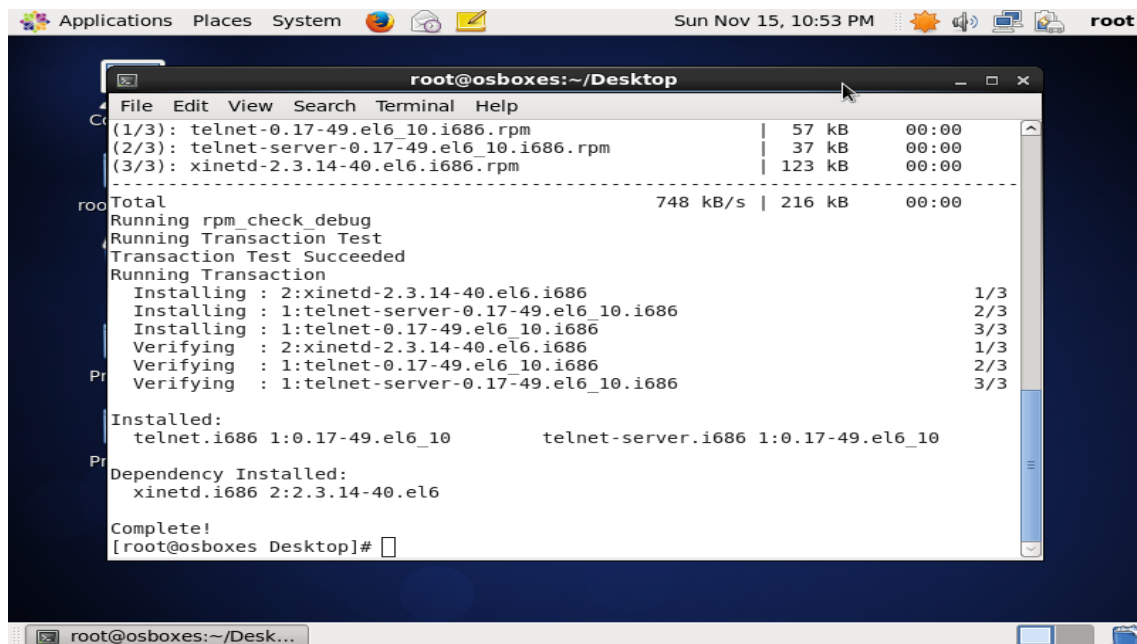


Práctica 8

Guillermo Marcos García

Servicio Telnet

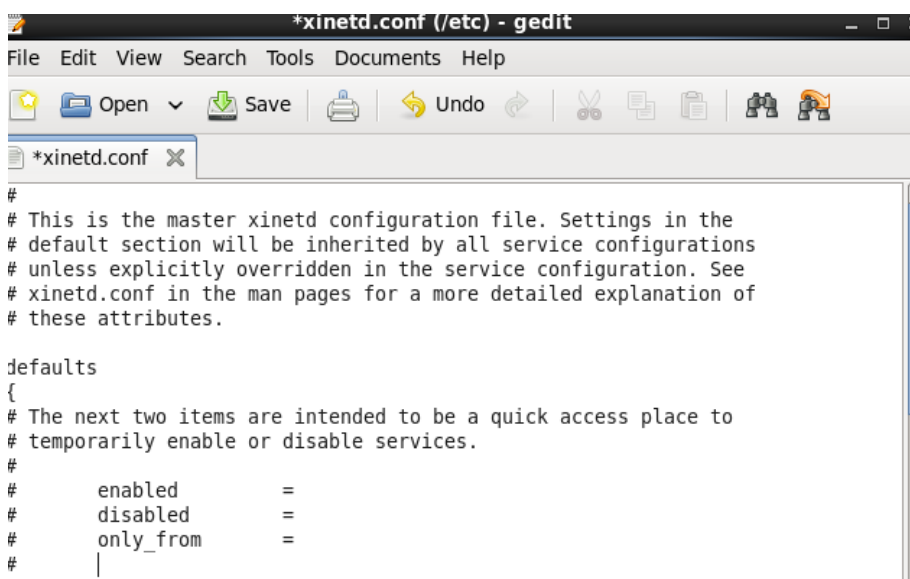
Proceso de Instalación de Telnet



```
root@osboxes: ~/Desktop
File Edit View Search Terminal Help
(1/3): telnet-0.17-49.el6_10.i686.rpm | 57 kB 00:00
(2/3): telnet-server-0.17-49.el6_10.i686.rpm | 37 kB 00:00
(3/3): xinetd-2.3.14-40.el6.i686.rpm | 123 kB 00:00
-----
Total | 748 kB/s | 216 kB 00:00
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
Installing : 2:xinetd-2.3.14-40.el6.i686 1/3
Installing : 1:telnet-server-0.17-49.el6_10.i686 2/3
Installing : 1:telnet-0.17-49.el6_10.i686 3/3
Verifying : 2:xinetd-2.3.14-40.el6.i686 1/3
Verifying : 1:telnet-0.17-49.el6_10.i686 2/3
Verifying : 1:telnet-server-0.17-49.el6_10.i686 3/3
Installed:
telnet.i686 1:0.17-49.el6_10 telnet-server.i686 1:0.17-49.el6_10
Dependency Installed:
xinetd.i686 2:2.3.14-40.el6
Complete!
[root@osboxes Desktop]#
```

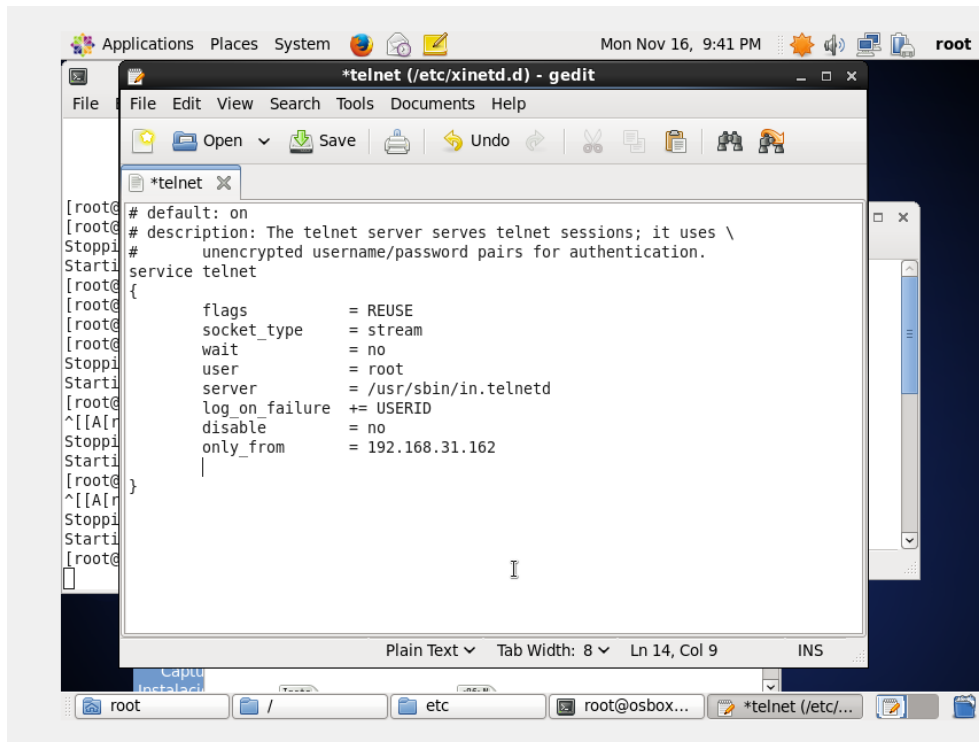
Ejercicio 1

Cerramos el acceso a los servicios desde cualquier IP con la directiva `only_from =`

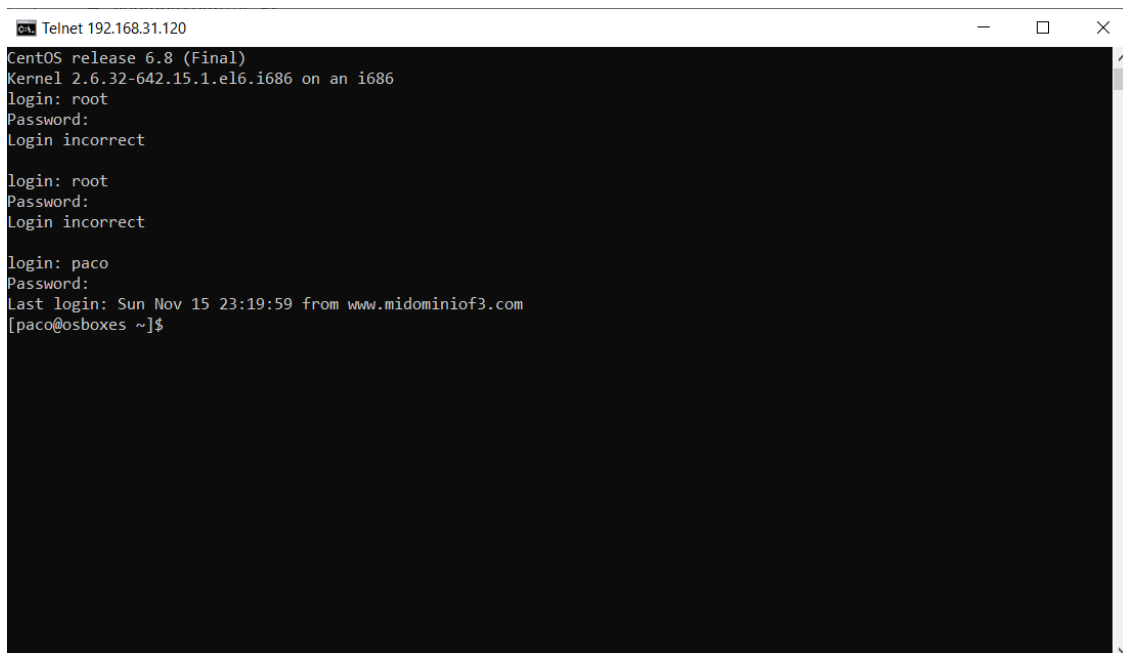


```
*xinetd.conf (/etc) - gedit
File Edit View Search Tools Documents Help
* Open Save Undo Cut Copy Paste
*xinetd.conf
#
# This is the master xinetd configuration file. Settings in the
# default section will be inherited by all service configurations
# unless explicitly overridden in the service configuration. See
# xinetd.conf in the man pages for a more detailed explanation of
# these attributes.
defaults
{
# The next two items are intended to be a quick access place to
# temporarily enable or disable services.
#
# enabled =
# disabled =
# only_from =
#
```

Después en la sección de configuración del telnet damos acceso al servicio a la IP del cliente (la de Windows en este caso) :

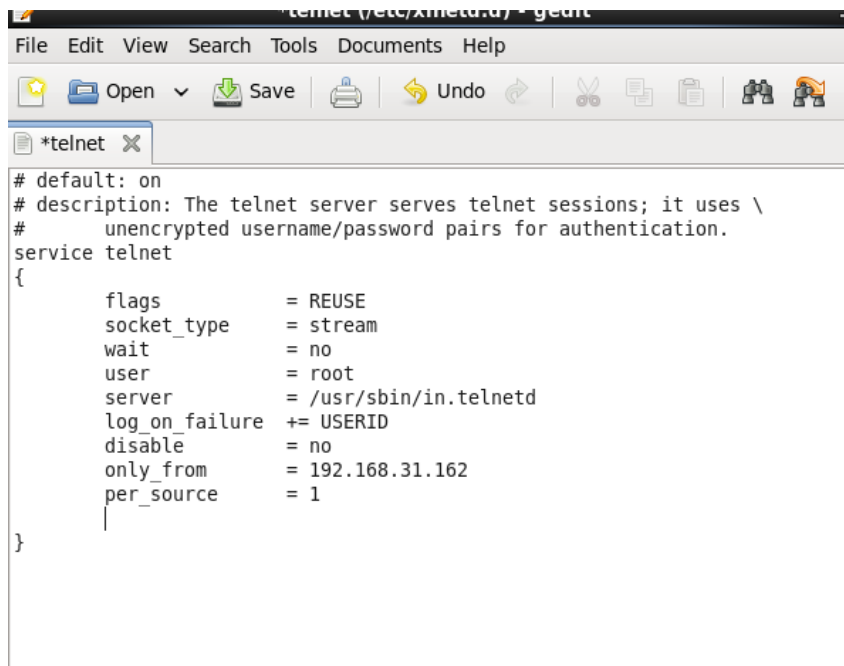


Vemos como nos conectamos desde Windows:



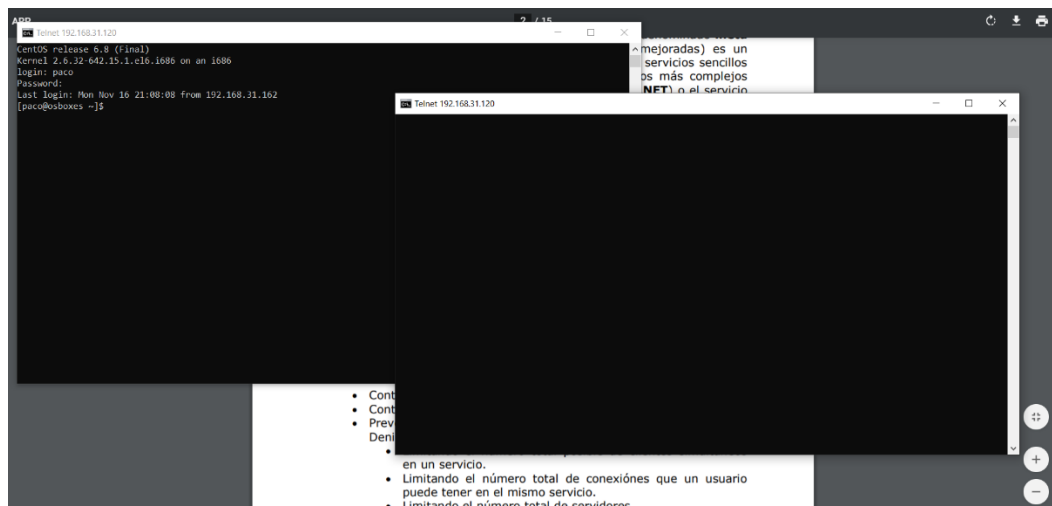
Ejercicio 2

Probamos la directiva `per_source`:

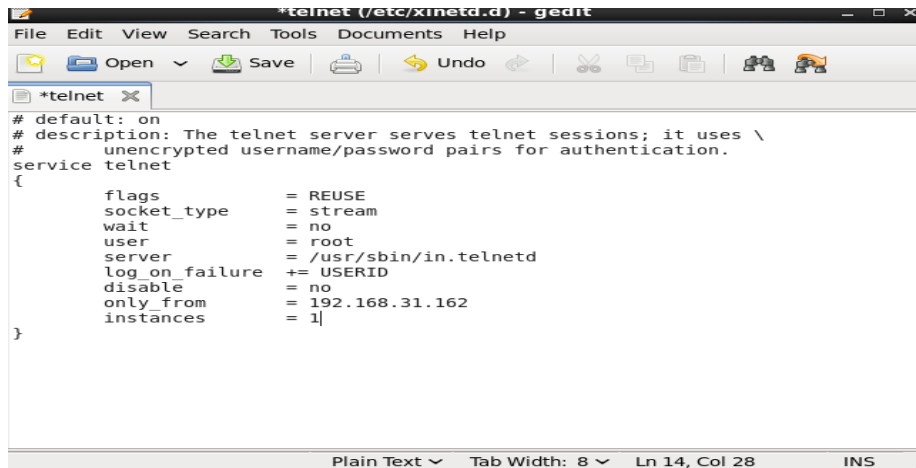


```
# default: on
# description: The telnet server serves telnet sessions; it uses \
#      unencrypted username/password pairs for authentication.
service telnet
{
    flags             = REUSE
    socket_type       = stream
    wait              = no
    user              = root
    server             = /usr/sbin/in.telnetd
    log_on_failure    += USERID
    disable           = no
    only_from         = 192.168.31.162
    per_source        = 1
}
```

Al intentar conectarnos dos veces al mismo tiempo desde el mismo cliente vemos que la segunda conexión no se realiza.

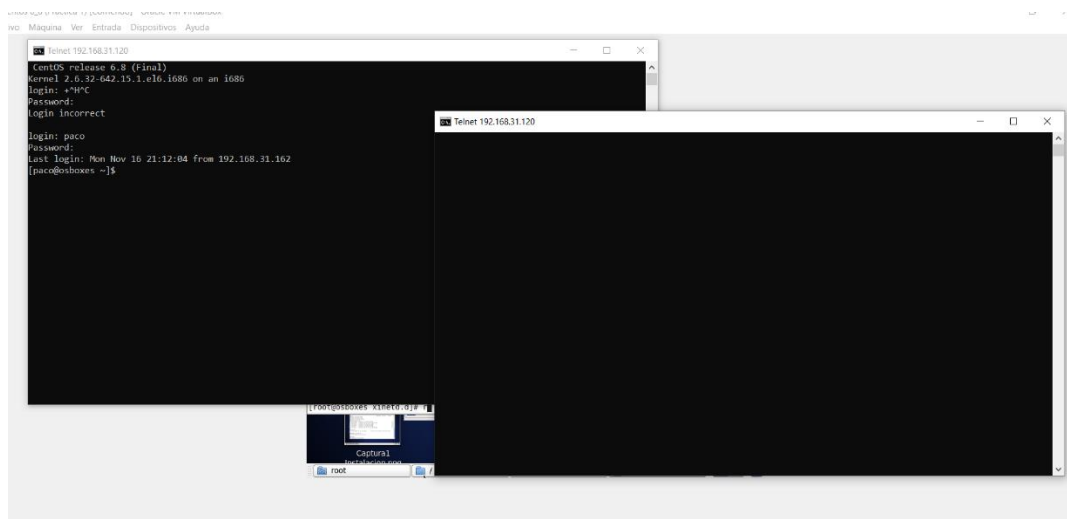


Ahora probamos la directiva `instances`:

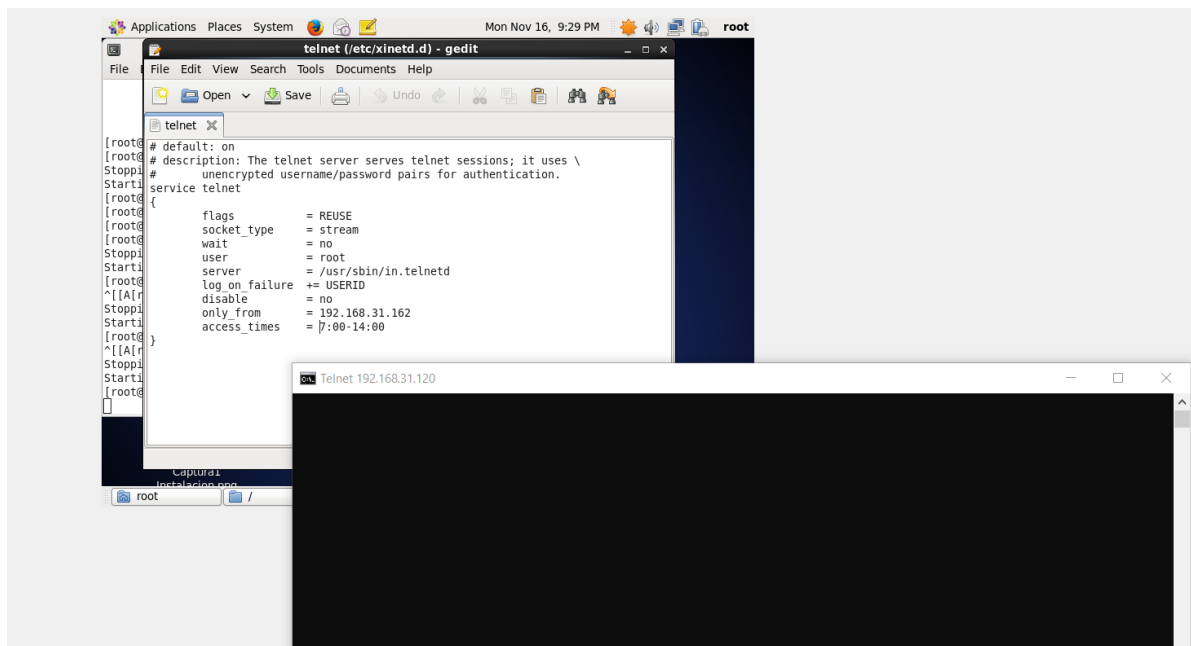


```
# default: on
# description: The telnet server serves telnet sessions; it uses \
#               unencrypted username/password pairs for authentication.
service telnet
{
    flags             = REUSE
    socket_type       = stream
    wait              = no
    user              = root
    server            = /usr/sbin/in.telnetd
    log_on_failure    += USERID
    disable           = no
    only_from         = 192.168.31.162
    instances         = 1
}
```

Instances nos marca el número máximo de conexiones que permite el servicio mientras que per_source marca el número máximo de conexiones que el mismo cliente puede hacer al mismo tiempo. Para comprobarlo hacemos lo mismo que hicimos para comprobar el funcionamiento de la directiva per_source:



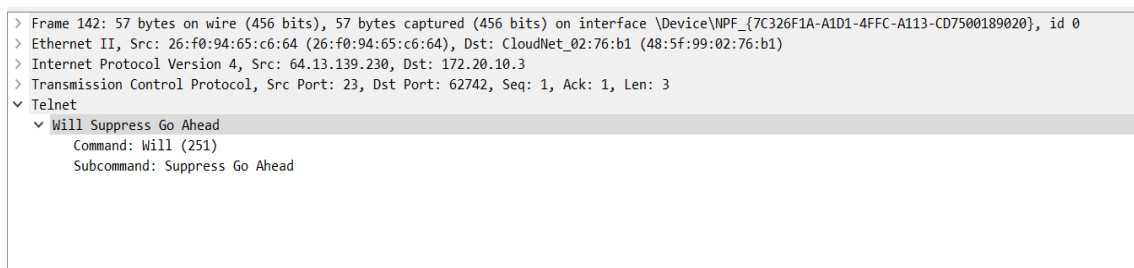
Finalmente probamos la directiva access_time:



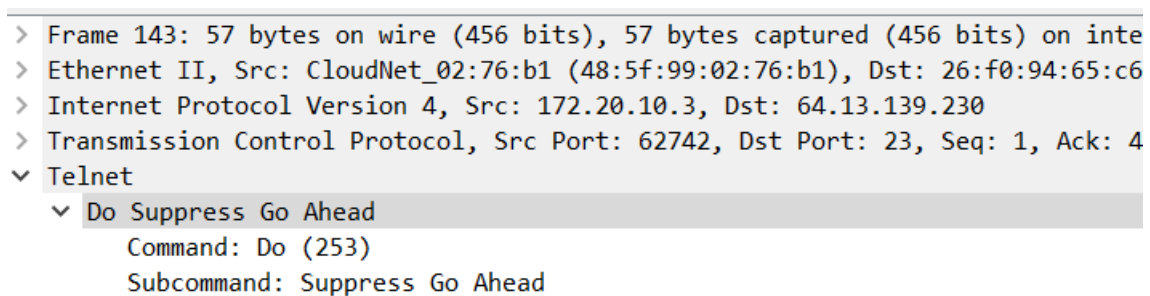
Vemos que si accedemos fuera de la franja horario 7:00-14:00 no nos deja conectarnos.

Ejercicio 3

Wireshark en Windows.



El código utilizado para eliminar el Go Ahead es 251, el comando “Will (251)” y el subcomando “Suppress Go Ahead”.



El cliente responde con el comando Do y el código 253 llevando a cabo la eliminación del go-ahead.

143	2020-11-16	23:43:00,781752	172.20.10.3	64.13.139.230
151	2020-11-16	23:43:01,024928	64.13.139.230	172.20.10.3
152	2020-11-16	23:43:01,025332	172.20.10.3	64.13.139.230
160	2020-11-16	23:43:01,605317	64.13.139.230	172.20.10.3
167	2020-11-16	23:43:01,907929	64.13.139.230	172.20.10.3
168	2020-11-16	23:43:01,908344	172.20.10.3	64.13.139.230
1560	2020-11-16	23:43:10,735078	172.20.10.3	64.13.139.230
1570	2020-11-16	23:43:11,027546	64.13.139.230	172.20.10.3
1571	2020-11-16	23:43:11,027691	172.20.10.3	64.13.139.230
1577	2020-11-16	23:43:11,271120	64.13.139.230	172.20.10.3
1588	2020-11-16	23:43:11,739857	172.20.10.3	64.13.139.230

> Frame 151: 96 bytes on wire (768 bits), 96 bytes captured (768 bits) on int

> Ethernet II, Src: 26:f0:94:65:c6:64 (26:f0:94:65:c6:64), Dst: CloudNet_02:7

> Internet Protocol Version 4, Src: 64.13.139.230, Dst: 172.20.10.3

> Transmission Control Protocol, Src Port: 23, Dst Port: 62742, Seq: 4, Ack:

▼ Telnet

 ▼ Will Echo

 Command: Will (251)

 Subcommand: Echo

 ▼ Do Terminal Type

 Command: Do (253)

 Subcommand: Terminal Type

 ▼ Do Negotiate About Window Size

 Command: Do (253)

Aquí vemos como el servidor propone implementar el eco remoto.

152	2020-11-16	23:43:01,025332	172.20.10.3	64.13.139.230	TELNET	57 Telnet Data ...
160	2020-11-16	23:43:01,605317	64.13.139.230	172.20.10.3	TELNET	1111 Telnet Data ...
167	2020-11-16	23:43:01,907929	64.13.139.230	172.20.10.3	TELNET	72 Telnet Data ...
168	2020-11-16	23:43:01,908344	172.20.10.3	64.13.139.230	TELNET	64 Telnet Data ...
1560	2020-11-16	23:43:10,735078	172.20.10.3	64.13.139.230	TELNET	55 Telnet Data ...
1570	2020-11-16	23:43:11,027546	64.13.139.230	172.20.10.3	TELNET	55 Telnet Data ...
1571	2020-11-16	23:43:11,027691	172.20.10.3	64.13.139.230	TELNET	55 Telnet Data ...
1577	2020-11-16	23:43:11,271120	64.13.139.230	172.20.10.3	TELNET	55 Telnet Data ...
1588	2020-11-16	23:43:11,739857	172.20.10.3	64.13.139.230	TELNET	55 Telnet Data ...

> Frame 152: 57 bytes on wire (456 bits), 57 bytes captured (456 bits) on interface \Device\NPF_{7C326F1A-A1D1}

> Ethernet II, Src: CloudNet_02:76:b1 (48:5f:99:02:76:b1), Dst: 26:f0:94:65:c6:64 (26:f0:94:65:c6:64)

> Internet Protocol Version 4, Src: 172.20.10.3, Dst: 64.13.139.230

> Transmission Control Protocol, Src Port: 62742, Dst Port: 23, Seq: 4, Ack: 46, Len: 3

▼ Telnet

 ▼ Do Echo

 Command: Do (253)

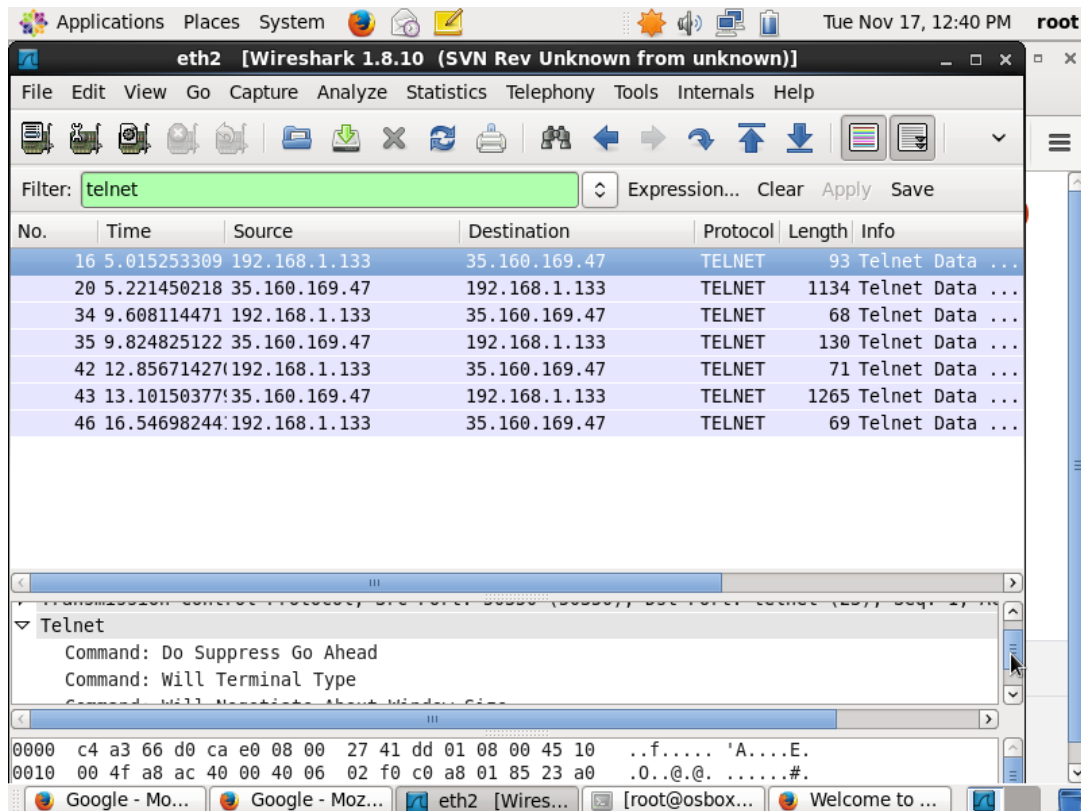
 Subcommand: Echo

Y aquí vemos como el cliente le responde con el Do Echo para que lo implemente.

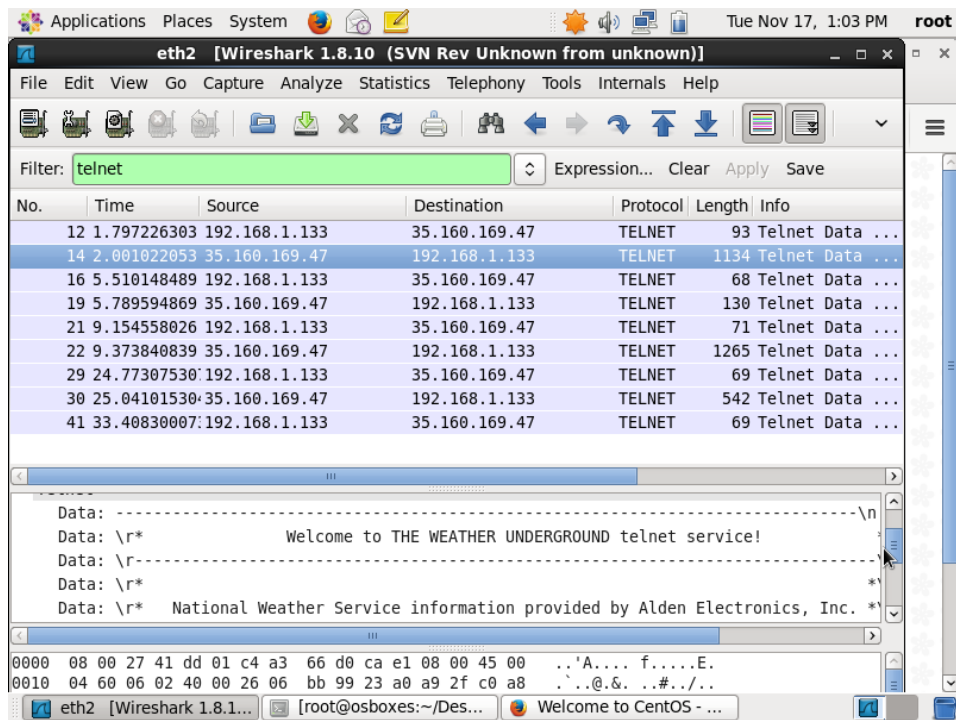
Ejercicio 4

Wireshark en CentOS.

En este caso, al utilizar Wireshark desde CentOS el cliente le manda al servidor la solicitud para que elimine el go-ahead:



Pero no obtenemos respuesta del servidor para ejecutar la solicitud. Este nos contesta únicamente con datos, sin dar respuesta a la solicitud:



Obviamente, al no contestarnos a la solicitud, tampoco se ofrece a implementar el eco remoto.