:05:30.667462901	164.81.6.183	164.81.1.45	SMTP	C: quit
18	15:05:30.729513789	164.81.1.45	164.81.6.183	SMTP	S: 221 2.0.0 smtp.unilim.fr closing co
19	15:05:30.729562980	164.81.6.183	164.81.1.45	TCP	37742 → 25 [ACK] Seq=24 Ack=358 Win=28
20	15:05:30.729869306	164.81.1.45	164.81.6.183	TCP	25 → 37742 [FIN, ACK] Seq=358 Ack=24 W
21	15:05:30.729935003	164.81.6.183	164.81.1.45	TCP	37742 → 25 [FIN, ACK] Seq=24 Ack=359 W

Client **Serveur**

SMTP -- le protocole

25

- Pour Outlook (ou d'autres clients WebMail), d'autres protocoles sont possibles

```
Stream Content
220 smtp.unilim.fr ESMTP Sendmail 8.13.1/8.13.1; Sun, 8 May 2016 17:04:44 +0200
EHLO pc.unilim.fr
250-smtp.unilim.fr Hello vpn183-personnel.unilim.fr [164.81.6.183], pleased to meet you
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-EXPN
250-VERB
250-8BITMIME
250-SIZE 30000000
250-DSN
250-STARTTLS
250-DELIVERBY
250 HELP
MAIL FROM: <emmanuel.rodès@unilim.fr>
250 2.1.0 <emmanuel.rodès@unilim.fr>... Sender ok
RCPT TO: <rodès01@unilim.fr>
550 5.1.1 <rodès01@unilim.fr>... User unknown
RCPT TO: <emmanuel.rodès@unilim.fr>
250 2.1.5 <emmanuel.rodès@unilim.fr>... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Bonjour,
Ceci est un test !
.
250 2.0.0 u48F4iHN005337 Message accepted for delivery
QUIT
221 2.0.0 smtp.unilim.fr closing connection
```