Cours sécurité des réseaux

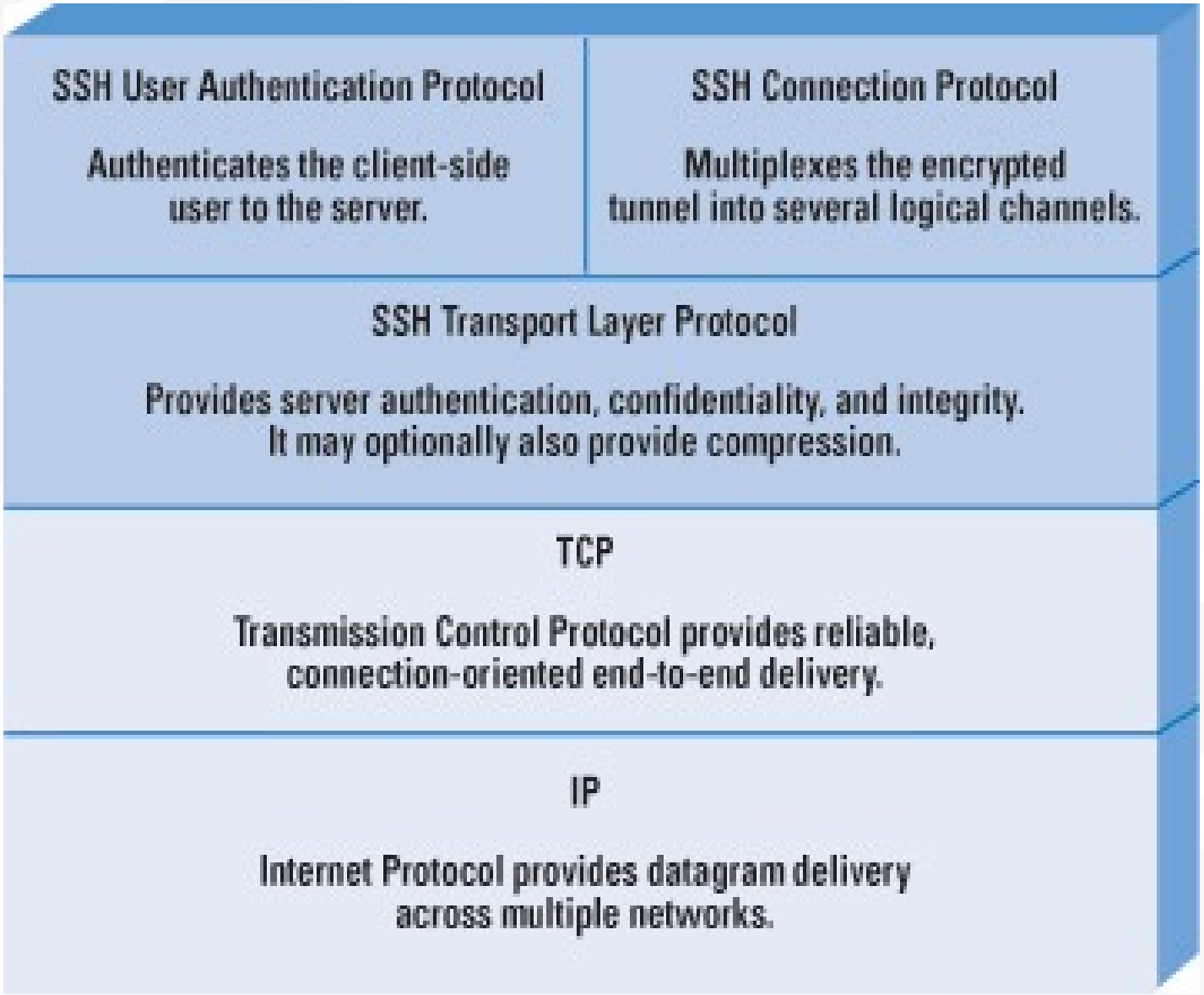
Secure Shell (SSH)

Mohamed Houcine HDHILI
med_elhdhili@yahoo.es

SSH: Présentation

- Permet l'accès sécurisée à une machine distante (routeurs, firewall, serveurs...) pour l'administrer.
- Remplace le service telnet (non sécurisé)
- Dernière version: **SSH2** (IETF RFC 4250 à 4256)
- Implémentation courante sous GNU Linux et BSD: **openSSH**
- **OpenSSH** =suite de logiciels (ssh, sshd, sscp, sftp)
- Exemple de client sous windows: **Putty**

SSH: couches protocolaires



Transport layer protocol (RFC 4253)

- Etape 1: échange des versions des protocoles entre le client et le serveur (après l'établissement de connexion TCP)

SSH-protoversion-softwareversion SP comments CR LF

No.	Time	Source	Destination	Protoc	Length	Info
348	4.058327000	192.168.0.122	192.168.0.192	TCP	74	40972 > ssh [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSva
349	4.058361000	192.168.0.192	192.168.0.122	TCP	74	ssh > 40972 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_
350	4.058538000	192.168.0.122	192.168.0.192	TCP	66	40972 > ssh [ACK] Seq=1 Ack=1 Win=29696 Len=0 TSval=4294956473 TS
351	4.074983000	192.168.0.192	192.168.0.122	SSHv2	109	Server Protocol: SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.8\r
352	4.075232000	192.168.0.122	192.168.0.192	TCP	66	40972 > ssh [ACK] Seq=1 Ack=44 Win=29696 Len=0 TSval=4294956477 TS
353	4.075354000	192.168.0.122	192.168.0.192	SSHv2	98	Client Protocol: SSH-2.0-OpenSSH_6.0p1 Debian-4\r
354	4.075395000	192.168.0.192	192.168.0.122	TCP	66	ssh > 40972 [ACK] Seq=44 Ack=33 Win=29056 Len=0 TSval=1448091 TSe
355	4.075627000	192.168.0.122	192.168.0.192	SSHv2	1338	Client: Key Exchange Init
356	4.075638000	192.168.0.192	192.168.0.122	TCP	66	ssh > 40972 [ACK] Seq=44 Ack=1305 Win=31872 Len=0 TSval=1448091 TS
357	4.075977000	192.168.0.192	192.168.0.122	TCP	1514	[TCP segment of a reassembled PDU]
358	4.075997000	192.168.0.192	192.168.0.122	SSHv2	266	Server: Key Exchange Init

Transport layer protocol (RFC 4253)

- Étape 2: négociation sur les algorithmes (cryptage, MAC, compression) à utiliser dans les deux sens.

Key exchange init

```
10 0.027309000    192.168.1.4    192.168.1.3    SSHv2    726 Client: Key Exchange Init

▼SSH Protocol
 ▼SSH Version 2 (encryption:aes256-ctr mac:hmac-sha2-256 compression:none)
   Packet Length: 668
   Padding Length: 10
  ▼Key Exchange
    Message Code: Key Exchange Init (20)
   ▼Algorithms
     Cookie: 2654b4596d8ef237a9f375acffeb36cf
     kex_algorithms length: 154
     kex_algorithms string: diffie-hellman-group-exchange-sha256,diffie-hellman-group-exchange-sha1,diffie-hellman-group14-sha1,diffie-hellman-group1-sha1,rsa2
     server_host_key_algorithms length: 15
     server_host_key_algorithms string: ssh-rsa,ssh-dss
     encryption_algorithms_client_to_server length: 159
     encryption_algorithms_client_to_server string: aes256-ctr,aes256-cbc,rijndael-cbc@lysator.liu.se,aes192-ctr,aes192-cbc,aes128-ctr,aes128-cbc,blowfish-ctr,b
     encryption_algorithms_server_to_client length: 159
     encryption_algorithms_server_to_client string: aes256-ctr,aes256-cbc,rijndael-cbc@lysator.liu.se,aes192-ctr,aes192-cbc,aes128-ctr,aes128-cbc,blowfish-ctr,b
     mac_algorithms_client_to_server length: 45
     mac_algorithms_client_to_server string: hmac-sha2-256,hmac-sha1,hmac-sha1-96,hmac-md5
     mac_algorithms_server_to_client length: 45
     mac_algorithms_server_to_client string: hmac-sha2-256,hmac-sha1,hmac-sha1-96,hmac-md5
     compression_algorithms_client_to_server length: 9
     compression_algorithms_client_to_server string: none,zlib
     compression_algorithms_server_to_client length: 9
     compression_algorithms_server_to_client string: none,zlib
```

Transport layer protocol (RFC 4253)

- Étape 3: Mise en place d'une clé symétrique K entre le client et le serveur
  - D'une manière sécurisé ==> protocole de Diffie Hellman (basé sur le problème du logarithme discret ou les courbes elliptiques)
  - Le client initie le protocole de DH

```
 10 0.027309000   192.168.1.4    192.168.1.3    SSHv2       726 Client: Key Exchange Init
 11 0.027366000   192.168.1.3    192.168.1.4    SSHv2       242 Server: Key Exchange Init
 13 0.030725000   192.168.1.4    192.168.1.3    SSHv2        78 Client: Diffie-Hellman GEX Request
 14 0.036291000   192.168.1.3    192.168.1.4    SSHv2       590 Server: Diffie-Hellman Key Exchange Reply
 15 0.190190000   192.168.1.4    192.168.1.3    SSHv2       582 Client: Diffie-Hellman GEX Init
 16 0.213926000   192.168.1.3    192.168.1.4    SSHv2      1158 Server: Diffie-Hellman GEX Reply

▶Frame 13: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface 0
▶Ethernet II, Src: LiteonTe_b4:f5:bf (24:fd:52:b4:f5:bf), Dst: LiteonTe_b4:f1:19 (24:fd:52:b4:f1:19)
▶Internet Protocol Version 4, Src: 192.168.1.4 (192.168.1.4), Dst: 192.168.1.3 (192.168.1.3)
▶Transmission Control Protocol, Src Port: 50025 (50025), Dst Port: ssh (22), Seq: 701, Ack: 1692, Len: 24
▼SSH Protocol
 ▼SSH Version 2 (encryption:aes256-ctr mac:hmac-sha2-256 compression:none)
   Packet Length: 20
   Padding Length: 6
  ▼Key Exchange
    Message Code: Diffie-Hellman GEX Request (34)
    DH GEX Min: 00000400
    DH GEX Numbers of Bits: 00001000
    DH GEX Max: 00002000
    Padding String: 4845a3f22f0d
```

Transport layer protocol (RFC 4253)

- Étape 3: Mise en place d'une clé symétrique K entre le client et le serveur
  - Le serveur envoi un grand nombre premier P et un générateur G

10	0.027309000	192.168.1.4	192.168.1.3	SSHv2	726	Client: Key Exchange Init
11	0.027366000	192.168.1.3	192.168.1.4	SSHv2	242	Server: Key Exchange Init
13	0.030725000	192.168.1.4	192.168.1.3	SSHv2	78	Client: Diffie-Hellman GEX Request
14	0.036291000	192.168.1.3	192.168.1.4	SSHv2	590	Server: Diffie-Hellman Key Exchange Reply
15	0.190190000	192.168.1.4	192.168.1.3	SSHv2	582	Client: Diffie-Hellman GEX Init
16	0.213926000	192.168.1.3	192.168.1.4	SSHv2	1158	Server: Diffie-Hellman GEX Reply

```
▸Frame 14: 590 bytes on wire (4720 bits), 590 bytes captured (4720 bits) on interface 0
▸Ethernet II, Src: LiteonTe_b4:f1:19 (24:fd:52:b4:f1:19), Dst: LiteonTe_b4:f5:bf (24:fd:52:b4:f5:bf)
▸Internet Protocol Version 4, Src: 192.168.1.3 (192.168.1.3), Dst: 192.168.1.4 (192.168.1.4)
▸Transmission Control Protocol, Src Port: ssh (22), Dst Port: 50025 (50025), Seq: 1692, Ack: 725, Len: 536
▾SSH Protocol
 ▾SSH Version 2 (encryption:aes256-ctr mac:hmac-sha2-256 compression:none)
   Packet Length: 532
   Padding Length: 8
  ▾Key Exchange
    Message Code: Diffie-Hellman Key Exchange Reply (31)
    Multi Precision Integer Length: 513
    DH modulus (P): 00ef07b0f39662dc8600224e46ab8be8cb72e552d52e8801...
    Multi Precision Integer Length: 1
    DH base (G): 05
    Padding String: 0000000000000000
```

Transport layer protocol (RFC 4253)

- Étape 3: Mise en place d'une clé symétrique K entre le client et le serveur
  - Le client choisit un nombre aléatoire C puis envoi au client $e=G^{C} \bmod P$

10	0.027309000	192.168.1.4	192.168.1.3	SSHv2	726	Client: Key Exchange Init
11	0.027366000	192.168.1.3	192.168.1.4	SSHv2	242	Server: Key Exchange Init
13	0.030725000	192.168.1.4	192.168.1.3	SSHv2	78	Client: Diffie-Hellman GEX Request
14	0.036291000	192.168.1.3	192.168.1.4	SSHv2	590	Server: Diffie-Hellman Key Exchange Reply
15	0.190190000	192.168.1.4	192.168.1.3	SSHv2	582	Client: Diffie-Hellman GEX Init
16	0.213926000	192.168.1.3	192.168.1.4	SSHv2	1158	Server: Diffie-Hellman GEX Reply

```
▸Frame 15: 582 bytes on wire (4656 bits), 582 bytes captured (4656 bits) on interface 0
▸Ethernet II, Src: LiteonTe_b4:f5:bf (24:fd:52:b4:f5:bf), Dst: LiteonTe_b4:f1:19 (24:fd:52:b4:f1:19)
▸Internet Protocol Version 4, Src: 192.168.1.4 (192.168.1.4), Dst: 192.168.1.3 (192.168.1.3)
▸Transmission Control Protocol, Src Port: 50025 (50025), Dst Port: ssh (22), Seq: 725, Ack: 2228, Len: 528
▾SSH Protocol
 ▾SSH Version 2 (encryption:aes256-ctr mac:hmac-sha2-256 compression:none)
   Packet Length: 524
   Padding Length: 5
  ▾Key Exchange
    Message Code: Diffie-Hellman GEX Init (32)
    Multi Precision Integer Length: 513
    DH client e: 00c9947c72d762ef32054ca6b8bce76dd22c801d8354c8b9...
    Padding String: 73662ff322
```

Transport layer protocol (RFC 4253)

- Étape 3: Mise en place d'une clé symétrique **K** entre le client et le serveur
  - Le serveur choisit un nombre aléatoire S puis calcule:
    - la clé partagé K= e^Smod P, génère les 6 clés (idem SSL)
    - f=G^Smod P
    - h=hash(Vc || Vs || client_Algo_list || server_Algo_list || KpubS || e ||f || K). Hash est déjà négocié. Vc et Vs: versions du client et du serveur
    - s=signature$_{KprivS}$(h) : signature calculé par le serveur
  - Le serveur envoi (KpubS, f, s)

```
16 0.213926000   192.168.1.3     192.168.1.4     SSHv2      1158 Server: Diffie-Hellman GEX Reply

▾SSH Protocol
 ▾SSH Version 2 (encryption:aes256-ctr mac:hmac-sha2-256 compression:none)
   Packet Length: 1084
   Padding Length: 8
  ▾Key Exchange
    Message Code: Diffie-Hellman GEX Reply (33)
    KEX DH host key length: 279
    KEX DH host key: 000000077373682d727361000000030100010000010100bb...
    Multi Precision Integer Length: 512
    DH server f: 79e7880373eda8c40a02dc478b88fba847f8472cd29f4fc2...
    KEX DH H signature length: 271
    KEX DH H signature: 000000077373682d72736100000100af238c54290e8c3179...
    Padding String: 0000000000000000
    MAC: 0000000c0a150000000000000000000000
```

Transport layer protocol (RFC 4253)

- Étape 4: vérifier l'authenticité du serveur puis échangé de trafic crypté et hashé
  - **Le client vérifie l'authenticité de la clé publique KpubS à partir de son empreinte donné en première connexion:**

```
myname@myname-K46CB:~$ ssh etudiant@192.168.1.3
The authenticity of host '192.168.1.3 (192.168.1.3)' can't be established.
ECDSA key fingerprint is 47:e3:7d:10:43:4c:4f:96:ec:38:74:0f:eb:91:a8:2d.
Are you sure you want to continue connecting (yes/no)?
```

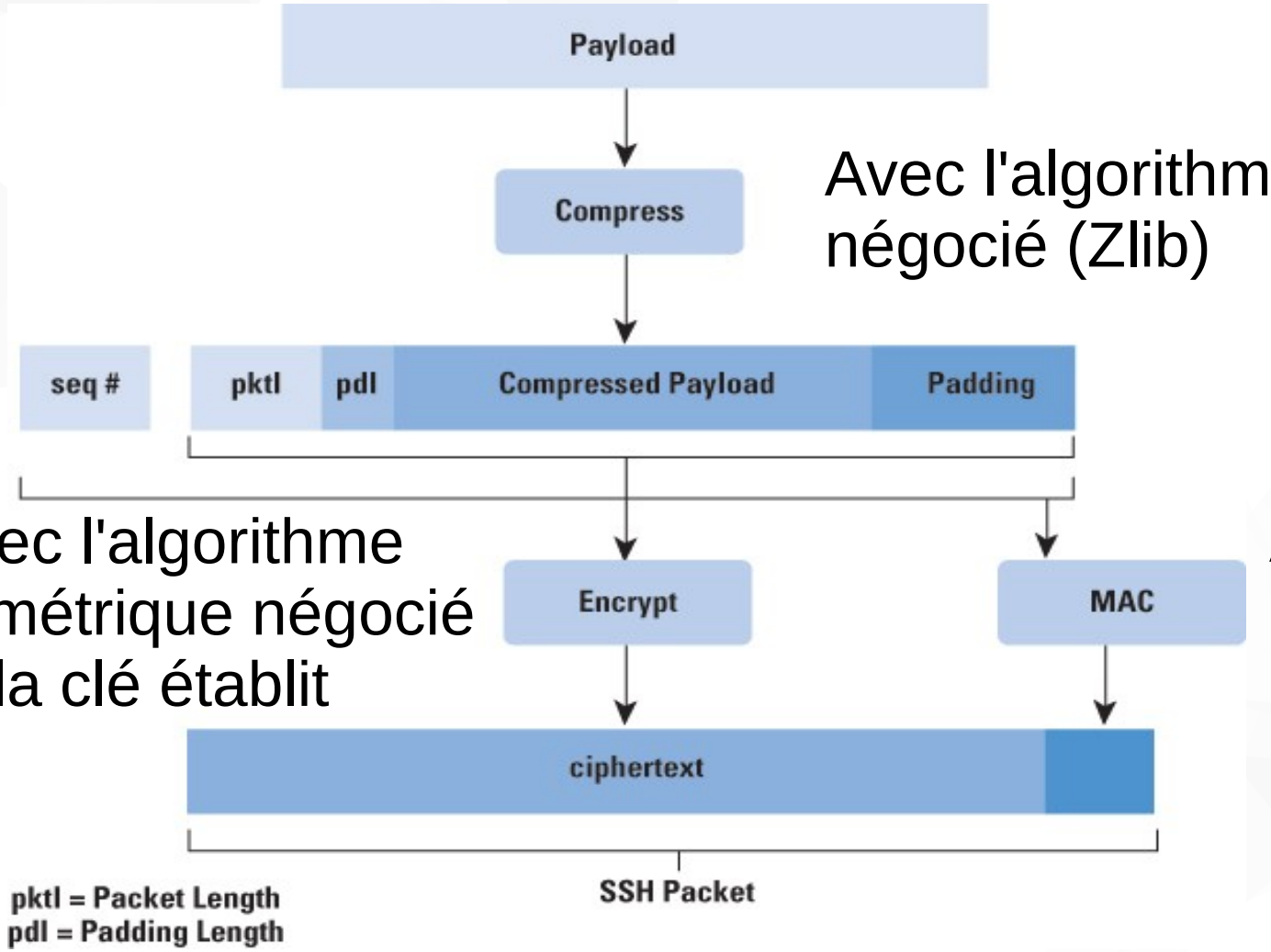
  - Après validation (yes), le serveur est reconnu et sa clé est stockée dans une base locale appelée “known hosts”.
  - Cryptage et hashage du trafic

```
032429000 192.168.1.3    192.168.1.4    SSHv2      902 Encrypted response packet len=848

▸Frame 65: 902 bytes on wire (7216 bits), 902 bytes captured (7216 bits) on interface 0
▸Ethernet II, Src: LiteonTe_b4:f1:19 (24:fd:52:b4:f1:19), Dst: LiteonTe_b4:f5:bf (24:fd:52:b4:f5:bf)
▸Internet Protocol Version 4, Src: 192.168.1.3 (192.168.1.3), Dst: 192.168.1.4 (192.168.1.4)
▸Transmission Control Protocol, Src Port: ssh (22), Dst Port: 50025 (50025), Seq: 3748, Ack: 1973, Len: 848
▾SSH Protocol
 ▾SSH Version 2 (encryption:aes256-ctr mac:hmac-sha2-256 compression:none)
   Encrypted Packet: f7fc8092b59a5f83315bc2f29997dc9d8599aac896fb39a7...
   MAC: 3e4811e0bd1c7d2218d4e4547a8990be8df46eba7c1f9adc...
```

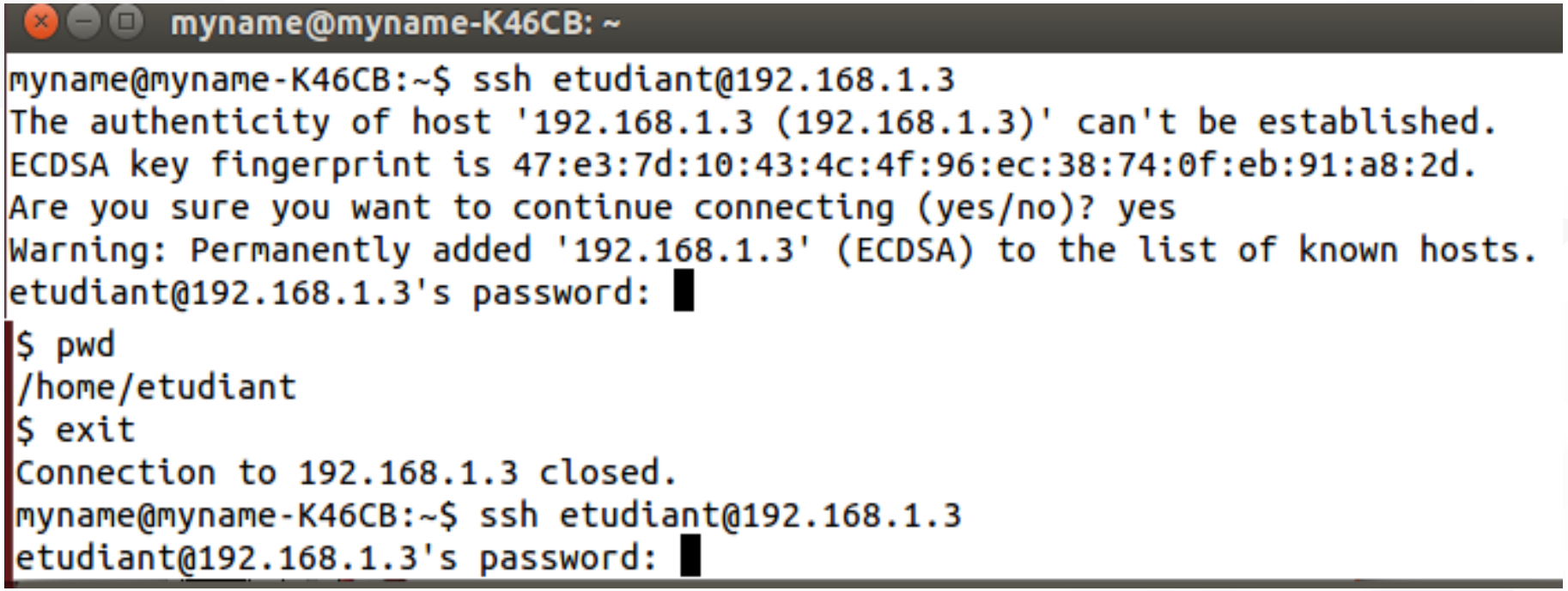
Transport Layer Protocol; Echange sécurisé de données

- Préparation d'un payload (message) à envoyer par le client ou le serveur



Authentification du client

- **Par mot de passe**
  - Le client fournit un mot de passe (de son compte sur le serveur) qui sera envoyé dans un paquet chiffré.
  - Il faut accepter le “fingerprint” pour la première connexion



Authentification du client

- **Par clé privée**
  - Phase de distribution de clés:
    - Le client génère une paire de clés KpubC/KprivC et **envoit KpubC au serveur** (par exemple, en utilisant scp qui permet de copier KpubC sur le serveur en utilisant SSH)
  - Phase d'authentification: basé sur le challenge-response
    - le client chiffre une information connu par le serveur (par exemple sa clé publique) par sa clé privée.
    - Si le challenge réussie, le client est authentifié

A faire…

- … TP sur SSH