

# **Administración de Sistemas Informáticos 2013-2014**

**Aitor Oms Pérez**

---

# **Administración de Sistemas Informáticos 2013-2014**

Aitor Oms Pérez

fecha de publicación 19/11/2013

Este documento está en construcción. Tiene fallos que se van mejorando, y puede no estar actualizado. Si tienes alguna propuesta al respecto me la puedes hacer llegar, y si quieres colaborar en su desarrollo, estás invitado a participar.

---

# Tabla de contenidos

1. Acceso Remoto .....	1
VNC .....	1
Escritorio remoto .....	3
Servidor de terminales .....	5
RemoteApp .....	6
2. Clientes ligeros con LTSP .....	8
Configuración del Servidor LTSP .....	8
Configuración del Cliente LTSP .....	9
3. Acceso remoto SSH (Servidor Xubuntu y Debian) .....	11
Instalación básica .....	11
Personalización del prompt Bash .....	13
Autenticación mediante claves públicas .....	14
Uso de SSH como túnel para X .....	15
Aplicaciones Windows Nativas .....	17
Restricciones de uso .....	17

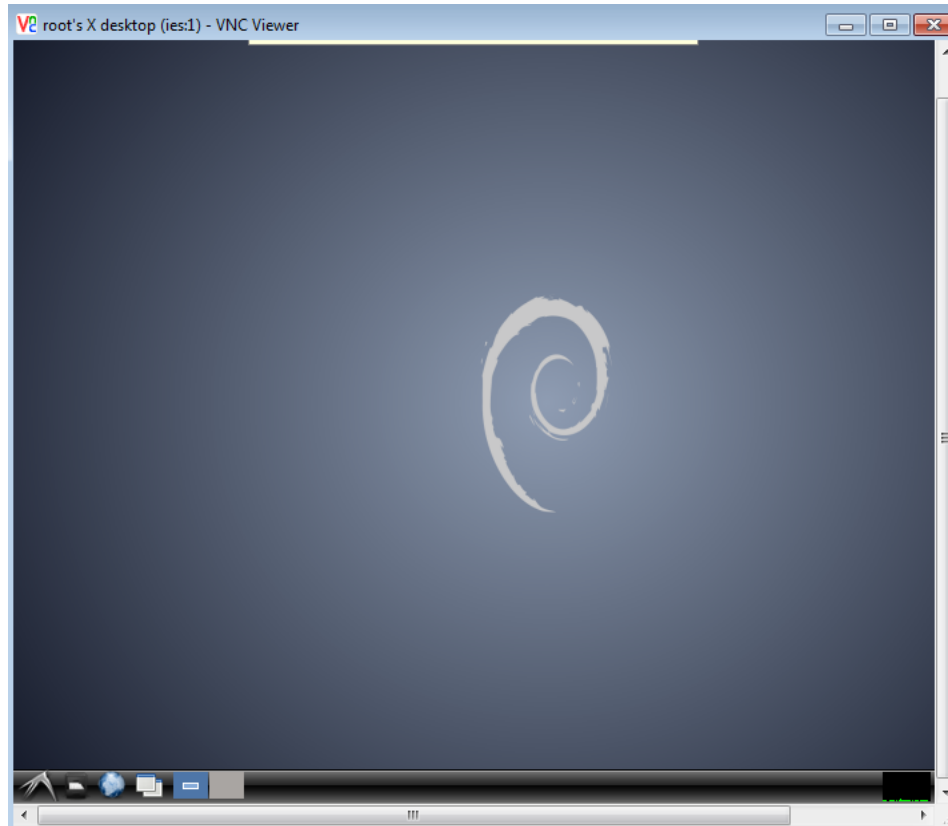
---

# Capítulo 1. Acceso Remoto

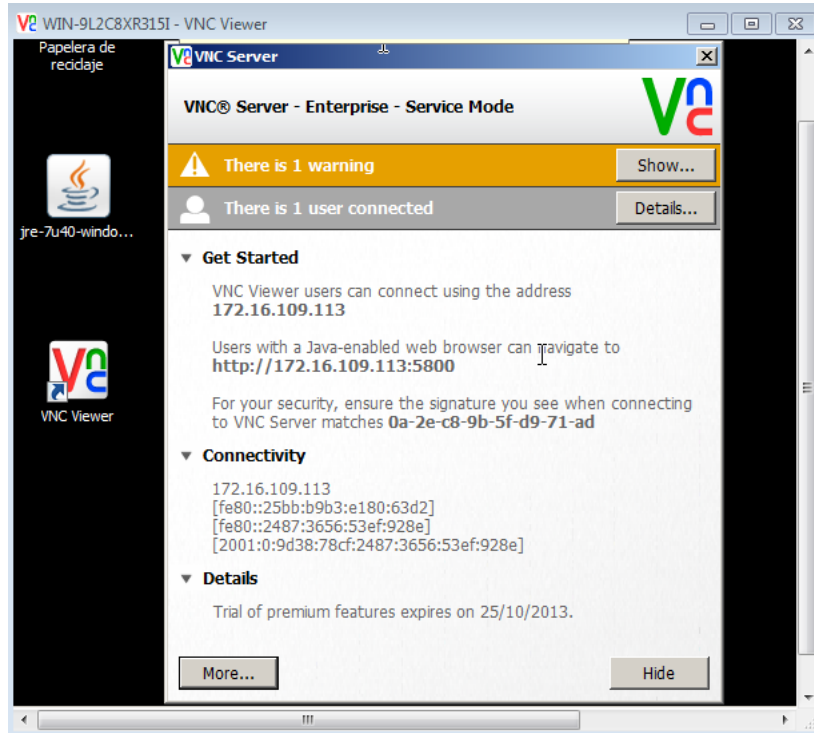
Lo primero que hicimos fue leer toda la documentación acerca de conexiones con escritorio remoto tanto en Debian como en Windows. Por una parte, mi compañero de grupo hizo la parte de Windows mientras que por otro lado yo hice la parte que corresponde con el Debian.

## VNC

Conexión desde Windows a Linux



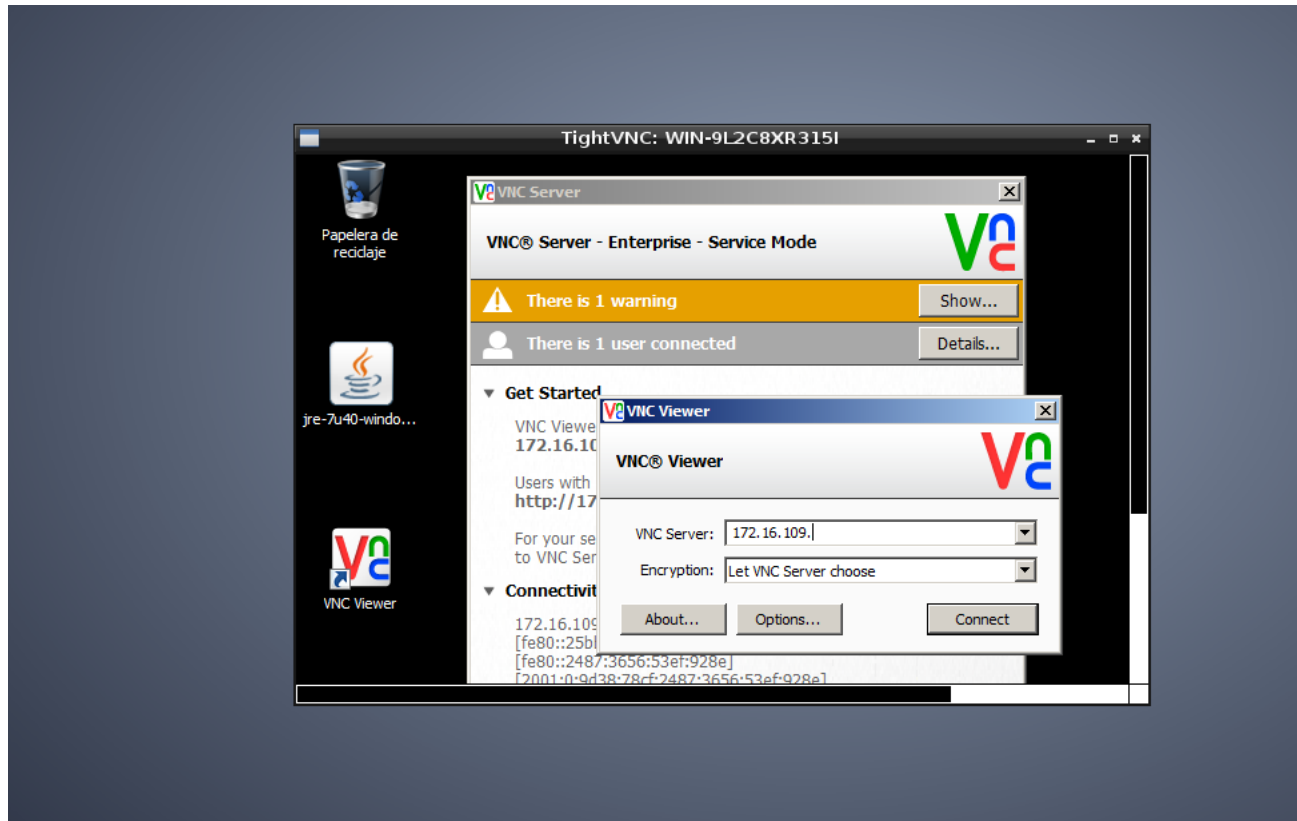
Conexión a Windows desde Windows



### Conexión a Debian desde Debian

```
root@ies:/# xtightvncviewer
Connected to RFB server, using protocol version 3.8
Enabling TightVNC protocol extensions
Performing standard VNC authentication
Authentication successful
Desktop name "root's X desktop (ies:1)"
VNC server default format:
 32 bits per pixel.
Least significant byte first in each pixel.
True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Warning: Cannot convert string "-*-helvetica-bold-r-*-*16-*-*-*-*-*" to typ
```

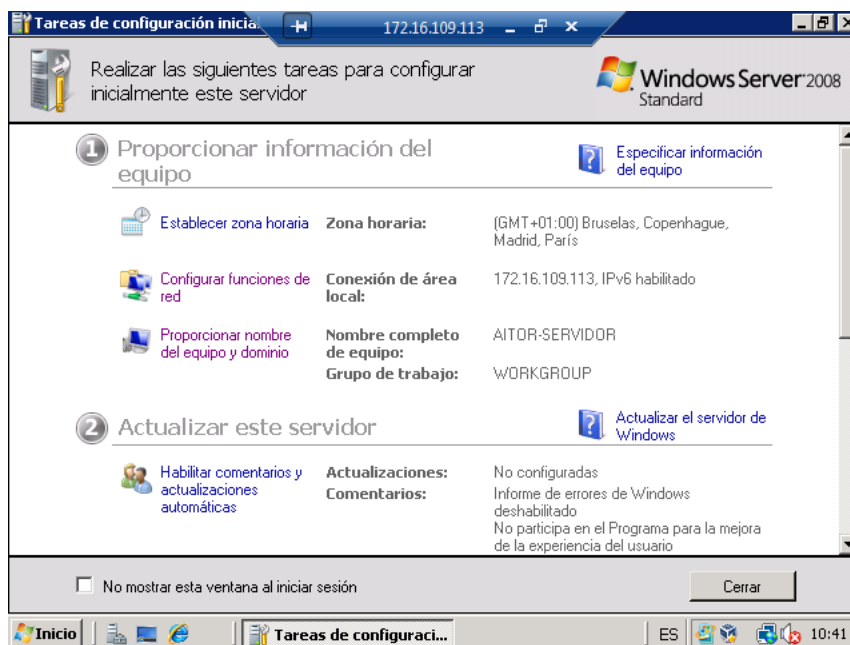
### Conexión a Windows desde Debian



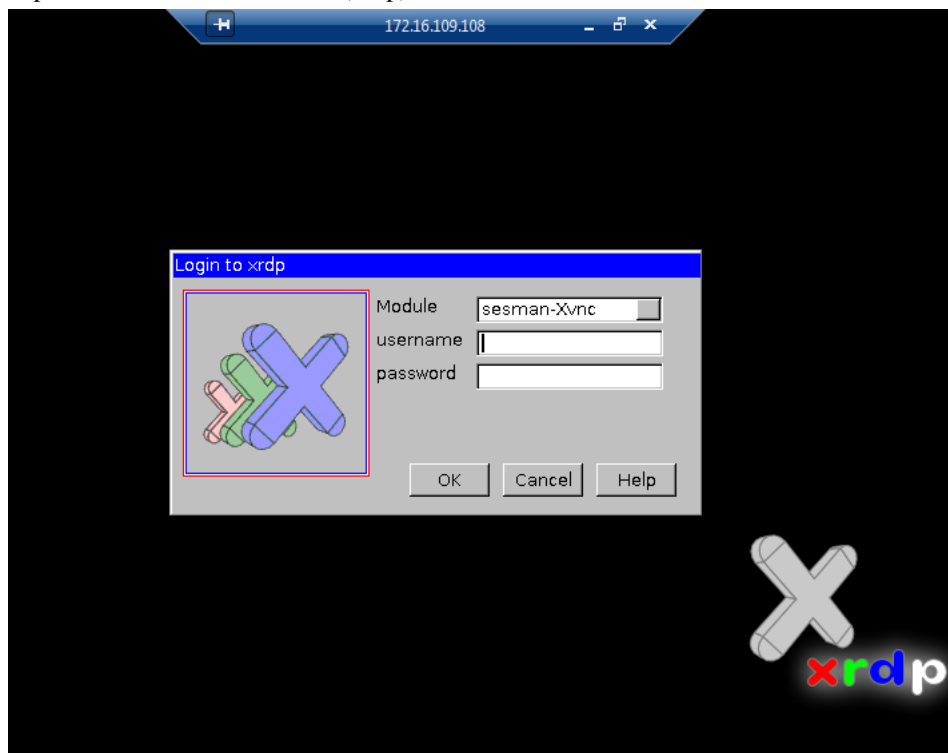
## Escritorio remoto

Para acceder desde cliente Windows 7 a Windows Server 2008 no hizo falta la descarga de ningún software ya que viene preinstalado. Lo único que tuvimos que hacer fue configurar el servidor para que permitiese el acceso remoto desde cualquier equipo aún teniendo un sistema operativo diferente. Para ello nos dirigimos en el servidor a la siguiente ruta( Panel de control->Sistema-> Configuración de acceso remoto) y en la pestaña acceso remoto seleccionamos la opción "Permitir las conexiones desde equipos que ejecuten cualquier versión de escritorio remoto. Luego simplemente para conectarnos hacemos click en inicio desde la máquina Windows 7 buscamos acceso remoto y ponemos la ip del servidor Windows 2008.

Conexión a Windows 2008 desde Windows 7

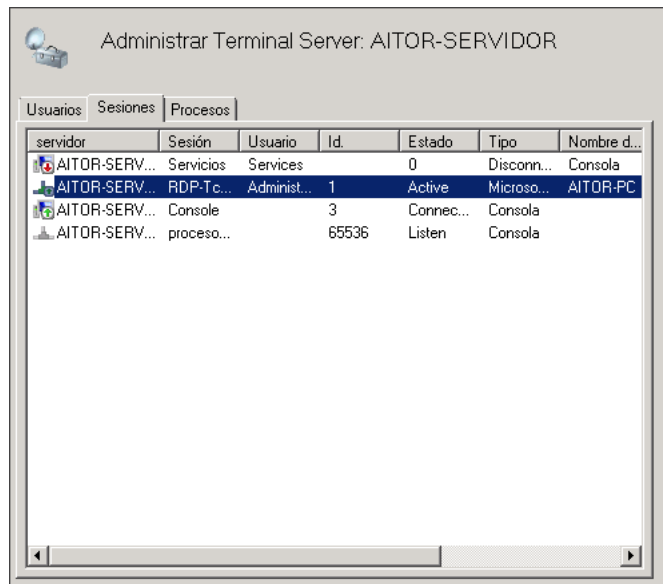


Para conectarnos desde Debian a Windows 2008 hizo falta la instalación en Debian de un paquete especial para interpretar el protocolo RDP usado por Windows. En Debian instalamos el siguiente paquete `xtightvncviewer` para la conexión de acceso remoto. Después de esto instalamos el paquete que permite el protocolo RDP de Windows (`xrdp`)

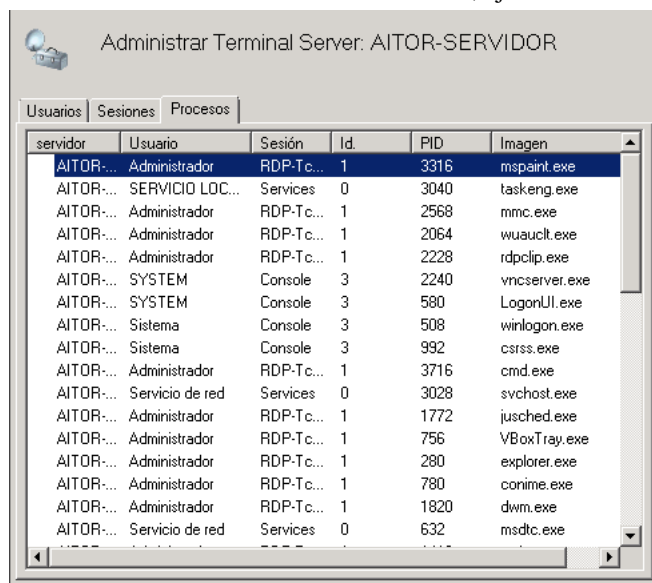


## Servidor de terminales

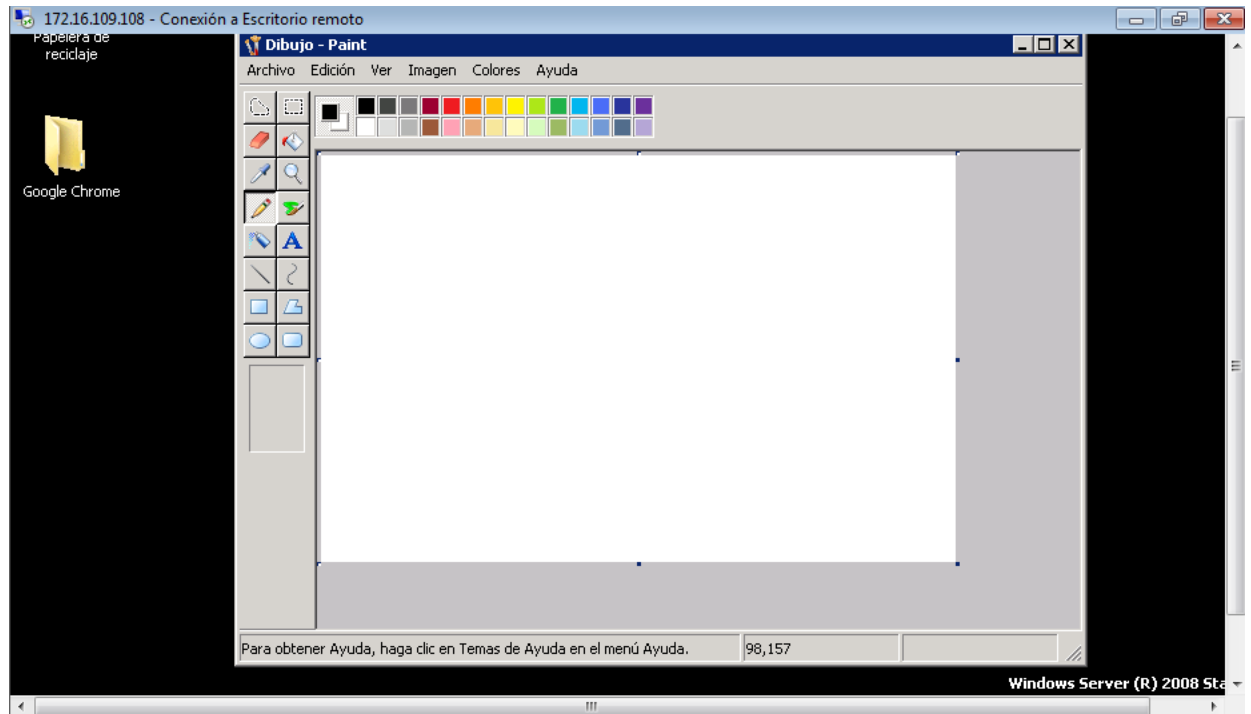
Para instalar Terminal Server, vamos a Administrador de Servidor->Funciones->Agregar Función->Terminal Server. Luego nos conectamos por Escritorio Remoto desde el equipo cliente al servidor, ya dentro de este último desde acceso remoto vamos a administrador de Terminal Services y comprobamos que el usuario esté conectado a través de dicho servicio.



Probamos desde el acceso remoto una aplicación del servidor y comprobamos que funciona a través del servicio de Terminal Services. En este caso, ejecutamos Paint



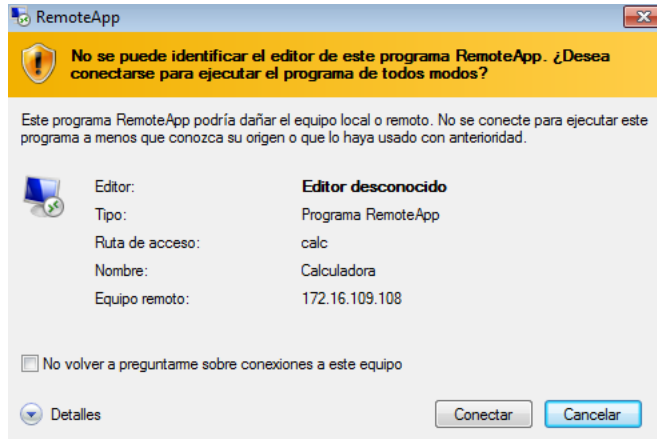




## RemoteApp

El RemoteApp no lo instalamos ya que viene preinstalado. En el Servidor Windows 2008 agregamos programas en este caso por ejemplo la calculadora. Creamos un archivo rdp como programa predeterminado la calculadora y con la ip del servidor del Windows 2008. Ese archivo lo exportamos al cliente y lo ejecutamos.





Por último observamos que la aplicación funciona correctamente



---

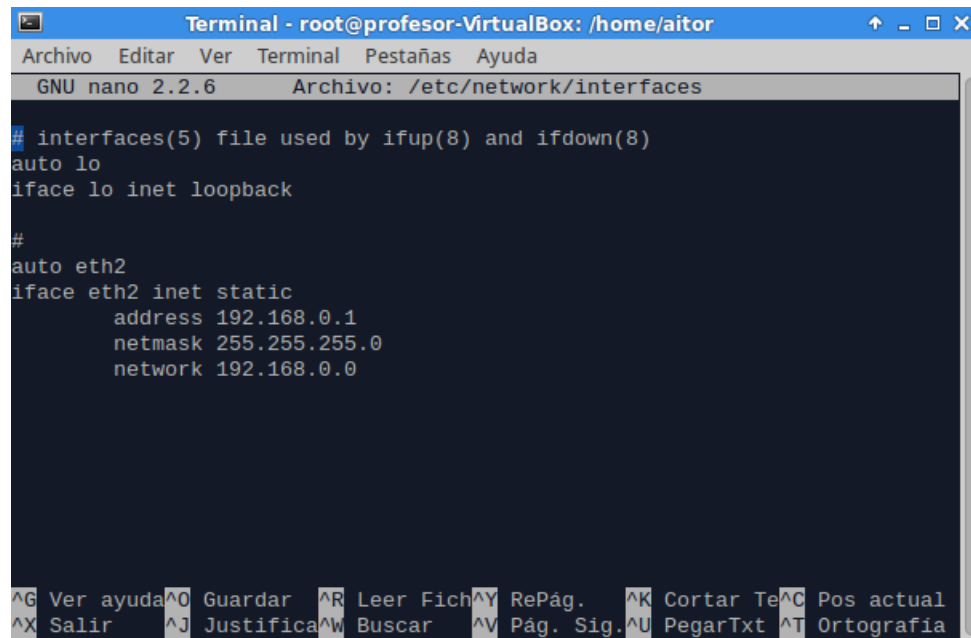
# Capítulo 2. Clientes ligeros con LTSP

Para la configuración y la instalación de un Servidor LTSP es necesario la consulta de una documentación. La documentación original para la instalación y configuración de un Servidor LTSP se encuentra en esta URL = <http://www.ltsp.org/>

## Configuración del Servidor LTSP

Lo primero que debemos hacer es añadir dos interfaces a nuestro servidor, una en modo adaptador-puente y otra en modo interna (Esta es la que usaremos para el conectarnos con el cliente ligero. Tras haber hecho esto, debemos crear un usuario con nuestro nombre y que posea como contraseña el DNI del Administrador. Tras haber hecho esto, configuramos la interfaz (Modo Interna) en `/etc/network/interfaces`.

Configuración de la Interfaz



```
Terminal - root@profesor-VirtualBox: /home/aitor
Archivo Editar Ver Terminal Pestañas Ayuda
GNU nano 2.2.6 Archivo: /etc/network/interfaces

# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

#
auto eth2
iface eth2 inet static
    address 192.168.0.1
    netmask 255.255.255.0
    network 192.168.0.0

^G Ver ayuda ^O Guardar ^R Leer Fich ^Y RePág. ^K Cortar Te ^C Pos actual
^X Salir ^J Justifica ^W Buscar ^V Pág. Sig. ^U PegarTxt ^T Ortografía
```

Una vez configurada la interfaz y habiendo comprobado que la configuración nueva funciona correctamente procedemos a instalar el paquete "ltsp-server-standalone" el cual nos permitirá tener los paquetes necesarios para el funcionamiento del Servidor LTSP. Tras haber instalado el servicio, configuramos el fichero `/etc/default/isc-dhcp-server` poniendo como interfaz principal la que vayamos a usar para el Servidor LTSP.

```

Terminal - root@profesor-VirtualBox: /home/aitor
GNU nano 2.2.6 Archivo: /etc/default/isc-dhcp-server Modificado

#DHCPCD_CONF=/etc/dhcp/dhcpd.conf

# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
#DHCPCD_PID=/var/run/dhcpd.pid

# Additional options to start dhcpd with.
# Don't use options -cf or -pf here; use DHCPCD_CONF/ DHCPCD_PID inst$
#OPTIONS=""

# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
# Separate multiple interfaces with spaces, e.g. "eth0 eth2".
INTERFACES="eth2"

^G Ver ayuda ^O Guardar ^R Leer Fich ^Y RePág. ^K Cortar Tex ^C Pos actual
^X Salir ^J Justificar ^W Buscar ^V Pág. Sig. ^U PegarTxt ^T Ortografia
  
```

Después de haber configurado esto, también debemos de configurar otro fichero en el servidor = /etc/ltsp/dhcpd.conf

```

Terminal - root@profesor-VirtualBox: /home/aitor
GNU nano 2.2.6 Archivo: /etc/ltsp/dhcpd.conf

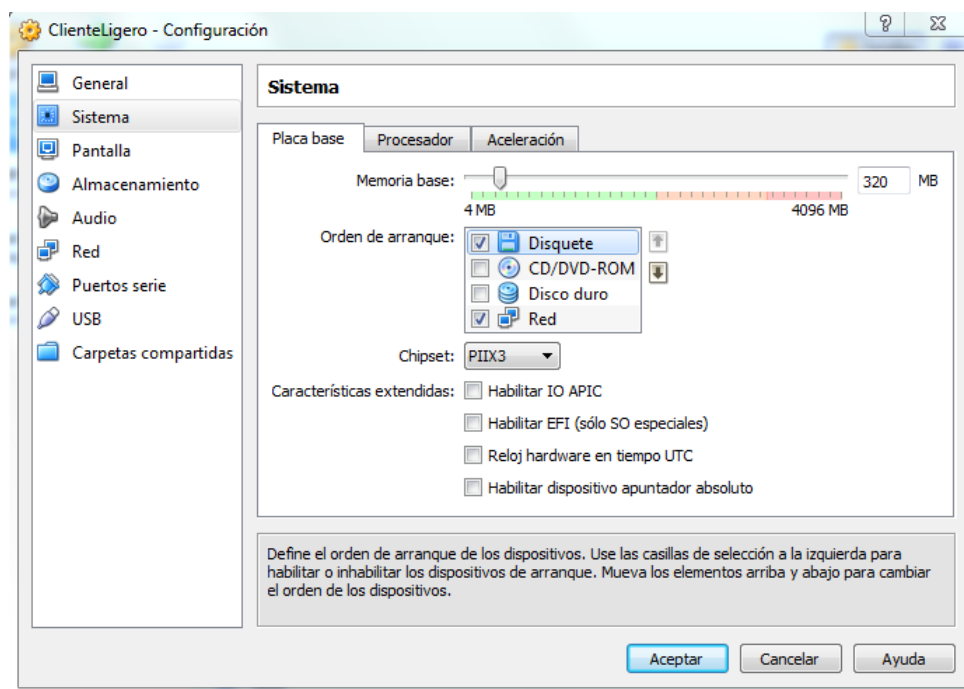
#
# Default LTSP dhcpd.conf config file.
#
authoritative;

subnet 192.168.0.0 netmask 255.255.255.0 {
    range 192.168.0.20 192.168.0.250;
    option domain-name "example.com";
    option domain-name-servers 192.168.0.1;
    option broadcast-address 192.168.0.255;
    option routers 192.168.0.1;
    # 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 {
        [ 22 líneas leídas ]
    }
}
^G Ver ayuda ^O Guardar ^R Leer Fich ^Y RePág. ^K Cortar Tex ^C Pos actual
^X Salir ^J Justificar ^W Buscar ^V Pág. Sig. ^U PegarTxt ^T Ortografia
  
```

Tras haber configurado todos los ficheros de el servidor, solamente queda crear la imagen que el usuario utilizará, para ello usamos el comando ltsp-build-client

## Configuración del Cliente LTSP

Tras haber configurado el servidor, el siguiente paso es la configuración de la máquina o equipo del cliente. Para configurar la máquina del cliente debemos crearla sin ningún dispositivo de almacenamiento, ya que, arrancará desde la red.



Para que el cliente pueda arrancar desde la red, es necesario que cargue la orden de arranque desde un disquete configurado para ello. Hay que apreciar que el Cliente LTSP no requiere configuración de red ya que el Servidor LTSP le proporciona una dirección ip dentro del rango DHCP que hayamos configurado. Para la creación del disquete accedemos a la siguiente página = <http://rom-o-matic.net/gpxe/gpxe-1.0.1/contrib/rom-o-matic/>



ROM-o-matic.net dynamically generates gPXE images

If this site helps you, please [Donate](#) to help us continue to make Free, Open Source, State-of-the-art network booting software. Thanks!

To create an image:

1. Choose an output format:
2. Choose a NIC type:
3. ( optional — for binary ROM image format only )  
If you choose *Binary ROM image* as your output format, you must enter 4 hex digits below for *PCI VENDOR CODE* and *PCI DEVICE CODE* that match the NIC device for which you are making this image.  
Information on how to determine NIC PCI IDs may be found [here](#).  
PCI VENDOR CODE:  PCI DEVICE CODE:   
**Please note for ROM images:**
  - If you enter PCI IDs, we will attempt to determine the correct driver to support them, and will ignore any NIC type entered above.
  - gPXE does not support all possible PCI IDs for supported NICs.
4. Generate and download an image: [Get Image](#)

Finalmente el Cliente Ligero ya estaría listo para arrancar por la red a través del Servidor de Clientes Ligeros. Y aquí la comprobación de que todo a funcionado correctamente: <http://www.youtube.com/watch?v=4DPhYnqZUI4>

---

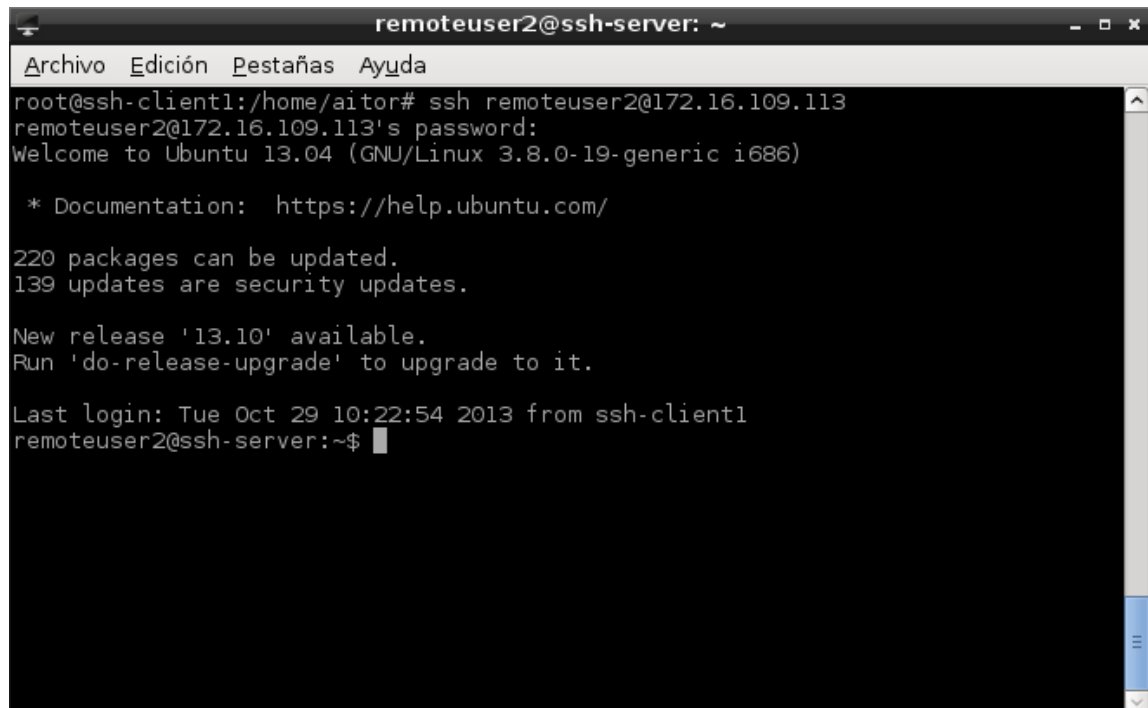
# Capítulo 3. Acceso remoto SSH (Servidor Xubuntu y Debian)

Para esta práctica usaremos como servidor un Xubuntu, como clientes, un Windows 7 y un Debian 7. También lo hicimos con un servidor Debian pero no lo hemos documentado puesto que no hay ningún cambio significativo en el proceso de uno y otro.

## Instalación básica

Lo primero que debemos hacer es instalar el Servidor SSH para ello hacemos `apt-get install openssh-server` en nuestro servidor Xubuntu.

Comprobamos el funcionamiento de la conexión SSH desde nuestro cliente Debian para ello escribimos en consola: `ssh remoteuser1@ipdelservidor`



```
remoteuser2@ssh-server: ~
Archivo Edición Pestañas Ayuda
root@ssh-client1:/home/aitor# ssh remoteuser2@172.16.109.113
remoteuser2@172.16.109.113's password:
Welcome to Ubuntu 13.04 (GNU/Linux 3.8.0-19-generic i686)

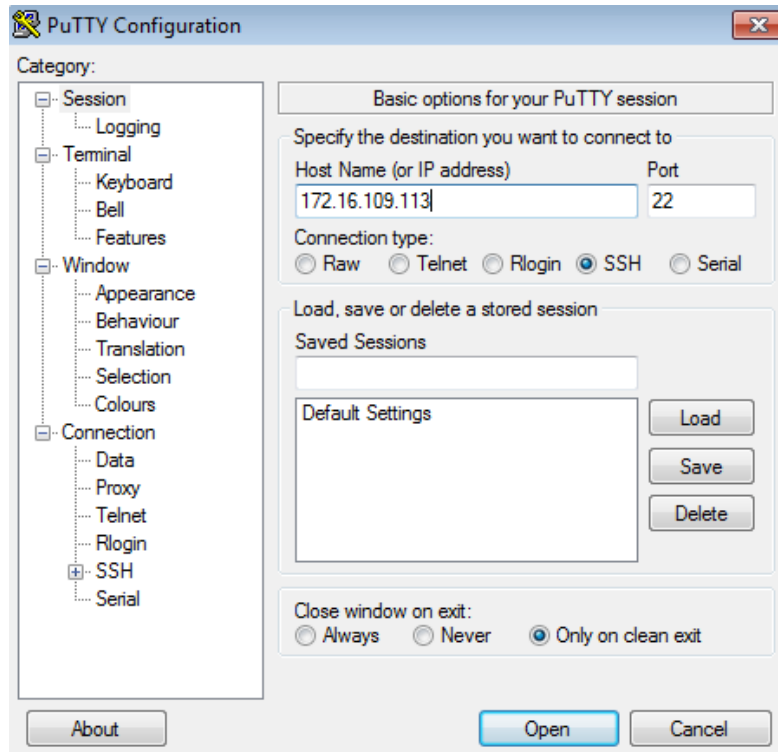
 * Documentation:  https://help.ubuntu.com/

220 packages can be updated.
139 updates are security updates.

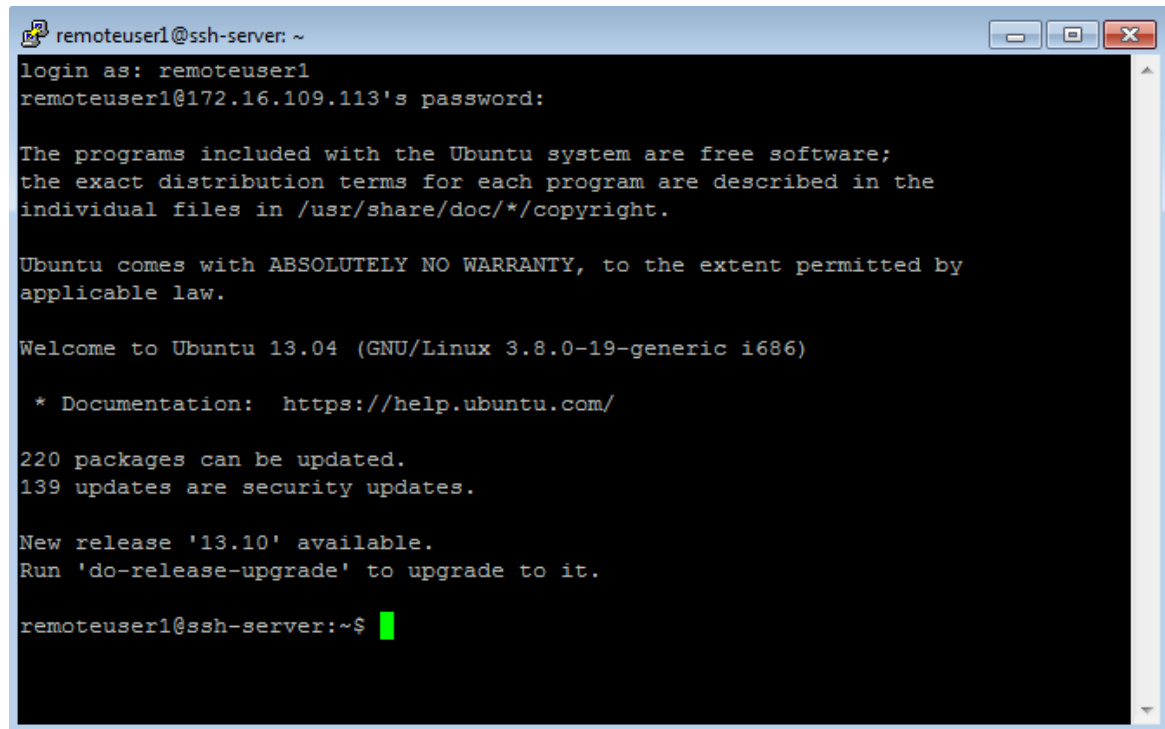
New release '13.10' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Oct 29 10:22:54 2013 from ssh-client1
remoteuser2@ssh-server:~$
```

Windows no tiene el paquete SSH instalado, para ello nos descargamos PUTTY, se trata de una aplicación que nos permite la conexión mediante este protocolo : <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>

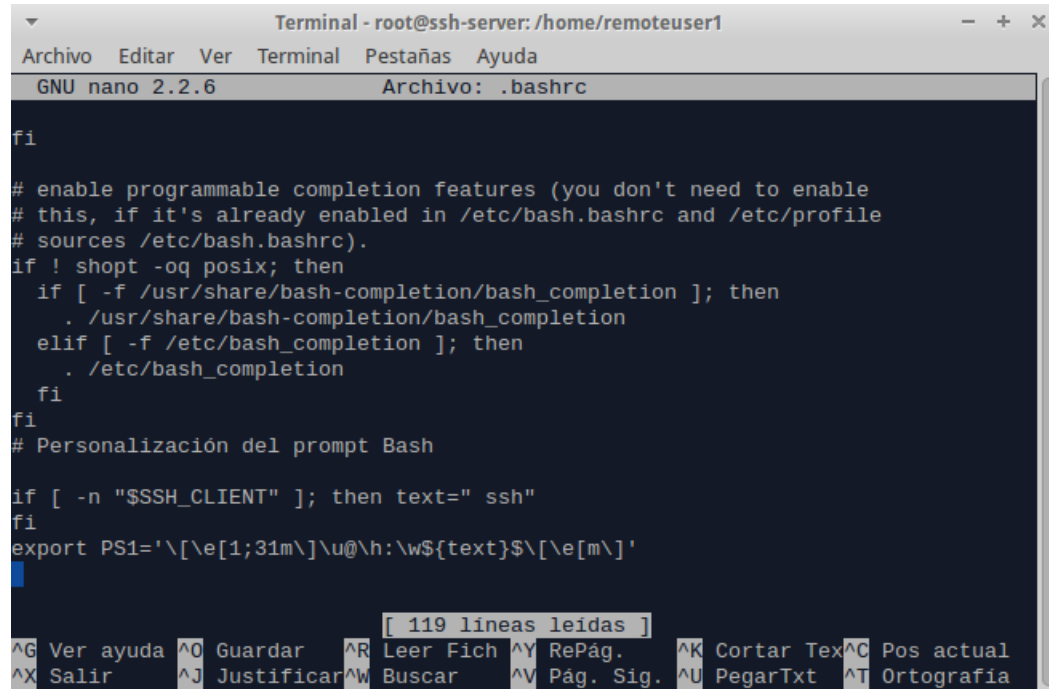


Abrimos PuTTY, luego en Host Name escribimos la Ip de nuestro servidor SSH y le damos click a la opción Open para conectarnos



## Personalización del prompt Bash

Para la configuración del Prompt de cada usuario modificamos el archivo oculto `.bashrc` de el usuario que queramos modificar. Este archivo se encuentra en el home de cada usuario y solamente hay que modificar la línea `export PS1='\[\e[1;31m\]....` donde `1;31m` es el parámetro del color



```
Terminal - root@ssh-server: /home/remotouser1
GNU nano 2.2.6 Archivo: .bashrc

fi

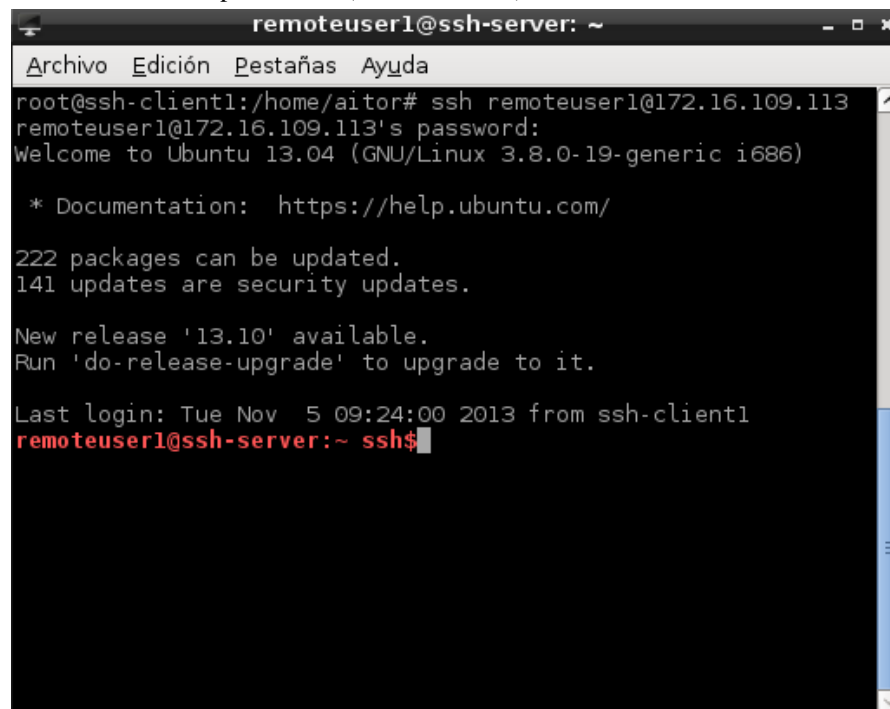
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion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

# Personalización del prompt Bash

if [ -n "$SSH_CLIENT" ]; then text=" ssh"
fi
export PS1='\[\e[1;31m\]\u@\h:\w${text}$\[\e[m\]'

[ 119 líneas leídas ]
^G Ver ayuda ^O Guardar ^R Leer Fich ^Y RePág. ^K Cortar Tex ^C Pos actual
^X Salir ^J Justificar ^W Buscar ^V Pág. Sig. ^U PegarTxt ^T Ortografia
```

Tras configurar el fichero `.bashrc` de cada usuario nos conectamos mediando SSH al servidor y comprobamos los cambios producidos (Cliente Debian)



```
remoteuser1@ssh-server: ~
Archivo Edición Pestañas Ayuda
root@ssh-client1:/home/aitor# ssh remoteuser1@172.16.109.113
remoteuser1@172.16.109.113's password:
Welcome to Ubuntu 13.04 (GNU/Linux 3.8.0-19-generic i686)

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

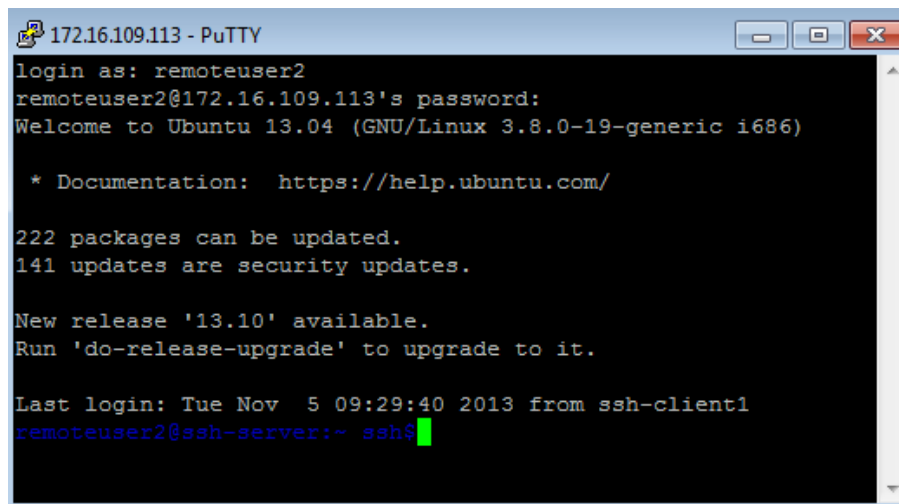
222 packages can be updated.
141 updates are security updates.

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

Last login: Tue Nov  5 09:24:00 2013 from ssh-client1
remoteuser1@ssh-server:~ ssh$
```



(Cliente Windows)



```
172.16.109.113 - PuTTY
login as: remoteuser2
remoteuser2@172.16.109.113's password:
Welcome to Ubuntu 13.04 (GNU/Linux 3.8.0-19-generic i686)

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

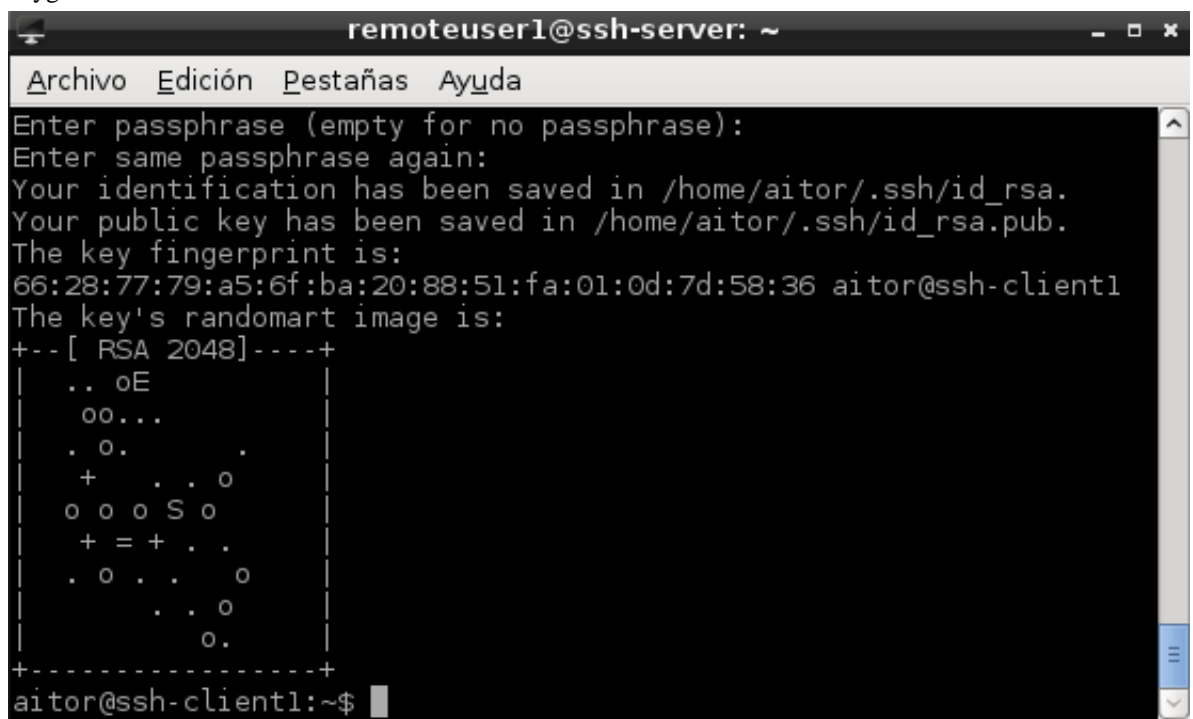
222 packages can be updated.
141 updates are security updates.

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

Last login: Tue Nov  5 09:29:40 2013 from ssh-client1
remoteuser2@ssh-server:~ ssh$
```

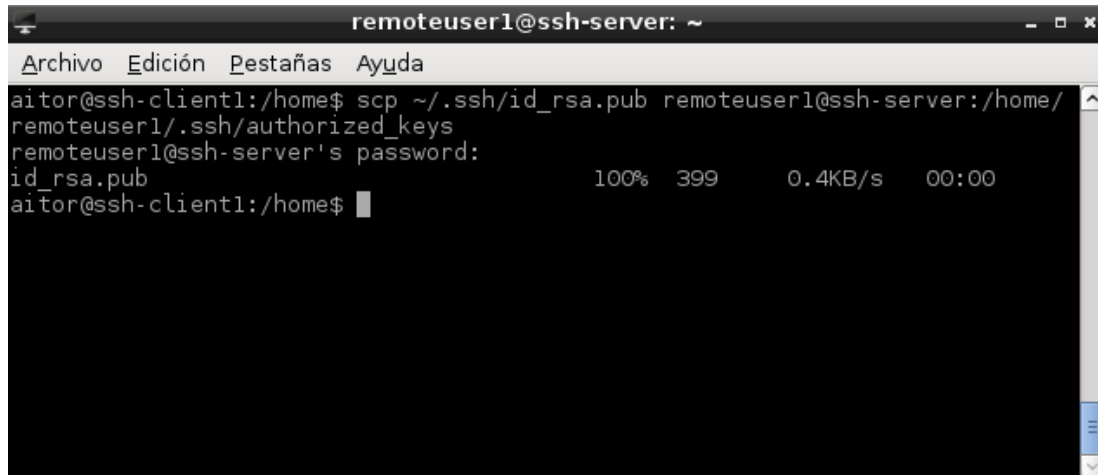
## Autenticación mediante claves públicas

Lo primero que debemos hacer es generar una nueva clave pública en el cliente para ello hacemos ssh-keygen en la consola.



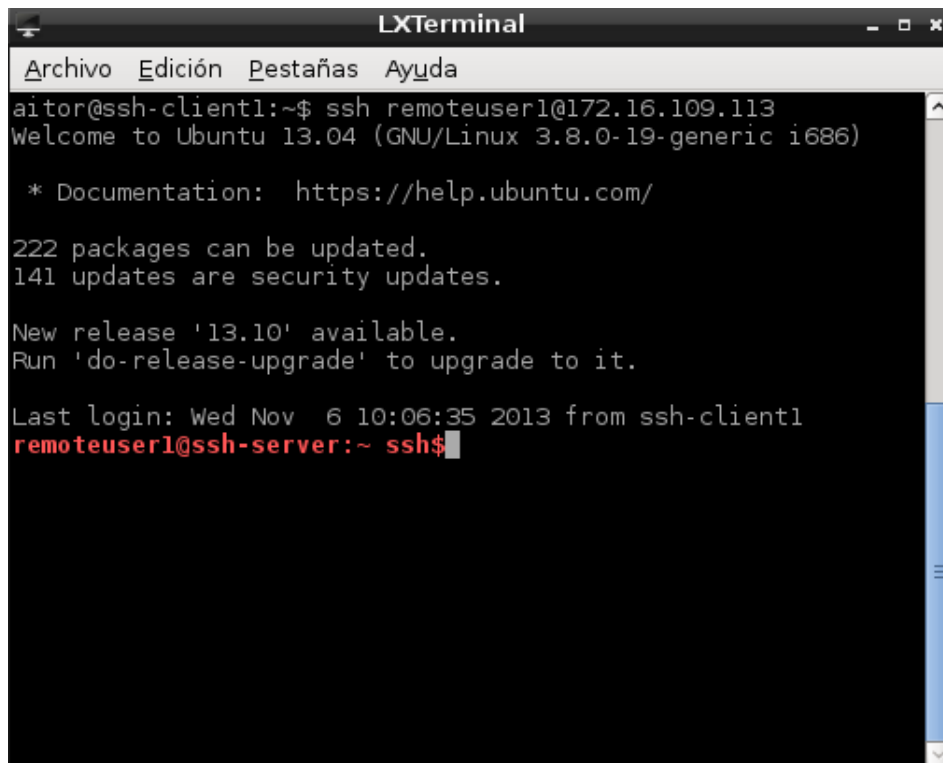
```
remoteuser1@ssh-server: ~
Archivo  Edición  Pestañas  Ayuda
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/aitor/.ssh/id_rsa.
Your public key has been saved in /home/aitor/.ssh/id_rsa.pub.
The key fingerprint is:
66:28:77:79:a5:6f:ba:20:88:51:fa:01:0d:7d:58:36 aitor@ssh-client1
The key's randomart image is:
+--[ RSA 2048 ]-----+
| .. oE                |
| oo...               |
| . o.                |
| + . . . o           |
| o o o S o           |
| + = + . .           |
| . o . . . o         |
| . . . o             |
| . . . o             |
| . . . o             |
+-----+
aitor@ssh-client1:~$
```

Tras haber generado la nueva clave pública se la enviamos al servidor y la introducimos en el fichero que se encuentra en `/.ssh/authorized_keys` del usuario que queramos que funcione mediante la autenticación de claves



```
remoteuser1@ssh-server: ~  
Archivo Edición Pestañas Ayuda  
aitor@ssh-client1:/home$ scp ~/.ssh/id_rsa.pub remoteuser1@ssh-server:/home/  
remoteuser1/.ssh/authorized_keys  
remoteuser1@ssh-server's password:  
id_rsa.pub                                100% 399      0.4KB/s   00:00  
aitor@ssh-client1:/home$
```

Después de haber hecho todo esto solo nos falta comprobar que funciona correctamente, para ello nos conectamos de forma normal a nuestro servidor SSH

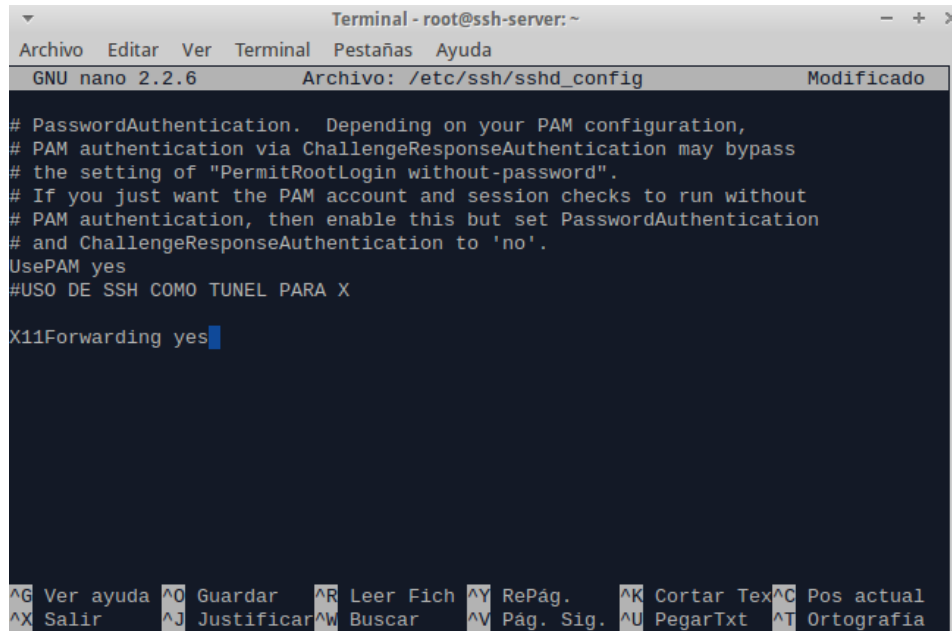


```
LXTerminal  
Archivo Edición Pestañas Ayuda  
aitor@ssh-client1:~$ ssh remoteuser1@172.16.109.113  
Welcome to Ubuntu 13.04 (GNU/Linux 3.8.0-19-generic i686)  
  
* Documentation:  https://help.ubuntu.com/  
  
222 packages can be updated.  
141 updates are security updates.  
  
New release '13.10' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Wed Nov  6 10:06:35 2013 from ssh-client1  
remoteuser1@ssh-server:~ ssh$
```

Como se puede apreciar en la imagen no ha sido necesario introducir la contraseña del usuario, es decir, se ha conectado de forma automática

## Uso de SSH como túnel para X

Para la utilización de SSH como túnel X configuraremos en el servidor el fichero `/etc/ssh/sshd_config` y le añadiremos la siguiente línea: `X11Forwarding yes`

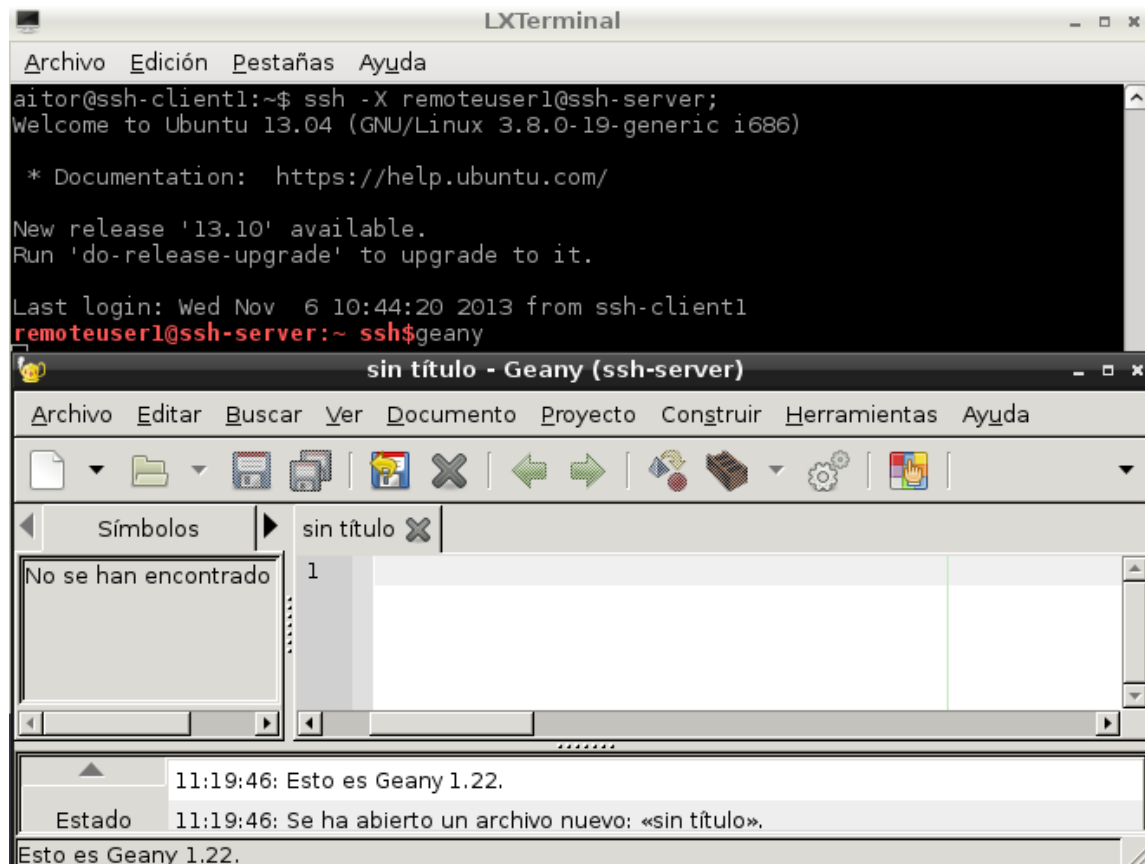


```
Terminal - root@ssh-server: ~
Archivo  Editar  Ver  Terminal  Pestañas  Ayuda
GNU nano 2.2.6      Archivo: /etc/ssh/sshd_config      Modificado

# PasswordAuthentication. Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin without-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to 'no'.
UsePAM yes
#USO DE SSH COMO TUNEL PARA X

X11Forwarding yes
```

Tras haber configurado ese archivo, nos conectamos desde el cliente a el servidor a través de tunel X de la siguiente forma: `ssh -X remoteuser1@ipservidor`; Y luego simplemente ejecutamos un programa que tenga el servidor, en este caso utilizaremos Geany como ejemplo.



```
LXTerminal
Archivo  Edición  Pestañas  Ayuda
aitor@ssh-client1:~$ ssh -X remoteuser1@ssh-server;
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: Wed Nov  6 10:44:20 2013 from ssh-client1
remoteuser1@ssh-server:~ ssh$ geany

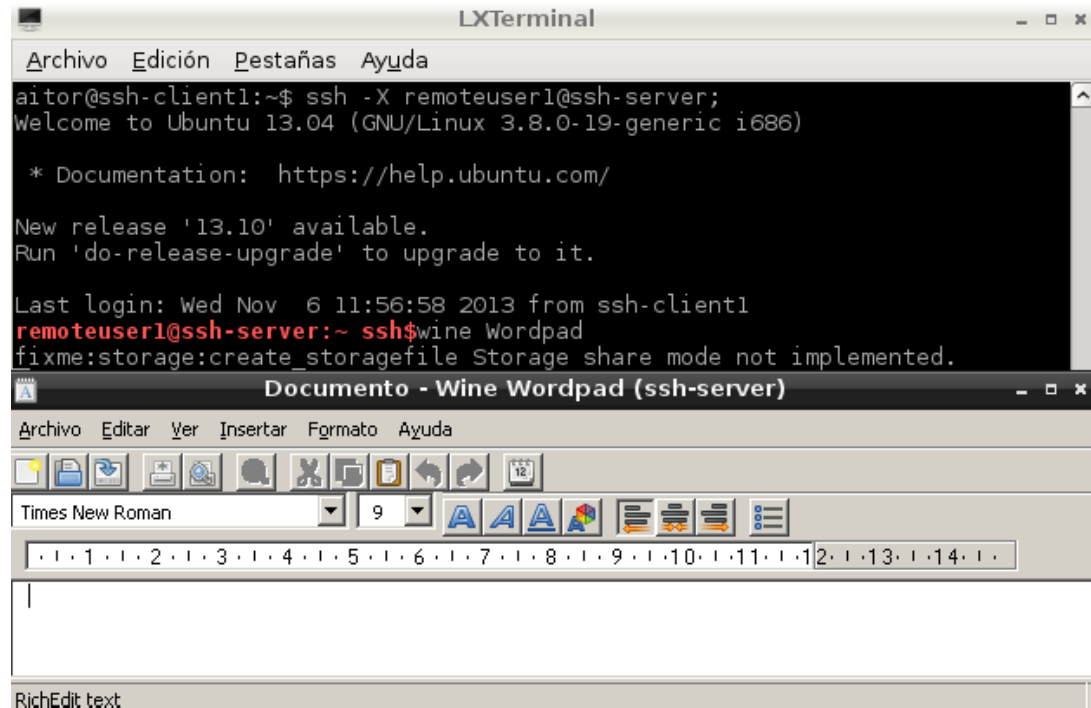
sin título - Geany (ssh-server)
Archivo  Editar  Buscar  Ver  Documento  Proyecto  Construir  Herramientas  Ayuda

No se han encontrado
1

11:19:46: Esto es Geany 1.22.
Estado 11:19:46: Se ha abierto un archivo nuevo: «sin título».
Esto es Geany 1.22.
```

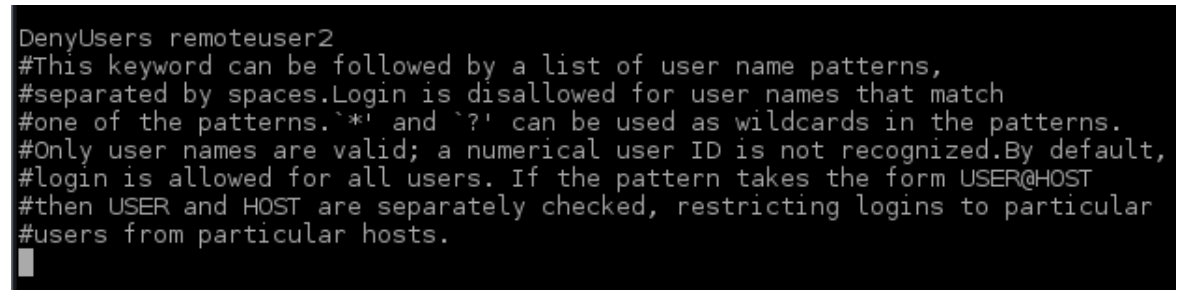
## Aplicaciones Windows Nativas

Podemos tener aplicaciones Windows nativas instaladas en ssh-server mediante el emulador WINE. Para instalarlo escribimos en la consola del servidor `apt-get install wine`. Tras instalarlo nos conectamos a el servidor mediante SSH por tunel X de la siguiente forma:



## Restricciones de uso

Restricción total: Para crear una restricción de uso del SSH para un usuario debemos de configurar el fichero `/etc/ssh/sshd_config` y poner la siguiente linea `DenyUsers remoteuser2`. Con esto conseguiremos que el usuario `remoteuser2` no pueda acceder a través de SSH a nuestro servidor



Restricción temporal: Para crear este tipo de restricción usaremos un script en código Ruby que nos ha facilitado el profesor. Consta de tres partes El fichero Ruby, donde se encuentra la tarea a programar, y los archivos `free` y `lock`. En el fichero `free` escribiremos la configuración para que todos los usuarios puedan acceder a través de SSH y en el fichero `lock`, escribiremos la configuración para restringir el uso a algun usuario.

```
Terminal - root@ssh-server: /home/aitor/Escritorio
Archivo Editar Ver Terminal Pestañas Ayuda
root@ssh-server:/home/aitor/Escritorio# ls
Guest limit-ssh-users.rb sshd_config.free sshd_config.lock
root@ssh-server:/home/aitor/Escritorio# ruby limit-ssh-users.rb
Ejecutando <limit-ssh-users.rb>...
Forma de uso:
* --lock , activar las restricciones SSH
* --free , desactivar las restricciones SSH
root@ssh-server:/home/aitor/Escritorio# ruby limit-ssh-users.rb --lock
Ejecutando <limit-ssh-users.rb>...
root@ssh-server:/home/aitor/Escritorio# ruby limit-ssh-users.rb --free
Ejecutando <limit-ssh-users.rb>...
root@ssh-server:/home/aitor/Escritorio#
```

Restricción sobre aplicaciones: Lo primero que debemos hacer es crear el grupo remoteapps e incluir al usuario remoteuser4 dentro de este.

```
root@ssh-server:/home/aitor# groupadd remoteapps
root@ssh-server:/home/aitor# adduser remoteuser4 remoteapps
Añadiendo al usuario 'remoteuser4' al grupo 'remoteapps' ..
Añadiendo al usuario remoteuser4 al grupo remoteapps
Hecho.
```

Tras haber hecho esto le damos los permisos de la siguiente forma a la aplicación Geany para que solo el usuario remoteuser4 pueda usarla .

```
root@ssh-server:/home/aitor# chgrp remoteapps /usr/bin/geany
root@ssh-server:/home/aitor# chmod 750 /usr/bin/geany
```