ING4 SI01

Network Security

Network Virtualization

Lab 2

Author(s):
Gabriel PADIS
Anastasia DUCHESNE

Teacher(s):
Mr. Nahle

Paris - Semester 7 - March 5, 2019

Contents

1 Network set-up **2**
1.1 Example of PC configuration . . . 2
1.2 Ping tables . . . 2
1.3 Failed pings . . . 2
1.4 Adding routes . . . 2
1.5 PC3-Host set-up . . . 3

2 DHCP server **4**
2.1 Router and PC interfaces configuration . . . 4
2.2 DHCP Configuration . . . 4
2.3 Linux command lines . . . 5
2.4 Ping . . . 5

3 HTTP Server **6**
3.1 Set-up . . . 6
3.2 Copy ssh . . . 6
3.3 HTTP traffic . . . 7

4 FTP Server **8**
4.1 TCP Understanding . . . 8
4.1.1 Connection understanding . . . 8
4.1.2 Sequence numbers sent . . . 10
4.1.3 Sequence numbers reception . . . 11
4.1.4 Retransmissions . . . 11
4.1.5 Buffers impact . . . 12
4.1.6 Throughput graph . . . 13
4.2 FTP Understanding . . . 13
4.2.1 Connections for one file . . . 13
4.2.2 Connections for three files . . . 13
4.2.3 Read content . . . 14
4.2.4 FTP sequence diagram . . . 14

1 Network set-up

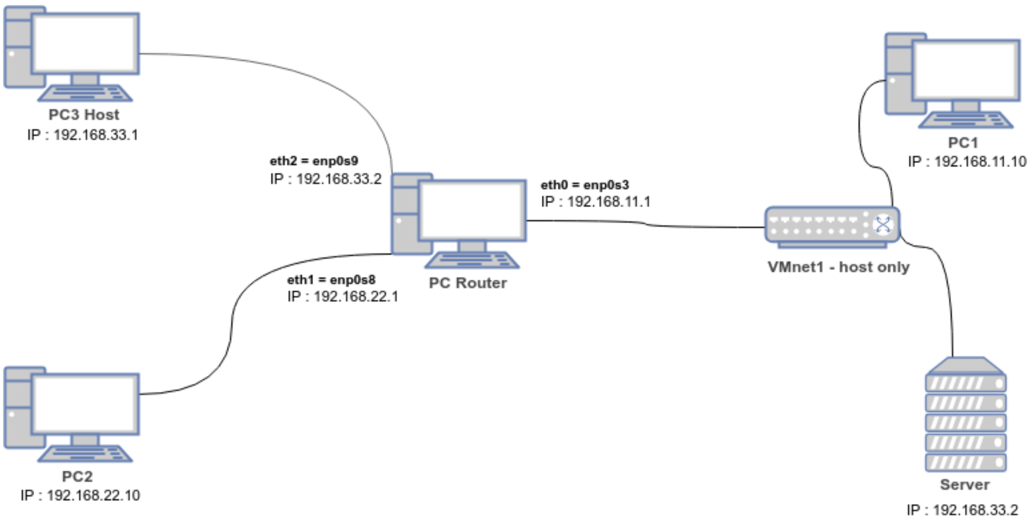


Figure 1: Network setup

1.1 Example of PC configuration

```
iface enp0s3 inet static
address 192.168.11.10
netmask 255.255.255.0
gateway 192.168.11.1
broadcast 192.168.11.255
```

Figure 2: PC1 configuration

1.2 Ping tables

Ping	**Status** (OK/fail)
PC1 → PC-Router	Ok
PC2 → PC-Router	Ok
PC3-Host → PC-Router	Ok
Server → PC2	Fail
PC3-Host → PC1	Fail

1.3 Failed pings

The ping failed because PC2 and the server or PC3-Host and PC1 are not on the same network and there is no route configured. The solution is adding routes to the router so it can reroute the packets to the different networks and/or PCs.

1.4 Adding routes

The basic command to add in the configuration file **/etc/network/interfaces** is :

```
up route add -net <subnet address> netmask <subnet mask> gw <subnet gateway>
```

```
auto enp0s3
iface enp0s3 inet static
address 192.168.11.1
netmask 255.255.255.0
up route add -net 192.168.22.0 netmask 255.255.255.0 gw 192.168.22.1
up route add -net 192.168.33.0 netmask 255.255.255.0 gw 192.168.33.2

auto enp0s8
allow-hotplug enp0s8
iface enp0s8 inet static
address 192.168.22.1
netmask 255.255.255.0
up route add -net 192.168.33.0 netmask 255.255.255.0 gw 192.168.33.2
up route add -net 192.168.11.0 netmask 255.255.255.0 gw 192.168.11.1

auto enp0s9
allow-hotplug enp0s9
iface enp0s9 inet static
address 192.168.33.2
netmask 255.255.255.0
gateway 192.168.33.2
up route add -net 192.168.11.0 netmask 255.255.255.0 gw 192.168.11.1
up route add -net 192.168.22.0 netmask 255.255.255.0 gw 192.168.22.1
root@debian:/home/debian#
```

Figure 3: PC Router routes configuration

The configuration file looks this way at the end of the changes :
The 4 PCs can ping each other.
One problem remains : we can not ping PC3-Host.

1.5 PC3-Host set-up

PC3-Host subnet did not have any default gateway on Windows. The default gateway indicates were the packets should go by default. So we had to affect one to it, for the packets to go to the router.

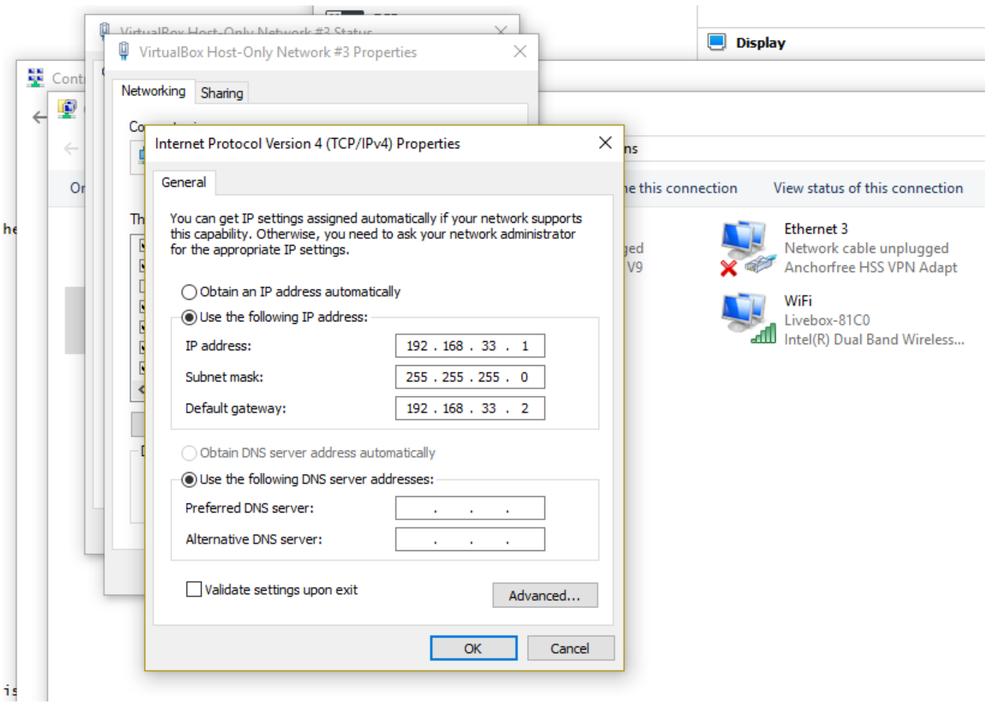
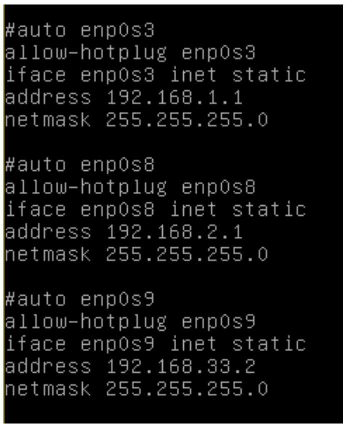


Figure 4: PC3-Host configuration

2 DHCP server

2.1 Router and PC interfaces configuration



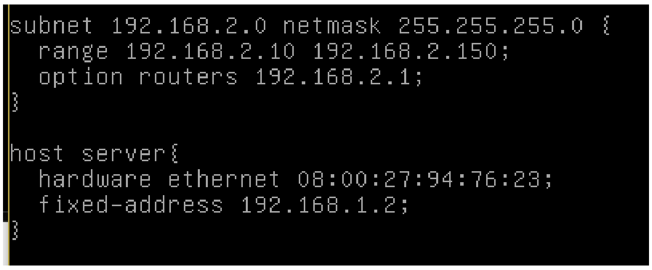
```
#auto enp0s3
allow-hotplug enp0s3
iface enp0s3 inet static
address 192.168.1.1
netmask 255.255.255.0

#auto enp0s8
allow-hotplug enp0s8
iface enp0s8 inet static
address 192.168.2.1
netmask 255.255.255.0

#auto enp0s9
allow-hotplug enp0s9
iface enp0s9 inet static
address 192.168.33.2
netmask 255.255.255.0
```

Figure 5: PC Router interfaces configuration file

In the file **/etc/network/interfaces** of PC Router, we deleted the lines related to the routes, since it is now done automatically thanks to the DHCP server. We also changed the IP addresses of the interfaces enp0s3 (which corresponds to VMNet1) and enp0s8 (VMNet2).

On the PC1 and PC2, we changed **iface enp0s3 inet static** to **iface enp0s3 inet dhcp**, since the interfaces configuration is now done by the dhcp and not manually. We also deleted the lines corresponding the the IP address and netmask in these files.

2.2 DHCP Configuration



```
subnet 192.168.2.0 netmask 255.255.255.0 {
  range 192.168.2.10 192.168.2.150;
  option routers 192.168.2.1;
}

host server{
  hardware ethernet 08:00:27:94:76:23;
  fixed-address 192.168.1.2;
}
```

Figure 6: DHCP Configuration

To create our subnets VMNet1 and VMNet2, we add the following lines to the **/etc/d-hcp/dhcpd.conf** file :

```
subnet <subnet address> netmask <subnet mask> {
    range <begin of range> <end of range>;
    option routers <gateway attributed to the router>;
}
```

The first line indicates the network address and network mask. For example, you can see on the screenshot above that we gave to VMNet2 network the address 192.168.2.0/24.

Range indicates the first and the last IP addresses of the range of addresses that are available on our network. This are the addresses that the DHCP can allocate on the given network. In our case, the available addresses range goes from the 10th to the 150th. The first ten addresses are left for the system.

Option routers attributes the given in parameters IP address on the subnet to the gateway. In our case, we gave the first available address which means 192.168.1.1 and 192.168.2.1 on VMNet1 and VMNet2 respectively.

To reserve a fixed address for a client on our Network through a DHCP server, we add the following lines :

```
host <name of the client> {
    hardware ethernet <MAC address of the client>;
    fixed-address <IP address reserved for the client>;
}
```

Our client is the PC Server. We named it just server. The PC Server virtual machine MAC address is : 08:00:27:94:76:23. The address we reserved for the server on our VMNet1 network is 192.168.1.2.

Finally, we add the following lines in the file **/etc/default/isc-dhcp-server** : We select

Figure 7: DHCP Configuration

the interfaces that will be configured on our network, and to which we want our dhcp server to listen.

2.3 Linux command lines

The Linux commands in order of use :

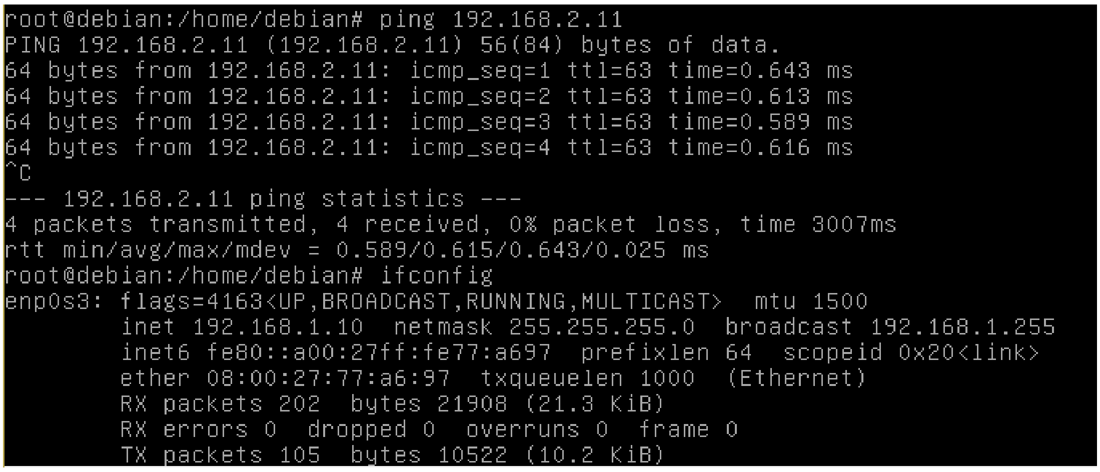
vi /etc/dhcp/dhcpd.conf and **vi /etc/default/isc-dhcp-server** : use vi to modify the configuration files

systemctl restart networking : restart networking for the latest version of Ubuntu server.

systemctl restart isc-dhcp-server or **/etc/init.d/isc-dhcp-server restart** *(script based command)* : restart the dhcp server.

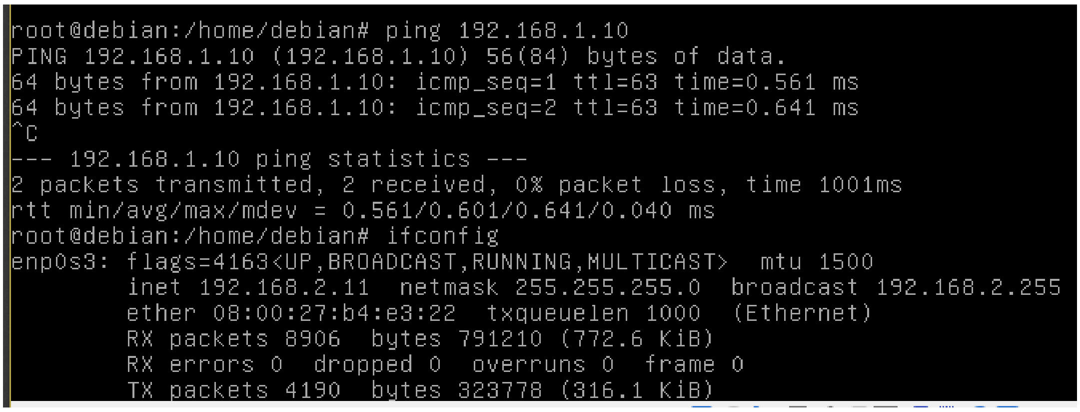
ifup -a : brings all interfaces up. This updates the configuration of the interfaces.

2.4 Ping



```
root@debian:/home/debian# ping 192.168.2.11
PING 192.168.2.11 (192.168.2.11) 56(84) bytes of data.
64 bytes from 192.168.2.11: icmp_seq=1 ttl=63 time=0.643 ms
64 bytes from 192.168.2.11: icmp_seq=2 ttl=63 time=0.613 ms
64 bytes from 192.168.2.11: icmp_seq=3 ttl=63 time=0.589 ms
64 bytes from 192.168.2.11: icmp_seq=4 ttl=63 time=0.616 ms
^C
--- 192.168.2.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 0.589/0.615/0.643/0.025 ms
root@debian:/home/debian# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a00:27ff:fe77:a697  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:77:a6:97  txqueuelen 1000  (Ethernet)
        RX packets 202  bytes 21908 (21.3 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 105  bytes 10522 (10.2 KiB)
```

Figure 8: Pinging PC2 from PC1



```
root@debian:/home/debian# ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=63 time=0.561 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=63 time=0.641 ms
^C
--- 192.168.1.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.561/0.601/0.641/0.040 ms
root@debian:/home/debian# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.11  netmask 255.255.255.0  broadcast 192.168.2.255
        ether 08:00:27:b4:e3:22  txqueuelen 1000  (Ethernet)
        RX packets 8906  bytes 791210 (772.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4190  bytes 323778 (316.1 KiB)
```

Figure 9: Pinging PC1 from PC2

3 HTTP Server

3.1 Set-up

Before we do anything with the ssh we need to configure it on the receiving machine. Indeed it needs to have a password set-up to access it and on Debian root login is disabled by default.
We went to the file : **/etc/ssh/shhd_config** and changed :
PermitRootLogin without-password to **PermitRootLogin yes**
Then we restarted the service :

```
/etc/init.d/ssh restart
```

From there on we can connect to the distant machine as root via ssh.

3.2 Copy ssh

Since we are connected on the server we are sending the file from, we have to use the following command line :

```
scp <src directory path> user@dest server:<dest directory path>
```

scp enables a secure files transfer through SSH between servers. It has two requirements :

- SSH access on the remote machines
- the SSH user needs access to the files or directories desired.

3.3 HTTP traffic

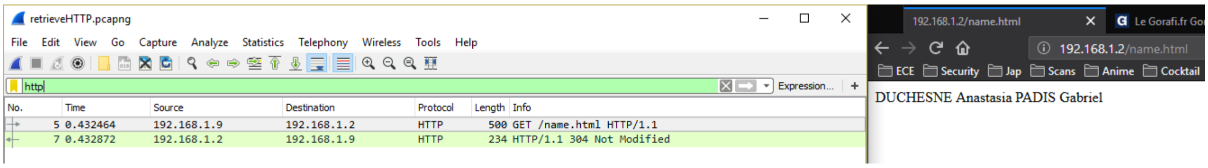


Figure 10: Retrieving HTTP

	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.9	239.255.255.250	SSDP	143	M-SEARCH * HTTP/1.1
2	0.432129	192.168.1.9	192.168.1.2	TCP	66	61811 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
3	0.432328	192.168.1.2	192.168.1.9	TCP	66	80 → 61811 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=32
4	0.432379	192.168.1.9	192.168.1.2	TCP	54	61811 → 80 [ACK] Seq=1 Ack=1 Win=525568 Len=0
5	0.432464	192.168.1.9	192.168.1.2	HTTP	500	GET /name.html HTTP/1.1
6	0.432589	192.168.1.2	192.168.1.9	TCP	60	80 → 61811 [ACK] Seq=1 Ack=447 Win=30272 Len=0

Figure 11: Wireshark capture

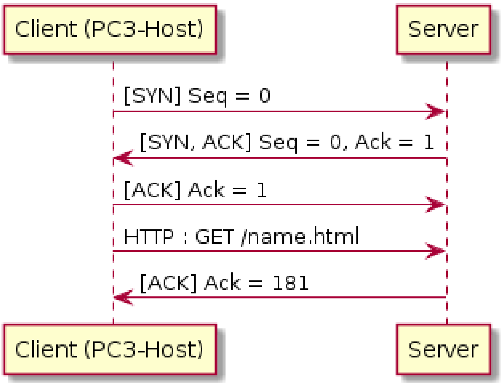


Figure 12: HTTP protocol sequence diagram

In order to retrieve a web page from our server on our PC3-Host, we first have to establish a connection, made of 3 steps :

- The client sends a SYN segment to the Server
- The server answers with a SYN, ACK segment
- the client confirms with a ACK segment

Once the connection is established, the client gets the web page (/name.html in this example)

4 FTP Server

4.1 TCP Understanding

4.1.1 Connection understanding

- Segments for connection establishment As we can see in the documents below there is a connection creation to transmit the file once it has be en requested by the source (192.168.1.9) which is the host to PC1 (192.168.1.10). It has 3 steps that goes as follows :
  PC1 send the SYN segment. It has Seq set at 0
  Host respond with SYN & ACK with Seq still at 0 and Ack set to 1
  PC1 send ACK with Seq at 1 and Ack at 1
  From here on PC1 start to transmit the data.

- Segments for connection release For the connection release the PC1 and the Host needs to make sure that everything has been transmitted. It has 4 steps that goes as follows :
  PC1 has finished to transmit so it sends FIN & ACK with Seq set to X and Ack to 1
  Host acknowledge that the transfer is finished from PC1, respond with ACK set to Y = X + 1
  Host has finished receiving so send it send FIN & ACK with Seq set to 1 and Ack set Y
  PC1 acknowledge that the transfert is finished from Host, respond with Ack set to Y + 1

27	9.979845	192.168.1.10	192.168.1.9	FTP	105	Response: 227 Entering Passive Mode (192,168,1,10,227,…
28	9.980160	192.168.1.9	192.168.1.10	FTP	75	Request: RETR monfichier.txt
29	9.980402	192.168.1.9	192.168.1.10	TCP	66	61829 → 58309 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=…
30	9.980553	192.168.1.10	192.168.1.9	TCP	66	58309 → 61829 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 M…
31	9.980590	192.168.1.9	192.168.1.10	TCP	54	61829 → 58309 [ACK] Seq=1 Ack=1 Win=4194304 Len=0
32	9.980813	192.168.1.10	192.168.1.9	FTP	126	Response: 150 Opening BINARY mode data connection for …
33	9.980912	192.168.1.10	192.168.1.9	FTP-DATA	83	FTP Data: 29 bytes (PASV) (RETR monfichier.txt)
34	9.980965	192.168.1.10	192.168.1.9	TCP	60	58309 → 61829 [FIN, ACK] Seq=30 Ack=1 Win=29216 Len=0
35	9.980982	192.168.1.9	192.168.1.10	TCP	54	61829 → 58309 [ACK] Seq=1 Ack=31 Win=4194176 Len=0
36	9.981097	192.168.1.9	192.168.1.10	TCP	54	61829 → 58309 [FIN, ACK] Seq=1 Ack=31 Win=4194176 Len=0
37	9.981194	192.168.1.10	192.168.1.9	TCP	60	58309 → 61829 [ACK] Seq=31 Ack=2 Win=29216 Len=0
38	9.981312	192.168.1.10	192.168.1.9	FTP	78	Response: 226 Transfer complete.

Figure 13: Connection informations

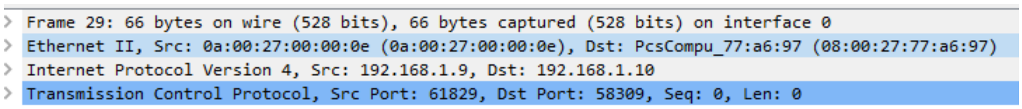


Figure 14: Informations on Source and Host

- IP addresses :
  - Source (HOST) : 192.168.1.9
  - Destination (PC1) :192.168.1.10
- MAC addresses :
  - Source : 0a:00:27:00:00:0e
  - Destination : 00:00:27:77:a6:97
- Port numbers
  - Source : 61829
  - Destination : 58309
- TCP sequence diagram

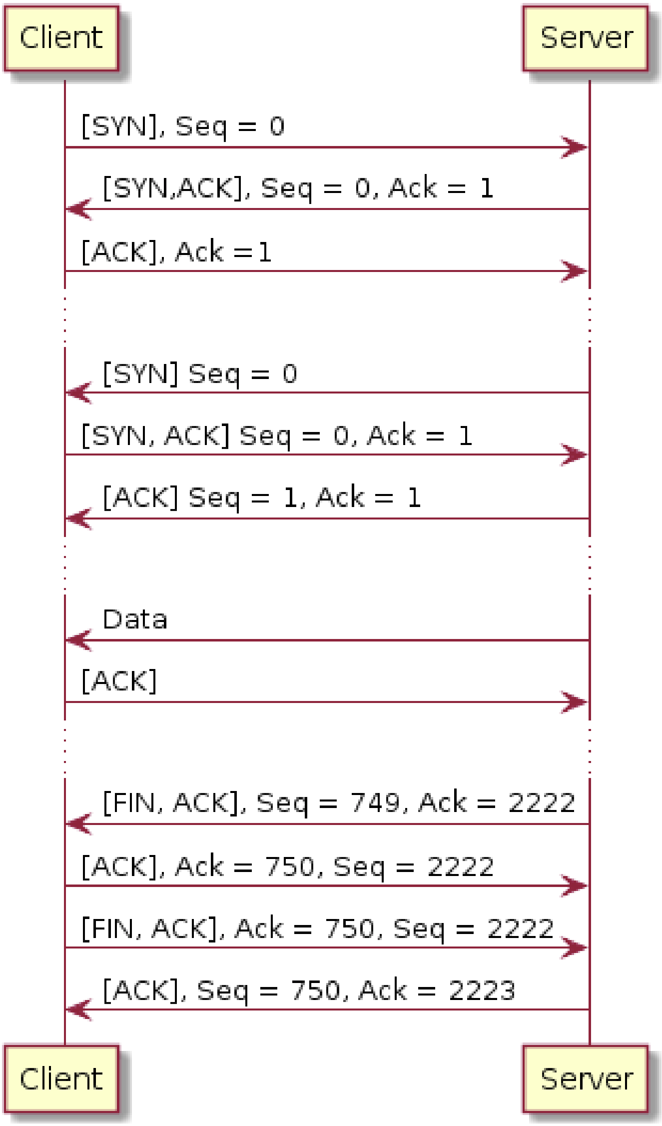


Figure 15: TCP Sequence diagram

4.1.2 Sequence numbers sent

44	3.016479	192.168.1.9	192.168.1.10	FTP	124	Request: RETR CeciN'estPasDuToutUnTitreDestin\303\251ACacherLeContenuDuDocument.txt
45	3.016728	192.168.1.9	192.168.1.10	TCP	66	62099 → 49060 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=128 SACK_PERM=1
46	3.016882	192.168.1.10	192.168.1.9	TCP	66	49060 → 62099 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=32
47	3.016918	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [ACK] Seq=1 Ack=1 Win=4194304 Len=0
48	3.017148	192.168.1.10	192.168.1.9	FTP	177	Response: 150 Opening BINARY mode data connection for CeciN'estPasDuToutUnTitreDestin\30
49	3.017237	192.168.1.10	192.168.1.9	TCP	1514	49060 → 62099 [ACK] Seq=1 Ack=1 Win=29216 Len=1460
50	3.017256	192.168.1.10	192.168.1.9	TCP	1514	49060 → 62099 [ACK] Seq=1461 Ack=1 Win=29216 Len=1460
51	3.017276	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [ACK] Seq=1 Ack=2921 Win=4194304 Len=0
52	3.017294	192.168.1.10	192.168.1.9	TCP	1514	49060 → 62099 [ACK] Seq=2921 Ack=1 Win=29216 Len=1460
53	3.017313	192.168.1.10	192.168.1.9	TCP	1514	49060 → 62099 [ACK] Seq=4381 Ack=1 Win=29216 Len=1460
54	3.017331	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [ACK] Seq=1 Ack=5841 Win=4194304 Len=0
55	3.017348	192.168.1.10	192.168.1.9	TCP	1514	49060 → 62099 [ACK] Seq=5841 Ack=1 Win=29216 Len=1460
56	3.017379	192.168.1.10	192.168.1.9	TCP	1144	49060 → 62099 [PSH, ACK] Seq=7301 Ack=1 Win=29216 Len=1090
57	3.017398	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [ACK] Seq=1 Ack=8391 Win=4194304 Len=0
58	3.017425	192.168.1.10	192.168.1.9	TCP	60	49060 → 62099 [FIN, ACK] Seq=8391 Ack=1 Win=29216 Len=0
59	3.017445	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [ACK] Seq=1 Ack=8392 Win=4194304 Len=0
60	3.017592	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [FIN, ACK] Seq=1 Ack=8392 Win=4194304 Len=0
61	3.017628	192.168.1.10	192.168.1.9	FTP	78	Response: 226 Transfer complete.

Figure 16: TCP Sent & Received

We can see here that the source sends the segments (the length of data is fixed at 1460) :

- SEQ = 1 & ACK = 1
- SEQ = 1461 & ACK = 1

- SEQ = 2921 & ACK = 1
- SEQ = 4381 & ACK = 1
- SEQ = 5841 & ACK = 1
- SEQ = 7301 & ACK = 1
- FIN : SEQ = 8391 & ACK = 1

4.1.3 Sequence numbers reception

- SEQ = 1 & ACK = 2921
- SEQ = 1461 & ACK = 1
- SEQ = 2921 & ACK = 5841
- SEQ = 4381 & ACK = 8391
- SEQ = 5841 & ACK = 8932

4.1.4 Retransmissions

There is no retransmissions in our case, since there is no packet marked as such in Wireshark and that all the numbers in the sequence numbers from the source and the acknowledgments of the host follow each others.

4.1.5 Buffers impact

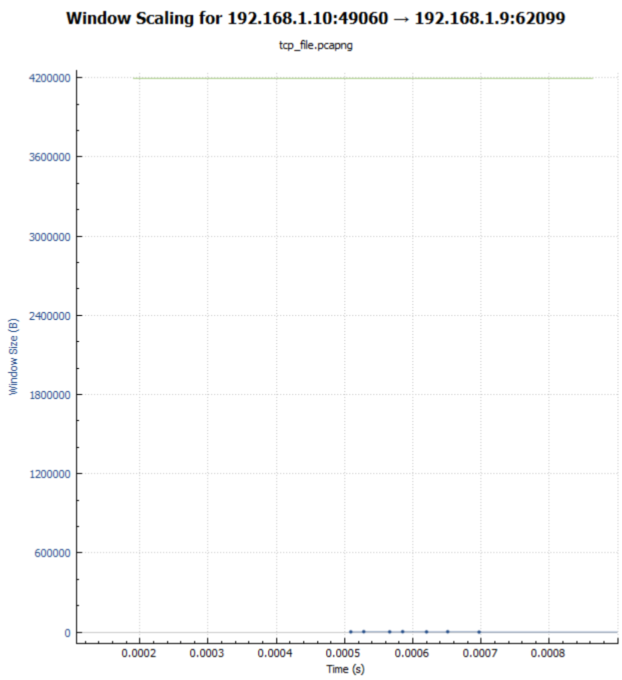


Figure 17: Window scaling from PC1 to Host

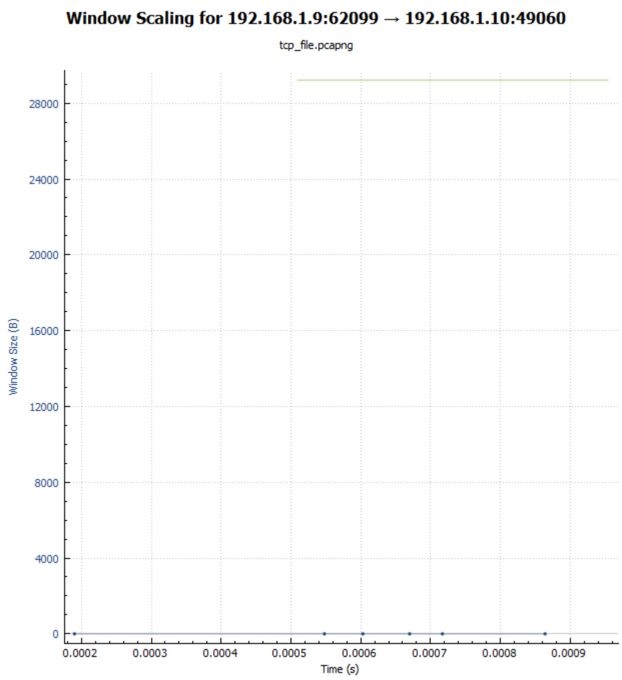


Figure 18: Window scaling from Host to PC1

We can see here that the maximum size of the window depends on each machine.
The Host sends information that its maximum window size is 4 194 304. The PC1 sends information that its maximum windows size is 29 216.

They are both represented by the green line on each graph. The blue line represents the size

of the data that is actually transmitted between each machine.
From the PC1 to Host it is equal to 0 since the host just receive data. PC1 one sends segments of a size of 1 460 which is very very small compared to the 4 000 000 the host can receive.

4.1.6 Throughput graph

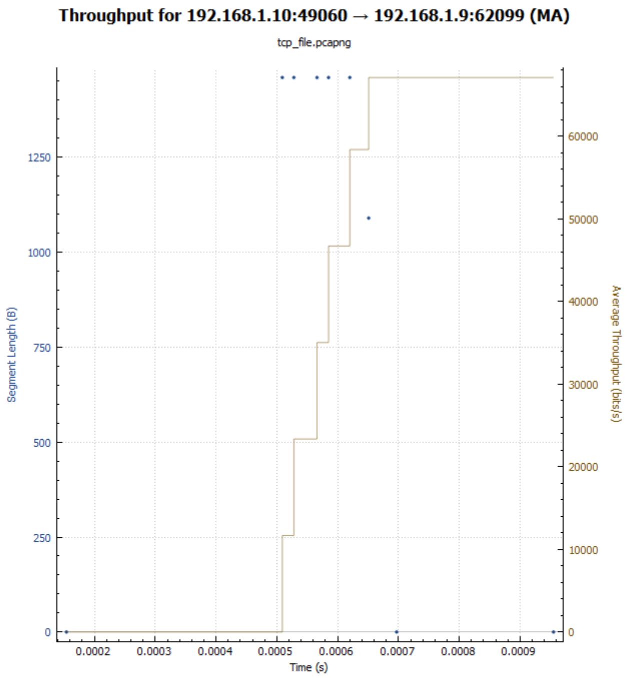


Figure 19: Throughput

The graph shows the amount of data being sent during the time of the exchange. Here we can see that it is consistent with no loss or latency.

4.2 FTP Understanding

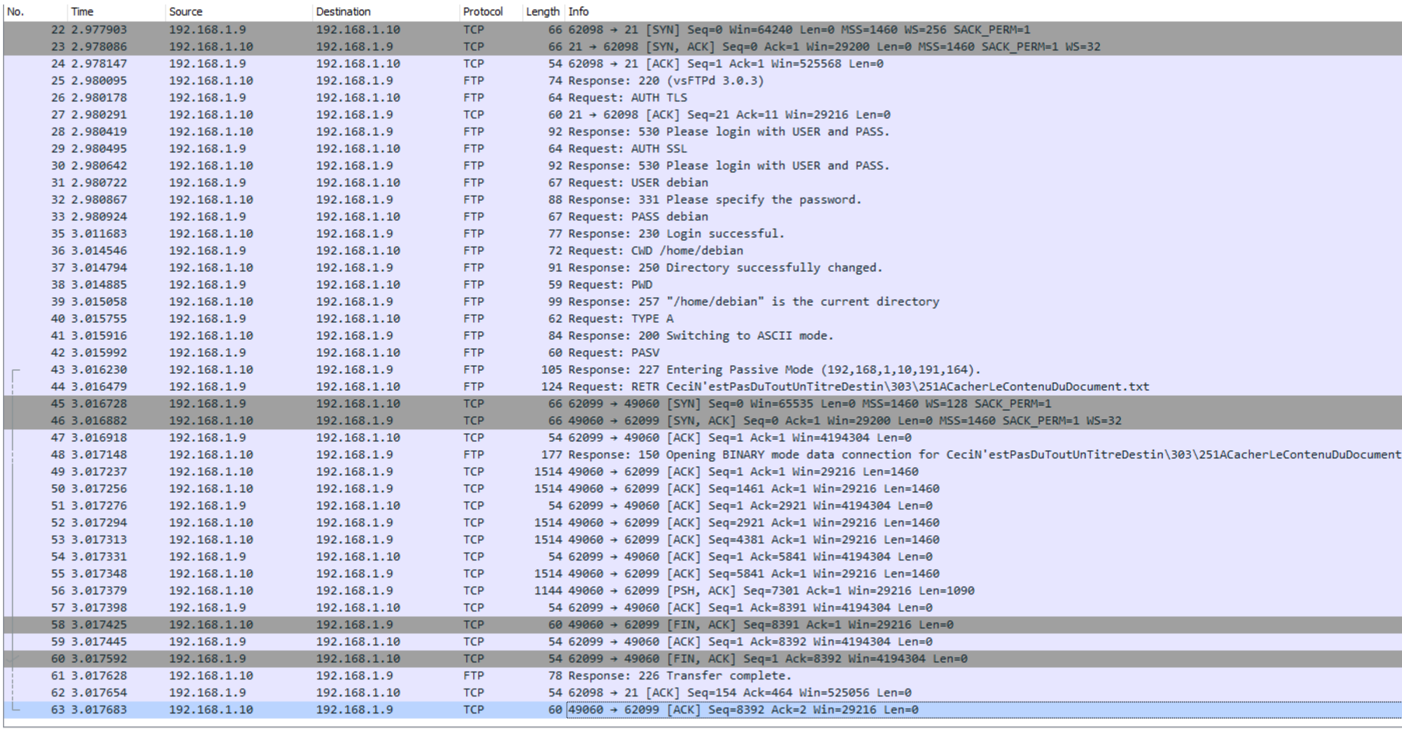
4.2.1 Connections for one file

We can see here that there is 2 connections needed to exchange a file the first one to securely connect to the machine using TLS and then navigate to the correct directory. The second one is to actually transfer the file as seen above.

- Authentication :
  - Source port (HOST) : 62098
  - Destination port (PC1) : 21
- File Transfer :
  - Source port : 62099
  - Destination port : 49060

4.2.2 Connections for three files

As before there is a connection made the authentication and then 3 distinct connections for the 3 files, to a connection for each.



No.	Time	Source	Destination	Protocol	Length	Info
22	2.977903	192.168.1.9	192.168.1.10	TCP	66	62098 → 21 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
23	2.978086	192.168.1.10	192.168.1.9	TCP	66	21 → 62098 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=32
24	2.978147	192.168.1.9	192.168.1.10	TCP	54	62098 → 21 [ACK] Seq=1 Ack=1 Win=525568 Len=0
25	2.980095	192.168.1.10	192.168.1.9	FTP	74	Response: 220 (vsFTPd 3.0.3)
26	2.980178	192.168.1.9	192.168.1.10	FTP	64	Request: AUTH TLS
27	2.980291	192.168.1.10	192.168.1.9	TCP	60	21 → 62098 [ACK] Seq=21 Ack=11 Win=29216 Len=0
28	2.980419	192.168.1.10	192.168.1.9	FTP	92	Response: 530 Please login with USER and PASS.
29	2.980495	192.168.1.9	192.168.1.10	FTP	64	Request: AUTH SSL
30	2.980642	192.168.1.10	192.168.1.9	FTP	92	Response: 530 Please login with USER and PASS.
31	2.980722	192.168.1.9	192.168.1.10	FTP	67	Request: USER debian
32	2.980867	192.168.1.10	192.168.1.9	FTP	88	Response: 331 Please specify the password.
33	2.980924	192.168.1.9	192.168.1.10	FTP	67	Request: PASS debian
35	3.011683	192.168.1.10	192.168.1.9	FTP	77	Response: 230 Login successful.
36	3.014546	192.168.1.9	192.168.1.10	FTP	72	Request: CWD /home/debian
37	3.014794	192.168.1.10	192.168.1.9	FTP	91	Response: 250 Directory successfully changed.
38	3.014885	192.168.1.9	192.168.1.10	FTP	59	Request: PWD
39	3.015058	192.168.1.10	192.168.1.9	FTP	99	Response: 257 "/home/debian" is the current directory
40	3.015755	192.168.1.9	192.168.1.10	FTP	62	Request: TYPE A
41	3.015916	192.168.1.10	192.168.1.9	FTP	84	Response: 200 Switching to ASCII mode.
42	3.015992	192.168.1.9	192.168.1.10	FTP	60	Request: PASV
43	3.016230	192.168.1.10	192.168.1.9	FTP	105	Response: 227 Entering Passive Mode (192,168,1,10,191,164).
44	3.016479	192.168.1.9	192.168.1.10	FTP	124	Request: RETR CeciN'estPasDuToutUnTitreDestin\303\251ACacherLeContenuDuDocument.txt
45	3.016728	192.168.1.9	192.168.1.10	TCP	66	62099 → 49060 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=128 SACK_PERM=1
46	3.016882	192.168.1.10	192.168.1.9	TCP	66	49060 → 62099 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=32
47	3.016918	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [ACK] Seq=1 Ack=1 Win=4194304 Len=0
48	3.017148	192.168.1.10	192.168.1.9	FTP	177	Response: 150 Opening BINARY mode data connection for CeciN'estPasDuToutUnTitreDestin\303\251ACacherLeContenuDuDocument
49	3.017237	192.168.1.10	192.168.1.9	TCP	1514	49060 → 62099 [ACK] Seq=1 Ack=1 Win=29216 Len=1460
50	3.017256	192.168.1.10	192.168.1.9	TCP	1514	49060 → 62099 [ACK] Seq=1461 Ack=1 Win=29216 Len=1460
51	3.017276	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [ACK] Seq=1 Ack=2921 Win=4194304 Len=0
52	3.017294	192.168.1.10	192.168.1.9	TCP	1514	49060 → 62099 [ACK] Seq=2921 Ack=1 Win=29216 Len=1460
53	3.017313	192.168.1.10	192.168.1.9	TCP	1514	49060 → 62099 [ACK] Seq=4381 Ack=1 Win=29216 Len=1460
54	3.017331	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [ACK] Seq=1 Ack=5841 Win=4194304 Len=0
55	3.017348	192.168.1.10	192.168.1.9	TCP	1514	49060 → 62099 [ACK] Seq=5841 Ack=1 Win=29216 Len=1460
56	3.017379	192.168.1.10	192.168.1.9	TCP	1144	49060 → 62099 [PSH, ACK] Seq=7301 Ack=1 Win=29216 Len=1090
57	3.017398	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [ACK] Seq=1 Ack=8391 Win=4194304 Len=0
58	3.017425	192.168.1.10	192.168.1.9	TCP	60	49060 → 62099 [FIN, ACK] Seq=8391 Ack=1 Win=29216 Len=0
59	3.017445	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [ACK] Seq=1 Ack=8392 Win=4194304 Len=0
60	3.017592	192.168.1.9	192.168.1.10	TCP	54	62099 → 49060 [FIN, ACK] Seq=1 Ack=8392 Win=4194304 Len=0
61	3.017628	192.168.1.10	192.168.1.9	FTP	78	Response: 226 Transfer complete.
62	3.017654	192.168.1.9	192.168.1.10	TCP	54	62098 → 21 [ACK] Seq=154 Ack=464 Win=525056 Len=0
63	3.017683	192.168.1.10	192.168.1.9	TCP	60	49060 → 62099 [ACK] Seq=8392 Ack=2 Win=29216 Len=0

Figure 20: Connections to transfer a file

- Authentication :
  - Source port (HOST) : 62182
  - Destination port (PC1) : 21
- File Transfer :
  - Source port : 62183
  - Destination port : 51990
- File Transfer :
  - Source port : 62184
  - Destination port : 55773
- File Transfer :
  - Source port : 62185
  - Destination port : 20477

4.2.3 Read content

We are able to read the content from the file. Since it is a txt file we see not only the binary data but also whole characters and sentences. In itself the protocol does not provide secure transfer so every file that passes can be read on the network. A good way to avoid since problem is to encrypt the data before sending it.

4.2.4 FTP sequence diagram

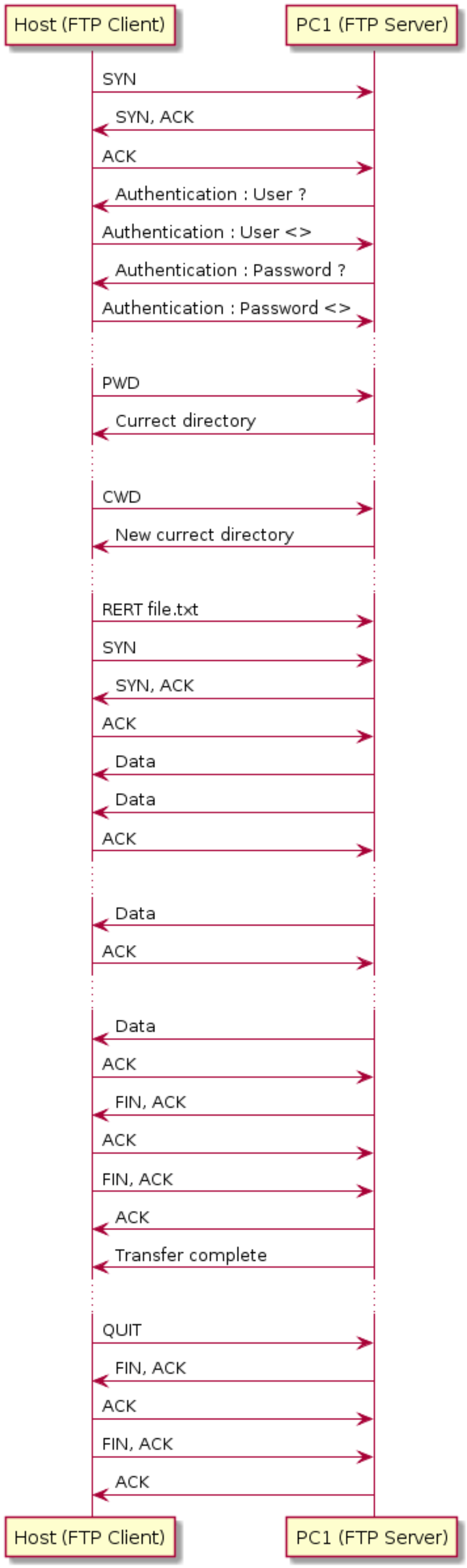


Figure 21: FTP sequence diagram