

Trabajos curso completo

Alejandro Fuentes León

Trabajos curso completo

Alejandro Fuentes León

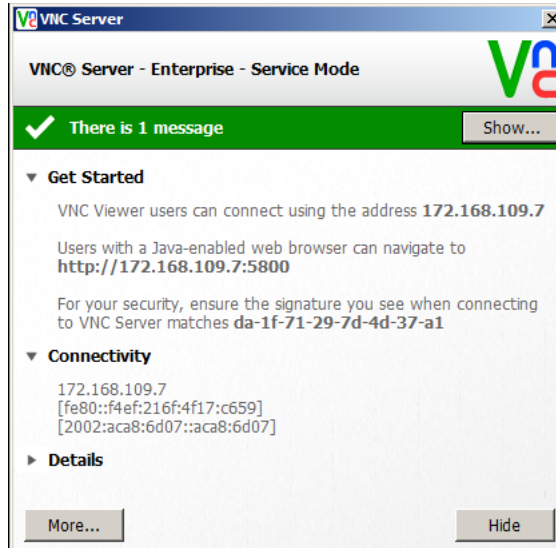
Tabla de contenidos

1. Acceso Remoto	1
Escritorio remoto con VNC	1
Escritorio Remoto con RDP	4
Terminal Server	6
RemoteApp	9
2. Clientes ligeros	11
Configuración del Servidor LTSP	11
Configuración del cliente	12
Comprobación final	12
3. Acceso remoto SSH	13
1. Linux	13
2. Windows	18

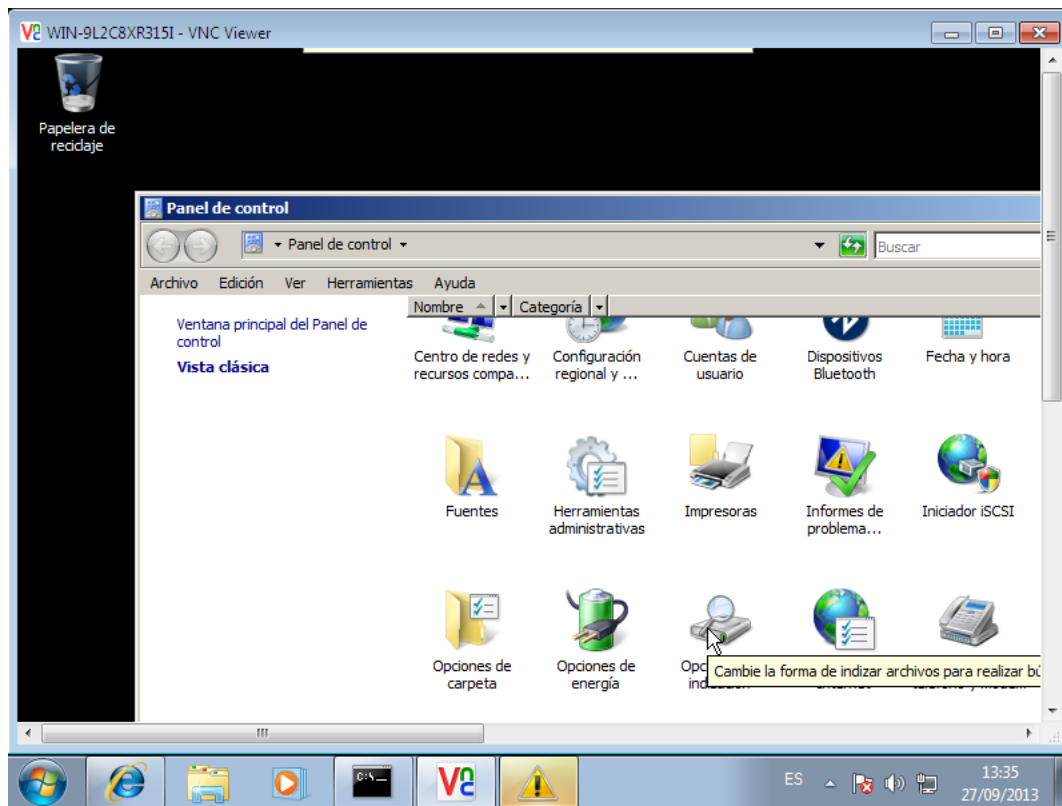
Capítulo 1. Acceso Remoto

Escritorio remoto con VNC

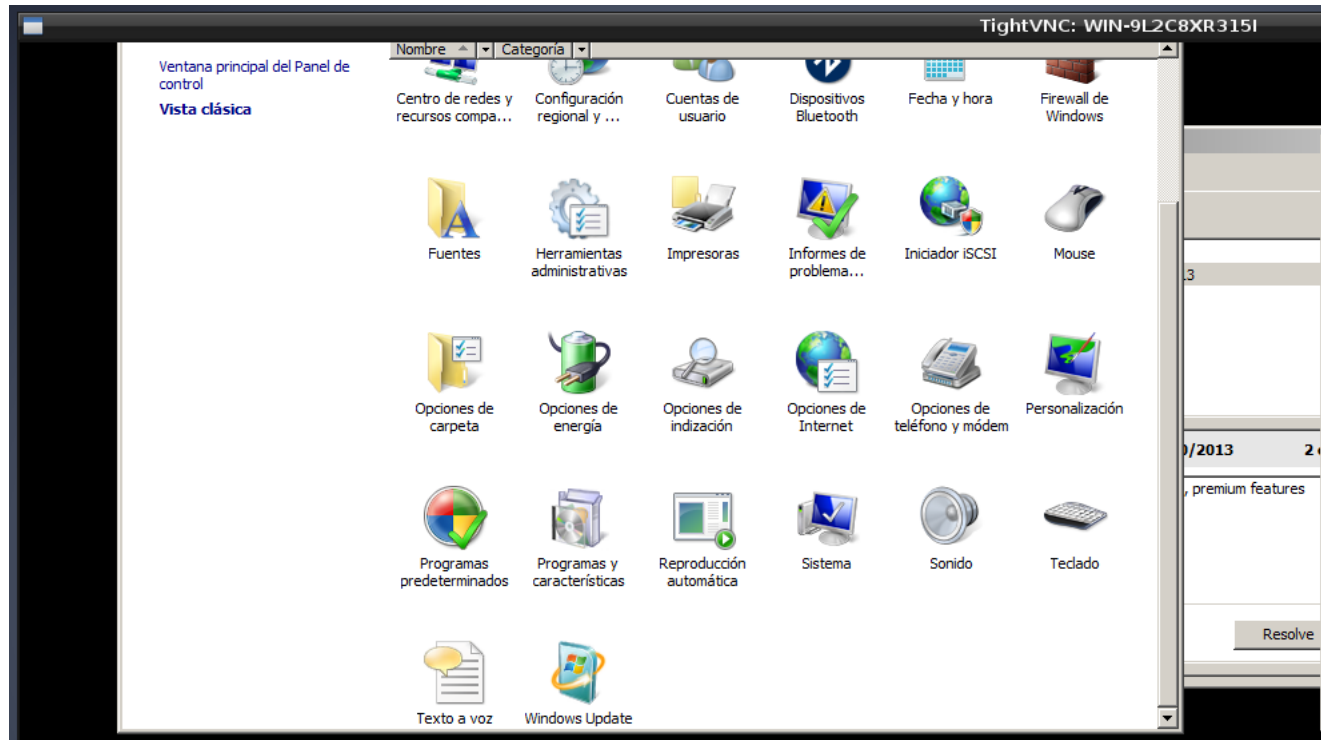
Instalar y configurar VNC para poder acceder a una máquina remota.



Aquí accedemos desde un Cliente Windows al servidor Windows



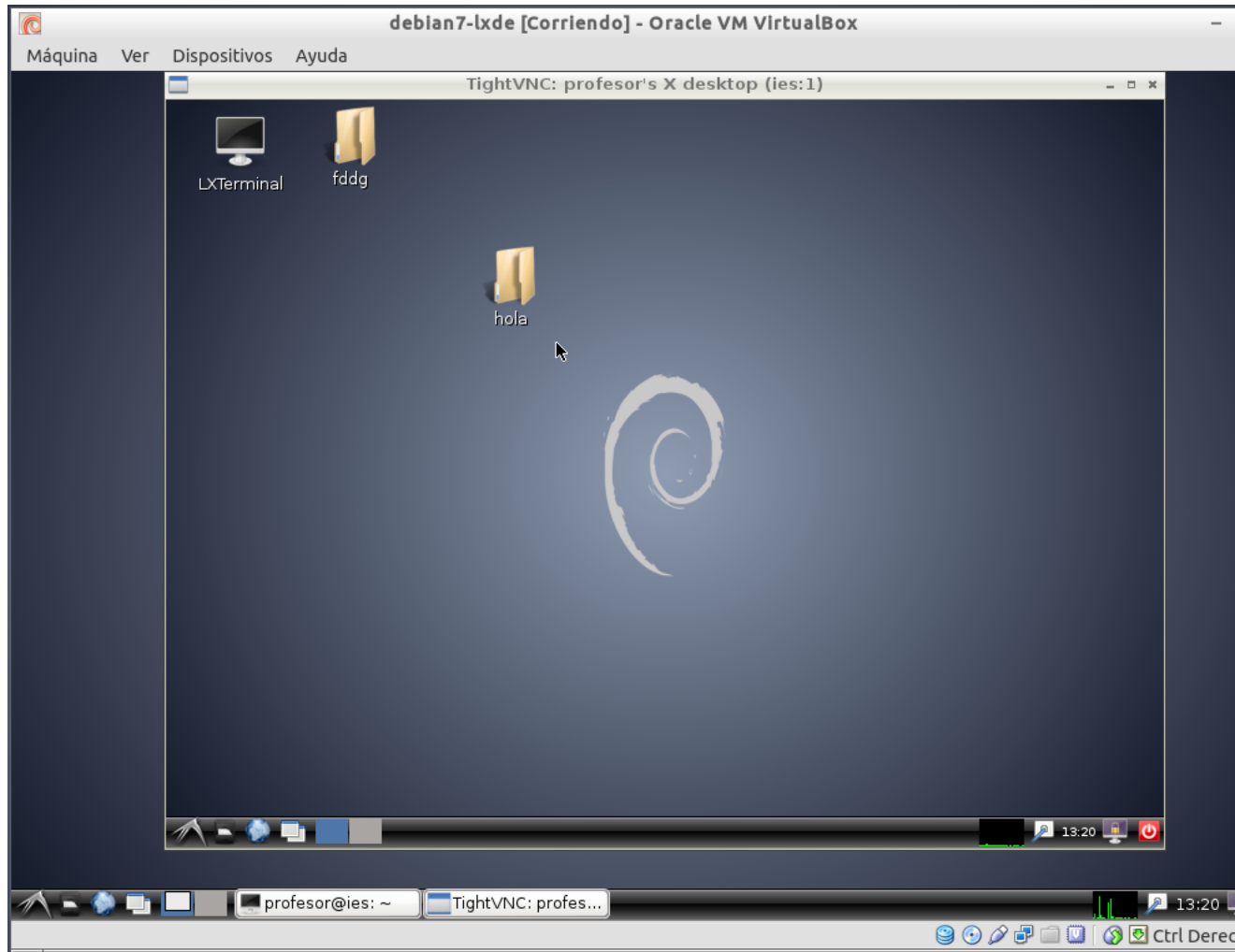
Aquí accedemos desde un Cliente Linux al servidor Windows



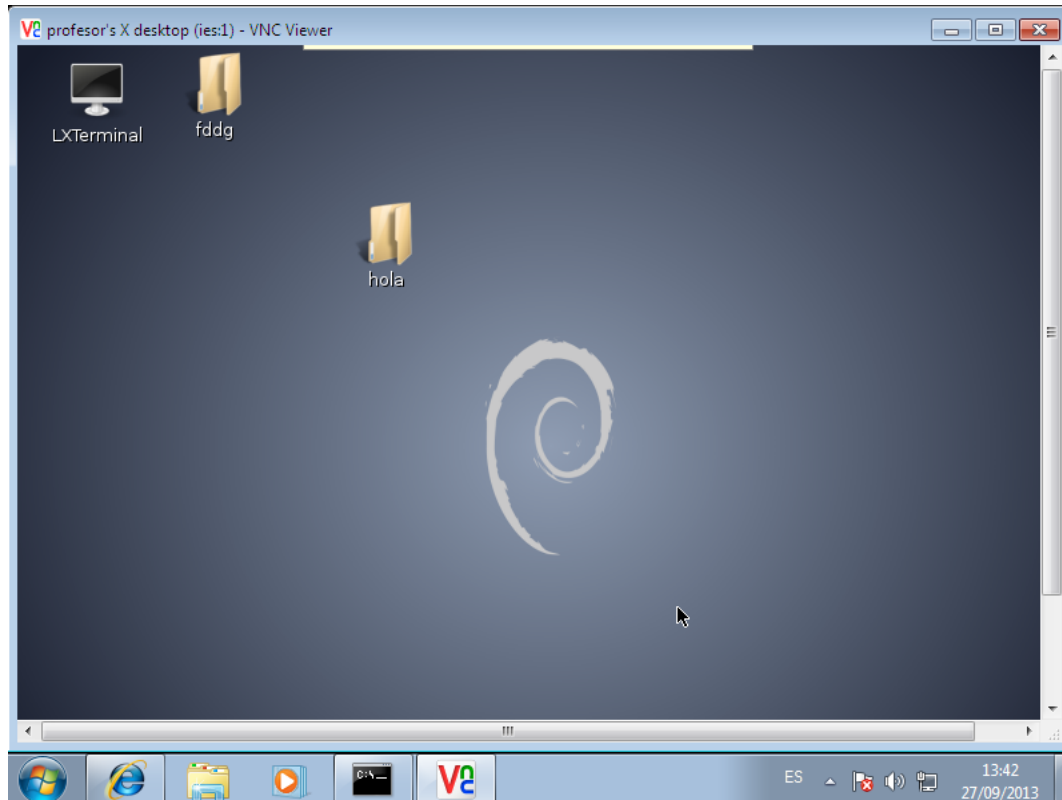
Instalamos las herramientas VNC en linux

```
profesor@ies: ~  
Archivo Edición Pestañas Ayuda  
exit  
profesor@ies:~$ tightvncserver  
You will require a password to access your desktops.  
Password:  
Verify:  
Would you like to enter a view-only password (y/n)? n  
New 'X' desktop is ies:1  
Creating default startup script /home/profesor/.vnc/xstartup  
Starting applications specified in /home/profesor/.vnc/xstartup  
Log file is /home/profesor/.vnc/ies:1.log  
profesor@ies:~$ tightvncserver :1 -geometry 800x600 -depth 24  
A VNC server is already running as :1  
profesor@ies:~$ ifconfig  
bash: ifconfig: no se encontró la orden  
profesor@ies:~$ tightvncserver :1 -geometry 800x600 -depth 24  
A VNC server is already running as :1  
profesor@ies:~$ tightvncserver :1 -geometry 800x600 -depth 24  
A VNC server is already running as :1  
profesor@ies:~$
```

Aquí accedemos desde un Cliente Linux al servidor Linux

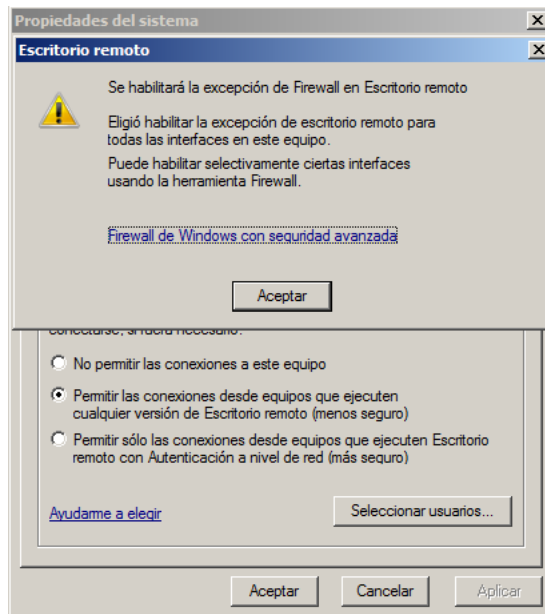


Aquí accedemos desde un Cliente Windows al servidor Linux

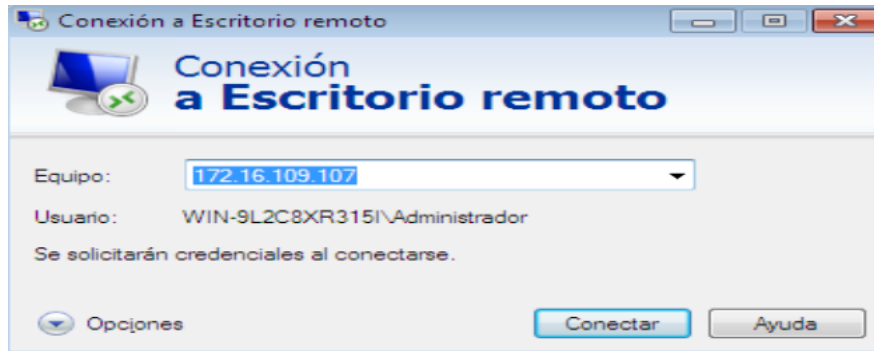


Escritorio Remoto con RDP

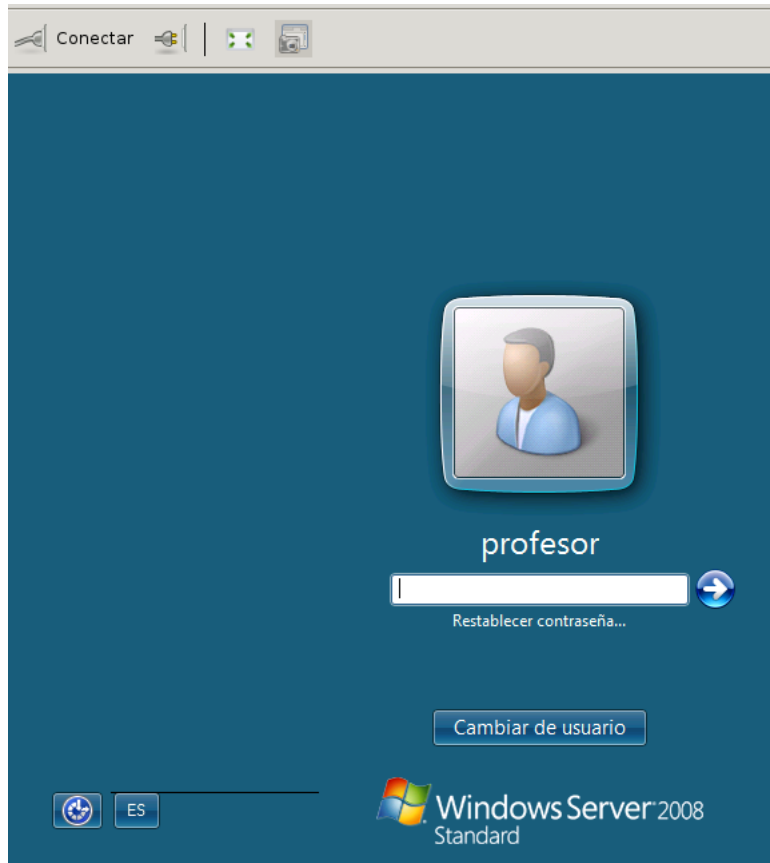
En Windows usamos el escritorio remoto, una herramienta que ya viene instalada.



Aquí accedemos a Windows Server 2008 desde Windows7



Aquí accedemos a Windows Server 2008 desde Linux

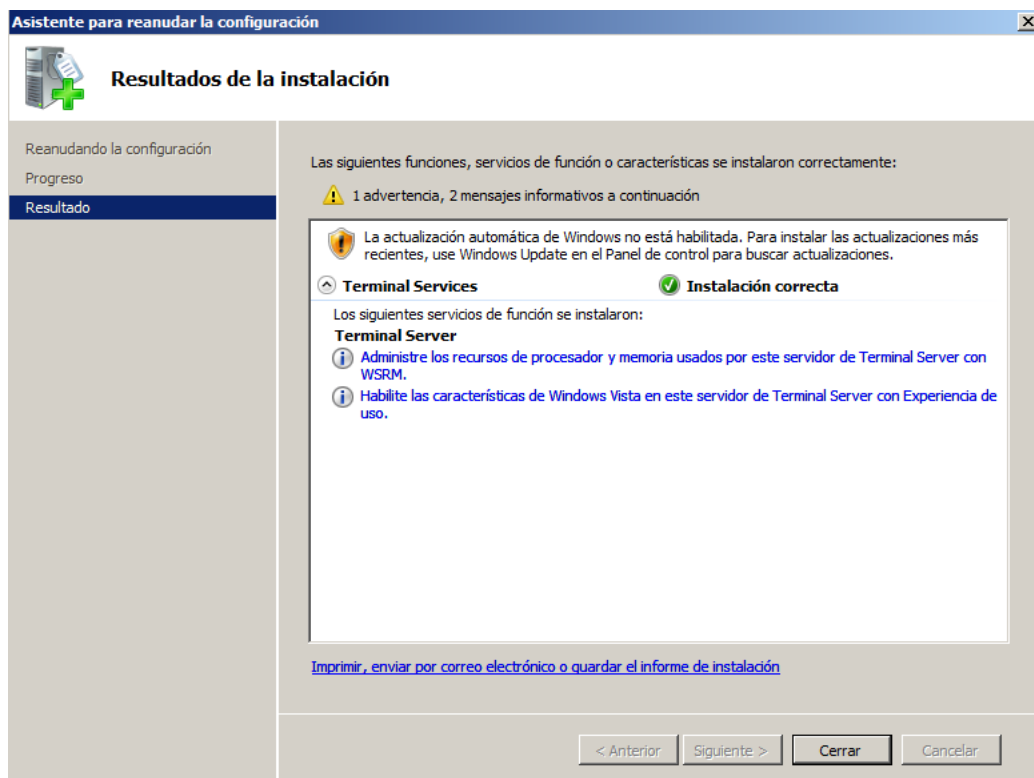


Aquí accedemos a Linux desde Windows instalando en Linux la herramienta xrdp

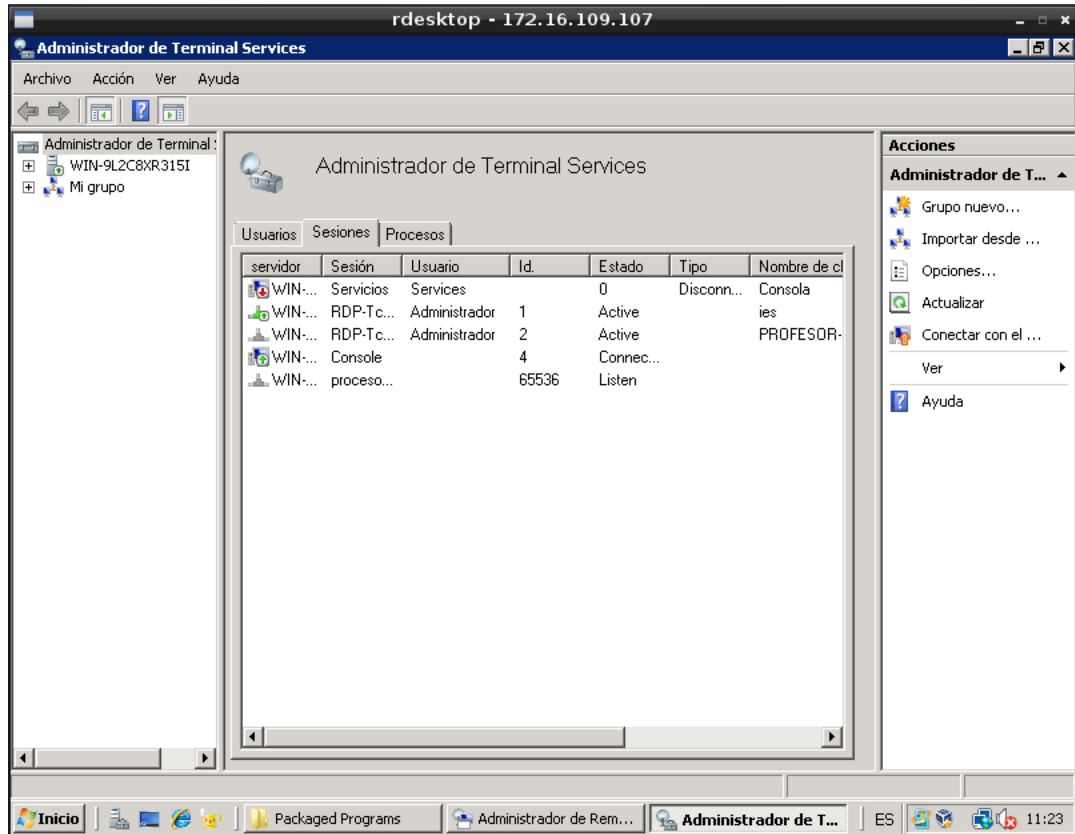


Terminal Server

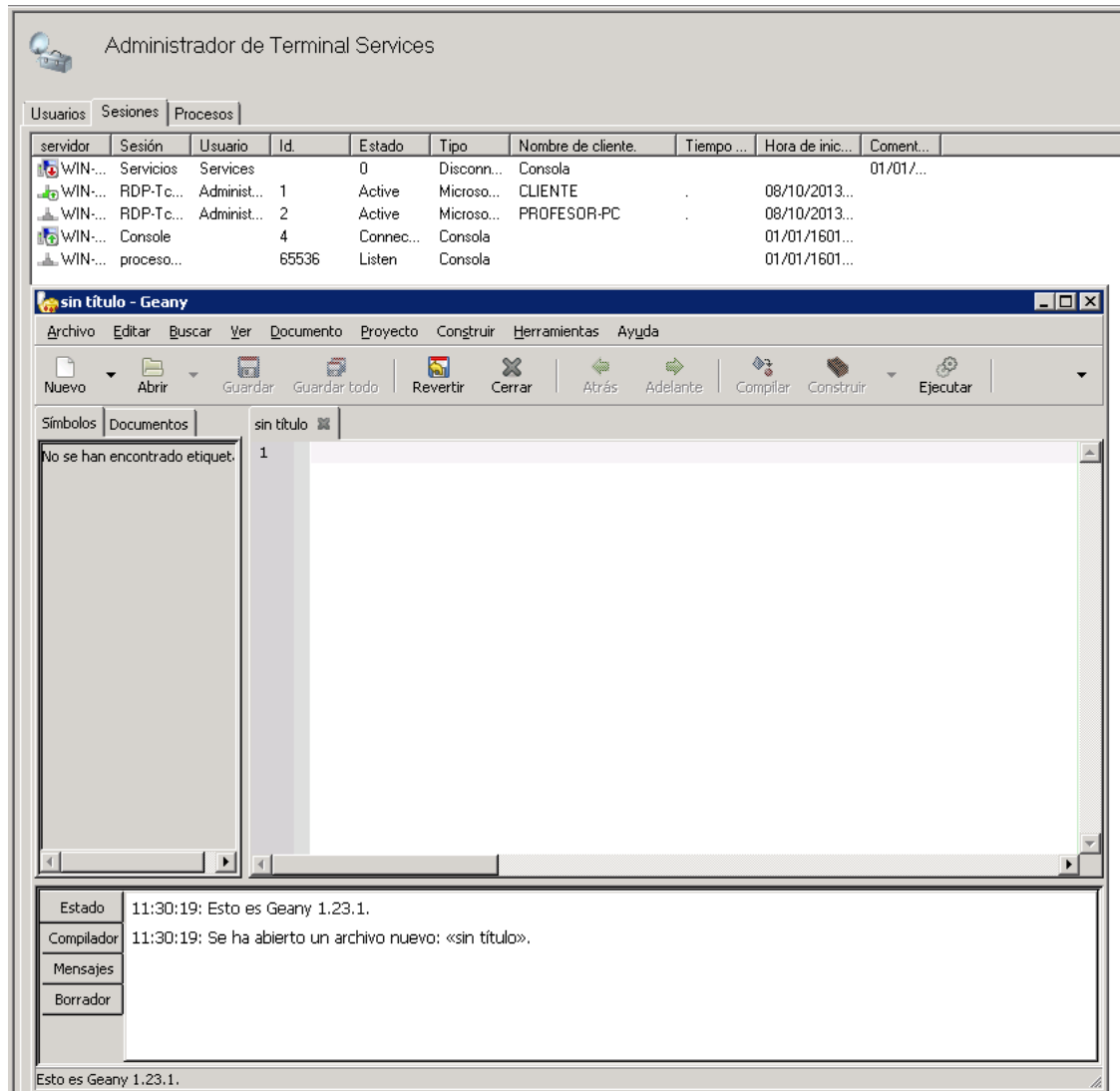
Asistencia de instalación.



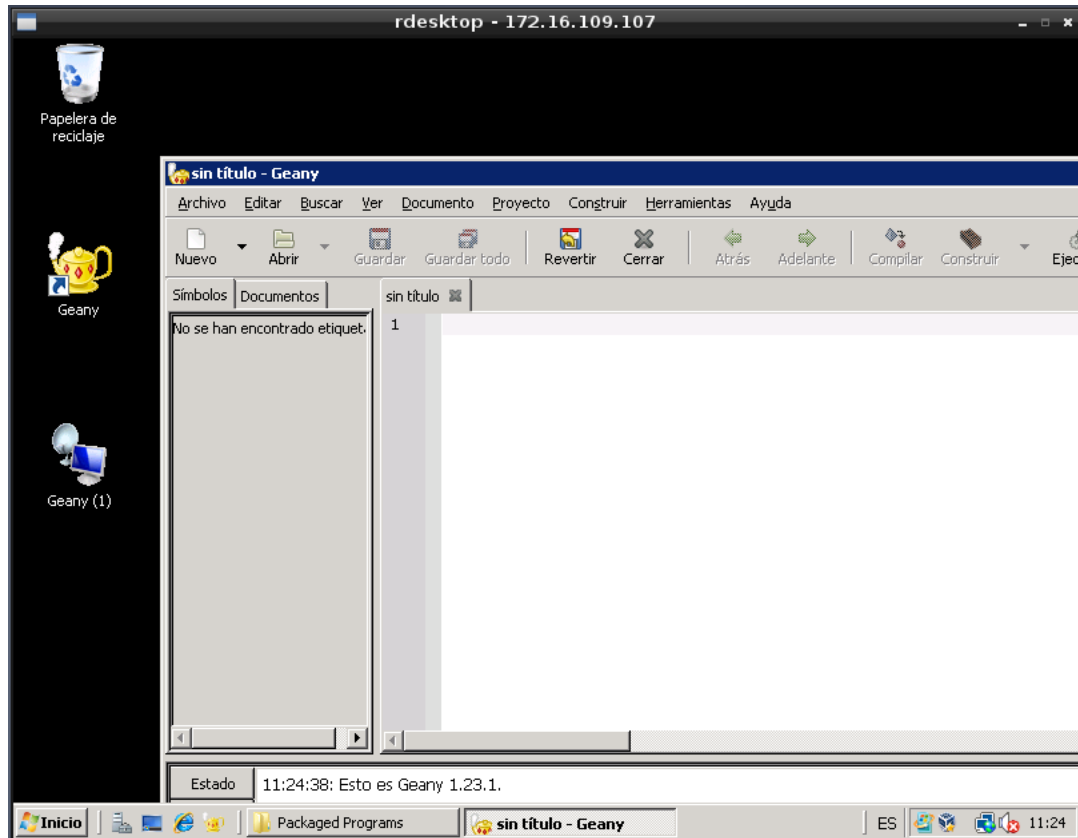
Demostración de que está funcionando de manera correcta el Terminal Server.



Probando una aplicación con el Terminal Server en Windows.

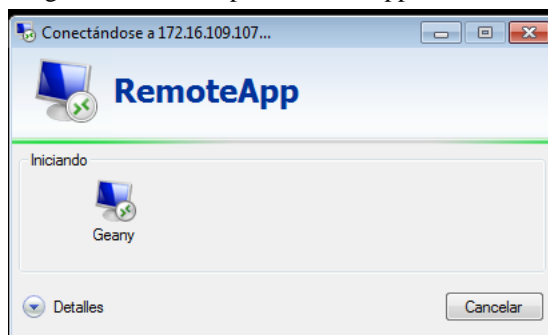


Probando una aplicación con el Terminal Server en Debian.

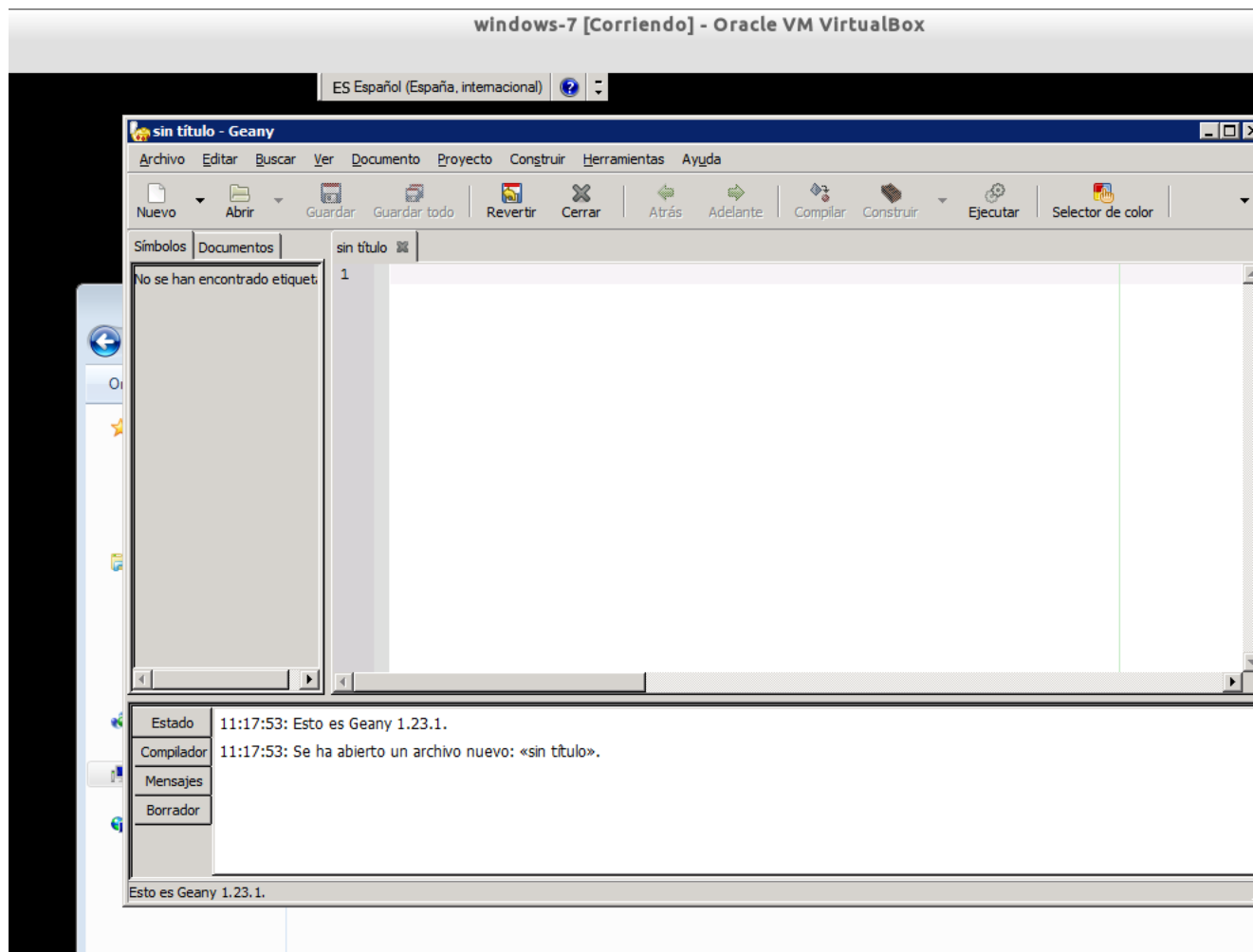


RemoteApp

Programas añadidos para RemoteApp.



Iniciando el RemoteApp.



Capítulo 2. Clientes ligeros

Configuración del Servidor LTSP

Como ya sabemos tenemos que tener dos adaptadores de red en el servidor uno para el acceso a internet, y otro para la red interna.

Lo primero que yo hago es establecer una ip estática al adaptador de red interna en mi caso eth1:

```
profesor@profesor-VirtualBox:~$ cat /etc/network/interfaces
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback
auto eth1
iface eth1 inet static
address 192.168.1.1
netmask 255.255.255.0
broadcast 192.168.1.255
gateway 192.168.1.1
```

Por otro lado, nos cambiamos el hostname de nuestro servidor:

```
profesor@profesor-VirtualBox:~$ cat /etc/hostname
ruentes
```

También establecemos un dominio:

```
profesor@profesor-VirtualBox:~$ cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 127.0.1.1

search invitados
domain leon
```

Instalamos ahora el LTSP en nuestro servidor:

```
root@profesor-VirtualBox:/home/profesor# apt-get install openssh-server ltsp-server-standalone
```

Ahora creamos la imagen LTSP, lo cual nos llevará un buen rato:

```
root@profesor-VirtualBox:/home/profesor# ltsp-build-client
```

Modificamos ahora el archivo NetworkManager.conf dándole el valor true donde pone false:

```
profesor@profesor-VirtualBox:~$ cat /etc/NetworkManager/NetworkManager.conf
[main]
plugins=ifupdown,keyfile
dns=dnsmasq

[ifupdown]
managed=true
profesor@profesor-VirtualBox:~$
```

Como en mi caso no he seguido la configuración por defecto que aconseja la página oficial de LTSP, y he querido utilizar una configuración determinada para el adaptador de red interna, tenemos que comunicárselo a los archivos de configuración LTSP, en este caso dhcp.conf el cual está dentro de /etc/ltsp. Lo establecemos de la siguiente manera:

```
profesor@profesor-VirtualBox:~$ cat /etc/ltsp/dhcpd.conf
#
# Default LTSP dhcpd.conf config file.
#

authoritative;

subnet 192.168.1.0 netmask 255.255.255.0 {
    range 192.168.1.20 192.168.1.250;
    option domain-name "example.com";
    option domain-name-servers 192.168.1.1;
    option broadcast-address 192.168.1.255;
    option routers 192.168.1.1;
    # next-server 192.168.0.1;
    # get-lease-hostnames true;
    option subnet-mask 255.255.255.0;
    option root-path "/opt/ltsp/i386";
    if substring( option vendor-class-identifier, 0, 9 ) = "PXEClient" {
        filename "/ltsp/i386/pxelinux.0";
    } else {
        filename "/ltsp/i386/nbi.img";
    }
}
```

Luego modificamos también el fichero `/etc/default/isc-dhcp-server` en el cual pondremos el adaptador de red destinado al LTSP en mi caso el `eth1`:

```
profesor@profesor-VirtualBox:~$ cat /etc/default/isc-dhcp-server
# Defaults for isc-dhcp-server initscript
# sourced by /etc/init.d/isc-dhcp-server
# installed at /etc/default/isc-dhcp-server by the maintainer scripts

#
# This is a POSIX shell fragment
#
# Path to dhcpd's config file (default: /etc/dhcp/dhcpd.conf).
#DHCPD_CONF=/etc/dhcp/dhcpd.conf
#
# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
#DHCPD_PID=/var/run/dhcpd.pid
#
# Additional options to start dhcpd with.
# Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""
#
# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
# Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACES="eth1"
```

Para ir acabando con el servidor lo que hacemos es reiniciar el servicio DHCP con `/etc/init.d/isc-dhcp-server restart`

Por último reiniciamos el servicio `tftpd-hpa` con `/etc/init.d/tftpd-hpa restart`

Sería interesante también crearnos unos usuarios, los cuales usaran los clientes para iniciar el sistema en red:



Configuración del cliente

Lo único que tenemos que hacer para la máquina cliente, es configurar una máquina virtual en red interna, la cual no tenga dispositivos de almacenamiento, de esta manera iniciará automáticamente por red gracias al servidor LTSP

Comprobación final

Para hacer la comprobación final he hecho un video donde podemos observar como funciona correctamente:

Video LTSP [<http://www.youtube.com/watch?v=AD4xz2fUgMs>]

Capítulo 3. Acceso remoto SSH

1. Linux

Preparativos

Configuramos las máquinas clientes y servidor tal y como nos dice el ejercicio. Usaremos:

- Un servidor Xubuntu
- Un cliente Opensuse
- Un cliente Windows

Instalación básica

En el servidor linux instalaremos la herramienta SSH para el mismo.

Haremos lo mismo para el cliente Opensuse, esta vez, le instalaremos la herramienta para cliente.

En el cliente windows usaremos el PuTTY, el cual solo hace falta ejecutar.

Ahora probaremos a conectarnos al servidor desde el cliente linux:

```
guzman@ssh-client1:~> ssh remoteuser1@172.16.109.14
The authenticity of host '172.16.109.14 (172.16.109.14)' can't be established.
ECDSA key fingerprint is 95:a9:75:88:ed:86:f8:f0:f9:e0:96:c3:8a:22:7f:e1.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': yes
Warning: Permanently added '172.16.109.14' (ECDSA) to the list of known hosts.
remoteuser1@172.16.109.14's password:

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Welcome to Ubuntu 13.04 (GNU/Linux 3.8.0-19-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '13.10' available.
Run 'do-release-upgrade' to upgrade to it.

remoteuser1@ssh-server:~$
```

Luego probaremos a conectarnos con la herramienta PuTTY desde el cliente windows:

```
login as: remoteuser1
remoteuser1@172.16.109.14's password:
Welcome to Ubuntu 13.04 (GNU/Linux 3.8.0-19-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '13.10' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Nov  5 10:40:12 2013 from 172.16.109.114
remoteuser1@ssh-server:~$
```

Crearemos un clave mediante el comando ssh-keygen:


```
guzman@ssh-server:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/guzman/.ssh/id_rsa):
Created directory '/home/guzman/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/guzman/.ssh/id_rsa.
Your public key has been saved in /home/guzman/.ssh/id_rsa.pub.
The key fingerprint is:
90:3a:45:a8:ad:ac:ab:9a:5c:c6:f0:58:94:46:5c:da guzman@ssh-server
The key's randomart image is:
+--[ RSA 2048]-----+
|    .o.    |
| ..=. .   |
|  B E+    |
| + .o .   |
| o oo  S  |
|  o .     |
| o =      |
|o.o       |
|Bo        |
+-----+
guzman@ssh-server:~$
```

Personalización del prompt Bash

Aquí simplemente lo que haremos es modificar el archivo bash para que se cambie el color del prompt cuando un cliente se conecte.

Aquí estamos modificando el archivo:

```
GNU nano 2.2.6 Archivo: /home/guzman/.bashrc

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

if [ -n "$SSH_CLIENT" ]; then text=" ssh"
fi
export PS1='\[\e[0;31m\]\u@\h:\w${text}$\[\e[m\] '
```

Ahora comprobamos su funcionamiento:

```
guzman@ssh-client1:~
Archivo Edición Pestañas Ayuda
guzman@ssh-client1:~$ ssh remoteuser1@ssh-server
Welcome to Ubuntu 13.04 (GNU/Linux 3.8.0-19-generic i686)

 * Documentation:  https://help.ubuntu.com/

225 packages can be updated.
144 updates are security updates.

New release '13.10' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Nov 12 09:26:39 2013 from 172.16.109.214
remoteuser1@ssh-server:~ ssh$
```

Autenticación mediante claves públicas

Empezaremos por generar una clave publica

```
guzman@ssh-client1:~  
Archivo Edici3n Pestaas Ayuda  
guzman@ssh-client1:~> ssh-keygen -t rsa  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/guzman/.ssh/id_rsa):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/guzman/.ssh/id_rsa.  
Your public key has been saved in /home/guzman/.ssh/id_rsa.pub.  
The key fingerprint is:  
1e:9b:08:bb:fb:c3:fc:a7:e3:f8:32:60:e9:f0:5e:c4 guzman@ssh-client1  
The key's randomart image is:  
+--[ RSA 2048]-----+  
|  
|  
|.E S  
|. ++ o +  
|=+o +  
|oo*.. .  
|.+ooB=+  
+-----+  
guzman@ssh-client1:~>
```

Ahora asociaremos la clave creada a el usuario deseado:

Una vez hecho lo anterior podemos comprobar, que `remoteuser1` en nuestro caso, no pide contraseña al iniciar la conexión:

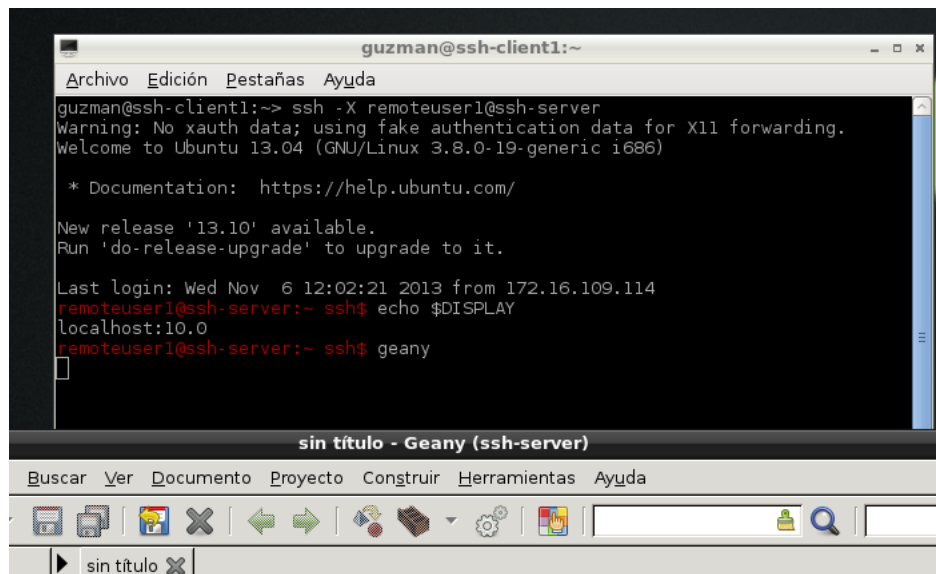
Uso de SSH como túnel para X

Un paso previo a seguir, es asegurarnos que la aplicación que queremos usar este solamente en el servidor y no en el cliente que solicitará los servicios.

Mediante ssh existe la posibilidad de ejecutar aplicaciones gráficas en el servidor y manejarlas y visualizarlas en el cliente. El servidor ssh deberá tener activada la redirección del protocolo X, es decir, deberá tener el siguiente parámetro en el archivo de configuración `/etc/ssh/ssh_config`:

```
X11Forwarding yes
```

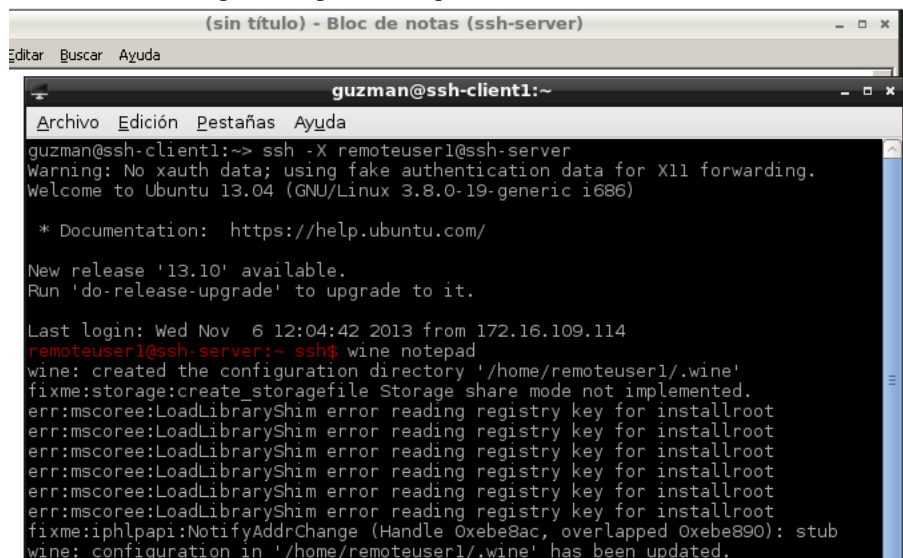
Una vez hecho lo anterior podremos conectarnos via SSH al servidor y ejecutar una aplicación que en el cliente no poseemos, en nuestro caso geany. Lo hacemos de la siguiente manera:



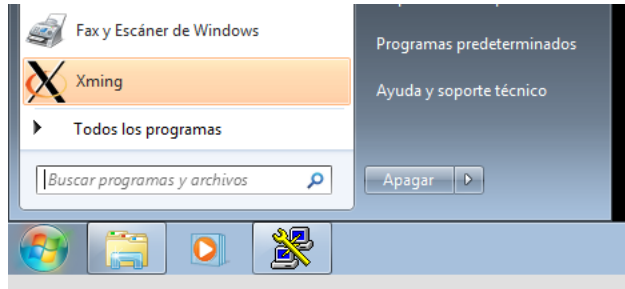
Aplicaciones Windows nativas

Podemos tener aplicaciones Windows nativas, instaladas en ssh-server mediante el emulador WINE, por eso lo primero que hacemos es instalarlo en el servidor.

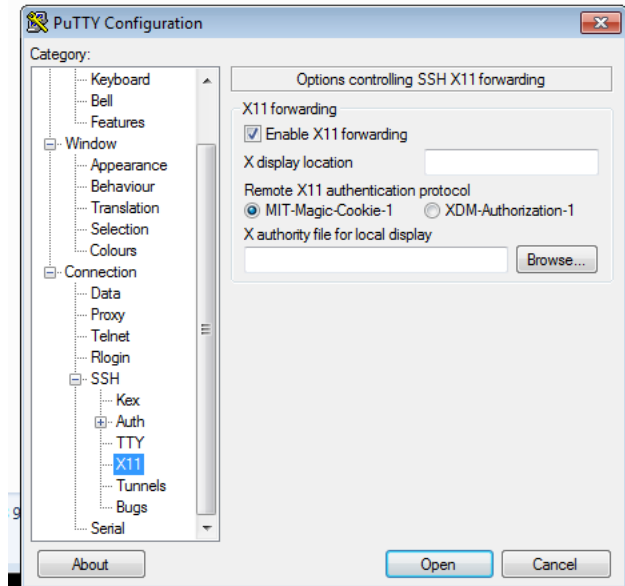
Instalamos una aplicación de Windows en el servidor SSH usando el emulador Wine. Por ejemplo en nuestro caso el notepad. Comprobamos que funciona en el servidor:



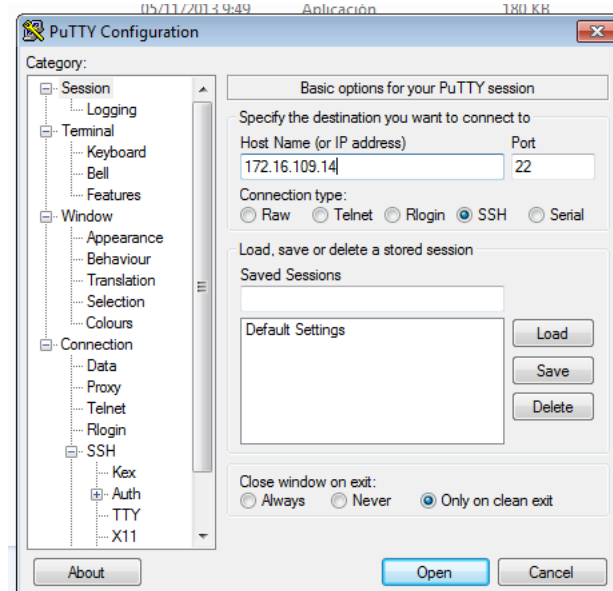
A continuación instalamos el Xming en nuestro cliente windows para que el proceso funcione correctamente:



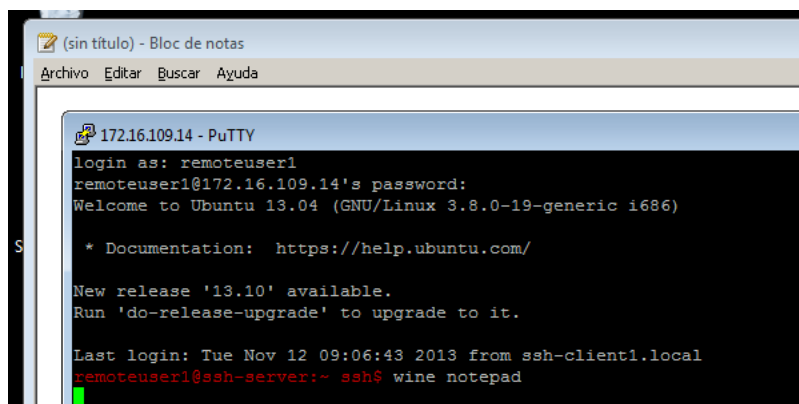
Ahora activaremos lo siguiente en el PuTTY:



Ahora nos conectamos desde el PuTTY al servidor:



Luego ejecutamos lo siguiente y vemos que funciona correctamente:



```

172.16.109.14 - PuTTY
login as: remoteuser1
remoteuser1@172.16.109.14's password:
Welcome to Ubuntu 13.04 (GNU/Linux 3.8.0-19-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '13.10' available.
Run 'do-release-upgrade' to upgrade to it.

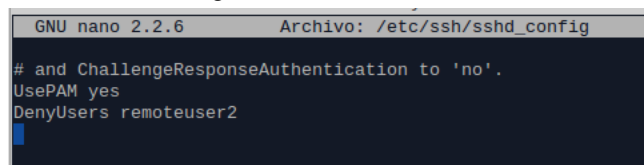
Last login: Tue Nov 12 09:06:43 2013 from ssh-client1.local
remoteuser1@ssh-server:~ ssh$ wine notepad

```

Restricciones de uso

- Restricción total:

En este caso lo que haremos sera modificar el siguiente fichero y escribir lo siguiente para que el usuario remoteuser2 no tenga acceso:

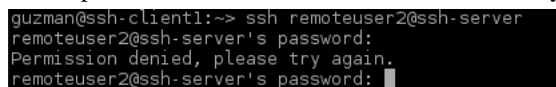


```

GNU nano 2.2.6 Archivo: /etc/ssh/sshd_config
# and ChallengeResponseAuthentication to 'no'.
UsePAM yes
DenyUsers remoteuser2

```

Ahora probaremos a conectarnos con remoteuser2 y nos muestra lo siguiente:



```

guzman@ssh-client1:~> ssh remoteuser2@ssh-server
remoteuser2@ssh-server's password:
Permission denied, please try again.
remoteuser2@ssh-server's password:

```

2. Windows

Preparativos

Configuramos las máquinas clientes y servidor tal y como nos dice el ejercicio. Usaremos:

- Un servidor Windows 2008 server
- Un cliente Debian
- Un cliente Windows

Instalación básica

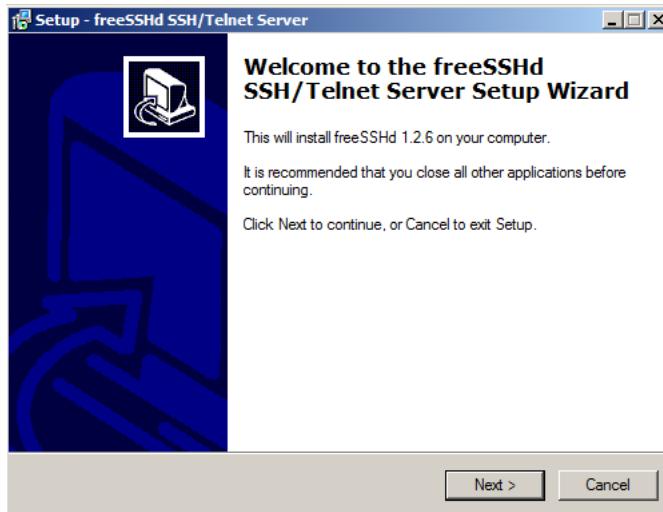
Para hacer la práctica en windows hemos seguido el tutorial propuesto en clase:

Tutorial de freeSSHd [<http://www.redeszone.net/windows/freesshd-para-windows-instalacion-y-manual-de-configuracion-de-freesshd-para-windows-servidor-ssh-y-sftp/>]

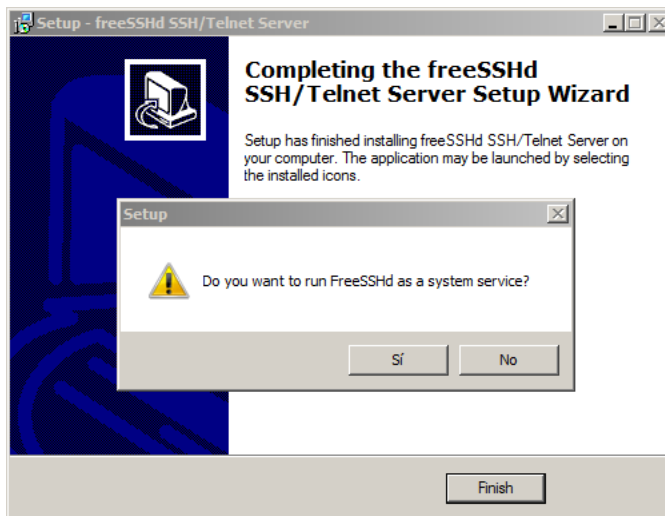
Lo primero que tenemos que hacer es bajarnos en la pagina oficial de freeSSHd el paquete adecuado:

name	version	size
freeSSHd.exe	1.2.6	764 KB

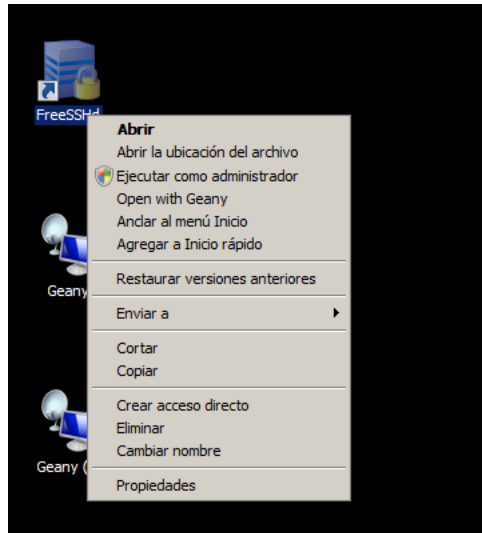
Una vez descargado lo ejecutamos y seguimos los pasos de instalación:



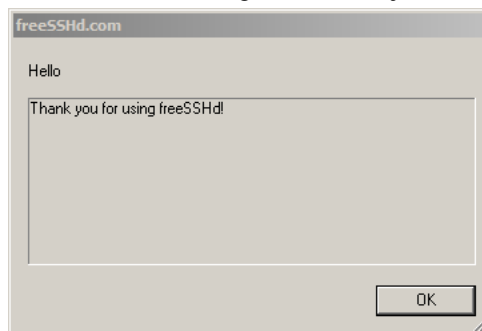
Aceptamos:



Lo ejecutamos como administrador:



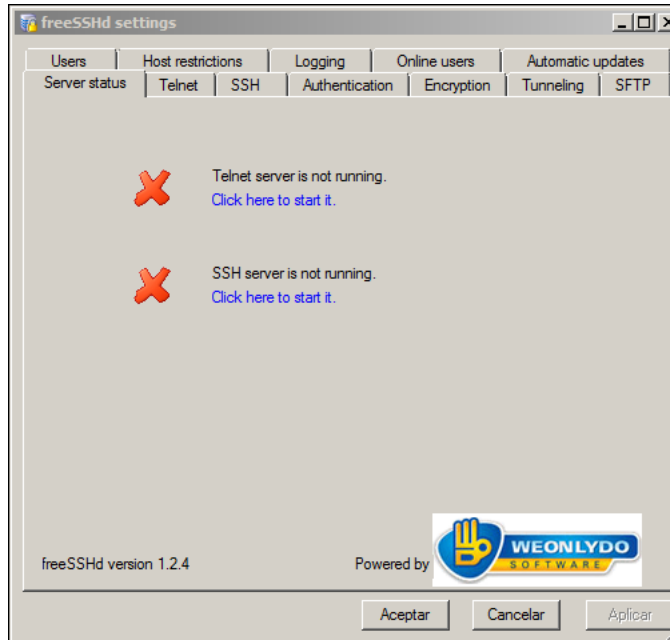
Nos debería salir el siguiente mensaje:



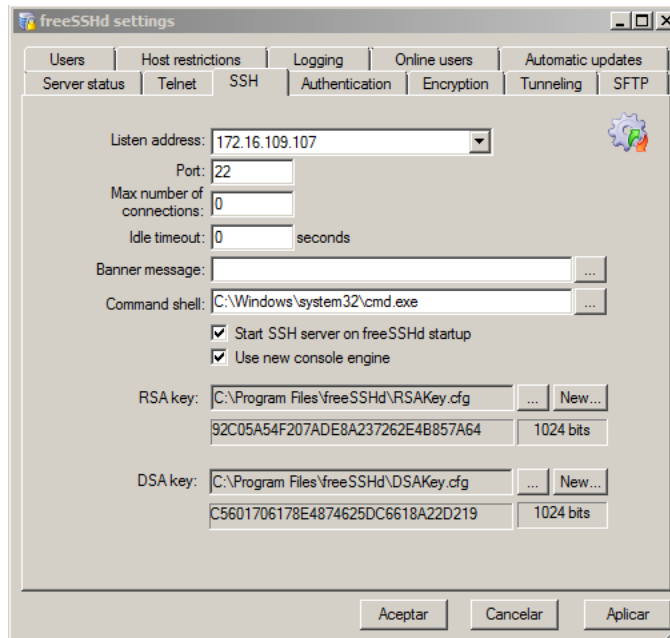
Una vez abierto deberíamos verlo en la esquina inferior derecha de nuestra pantalla:



Este sería el freeSSHd iniciado:

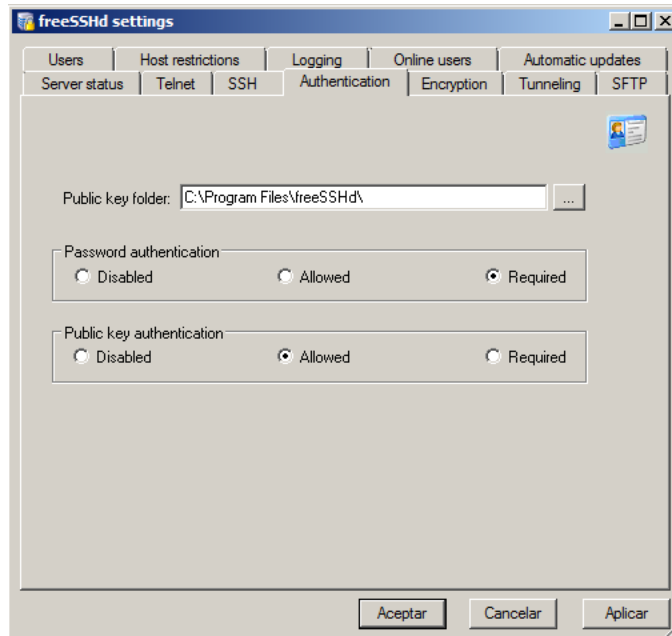


Lo configuraremos con la ip del servidor y el puerto por defecto el 22:

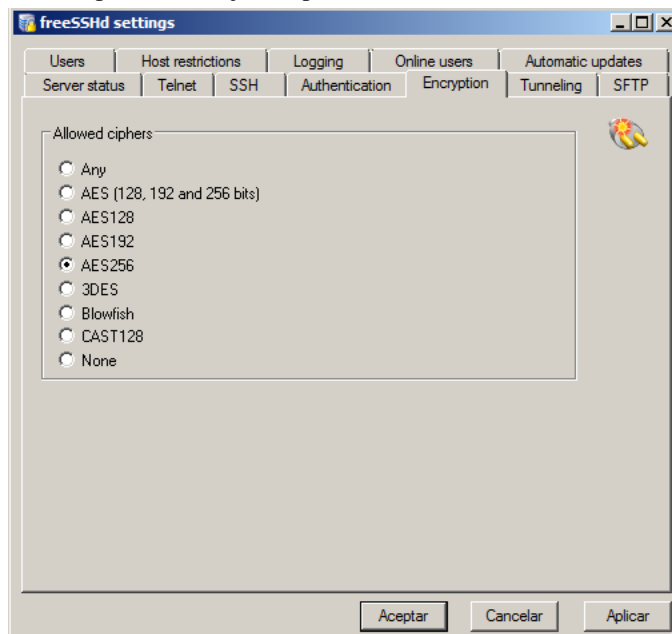


En algunos casos puede que el puerto nos de erro, a nosotros nos funcionó probando con diferentes puertos.

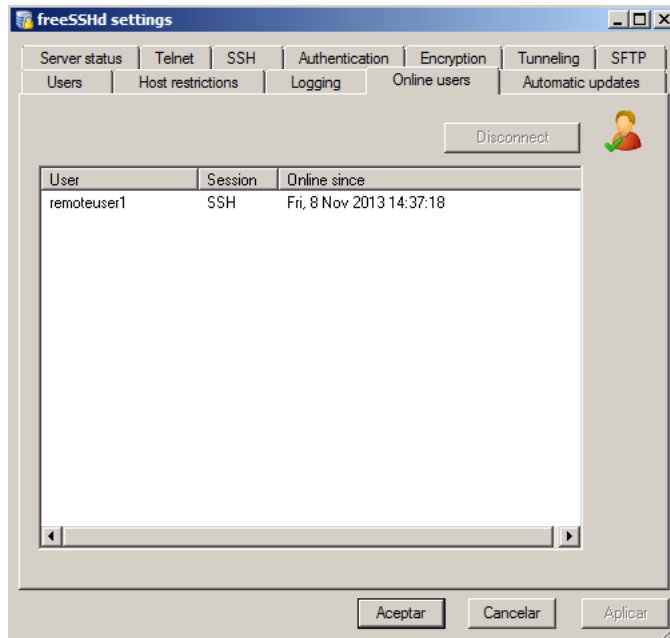
La autenticación la estableceremos de la siguiente manera, tal y como nos dice el tutorial:



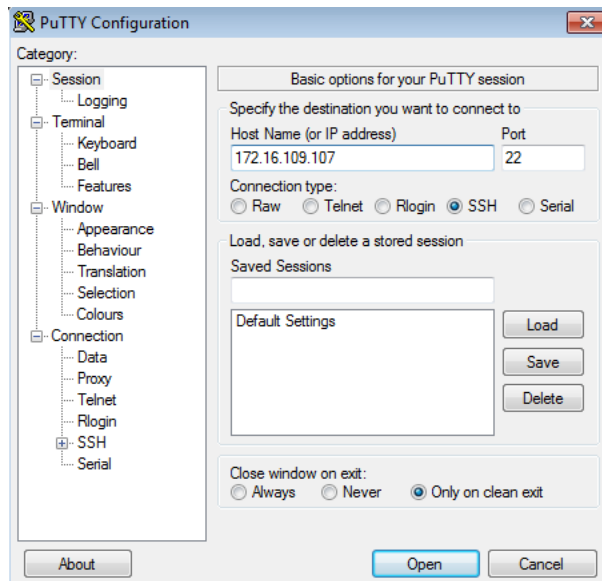
La encriptación la dejamos por defecto:



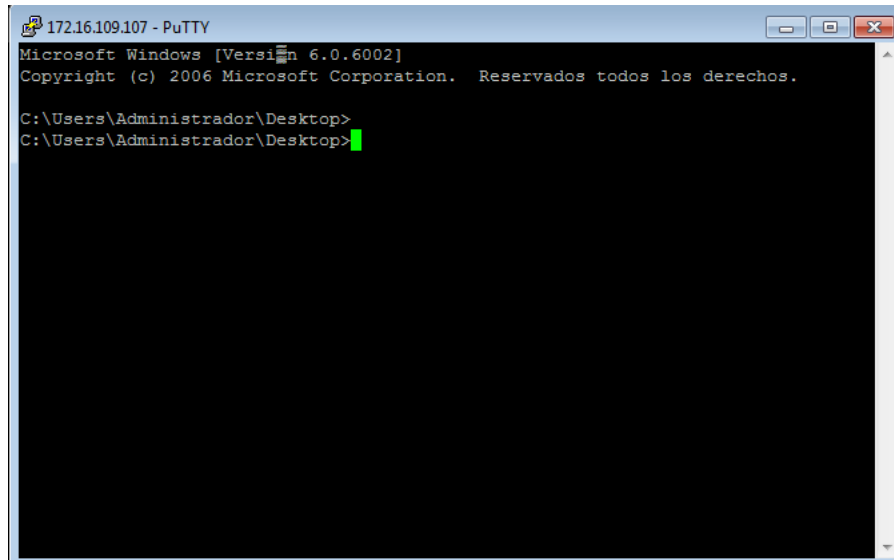
Ahora crearemos un usuario al cual se van a conectar los clientes, en nuestro caso remoteuser1



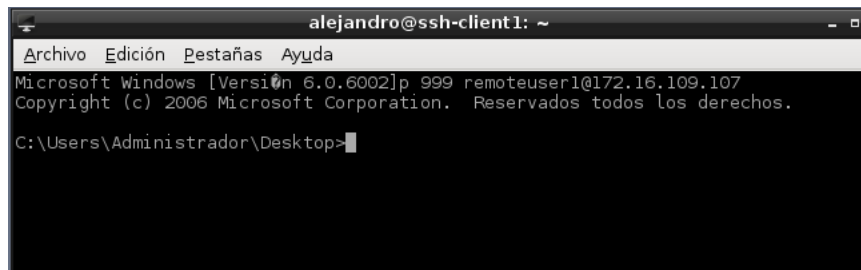
Ahora desde el cliente windows, ejecutaremos el programa PuTTY que podemos descargar desde su página oficial, y le establecemos la ip del servidor y el puerto establecido en el mismo anteriormente:



Si todo esta bien se nos habrira una consola pidiendo el usuario y la contraseña y se podrá establecer la conexión como se ve a continuación:



Nos debería salir el siguiente mensaje:



Ahora nos conectamos a través del cliente debian, pero por el puerto 999 ya que con el de por defecto nos daba error:

Aquí vemos el contenido del archivo `known_hosts`:

