

MP0483.

Sistemas informáticos

UF6. Gestión de recursos en una red

6.3. Servidores

Índice

☰	Objetivos	3
☰	Servidores en la red	4
☰	Ejemplo: compartir carpetas en Windows	8
☰	Servidores de ficheros	13
☰	Ejemplo de servidor FTP en Linux y conexión desde Windows 10	18
☰	Ejemplo de conexión con servidor SAMBA	22
☰	Servidores de impresión	26
☰	Servidores de impresión en Windows y Linux	30
☰	Qué es un servidor de aplicaciones	35
☰	Estructura de un servidor de aplicaciones	39
☰	Instalación de varios servidores de aplicación	41
☰	¿En qué consiste la conexión remota?	42
☰	Protocolos y aplicaciones para conexión remota	45
☰	Resumen	56

Objetivos

Siempre que vayamos a trabajar con un sistema en red en el que usemos el modelo cliente-servidor, el diseño de los servidores resulta de especial importancia, pues sobre ellos recae la labor de dar el servicio a los demás equipos y usuarios.

Sabemos que podemos tener diferentes tipos de servidores: de ficheros, de impresión y de aplicación, fundamentalmente, y dentro de estos últimos incluiremos los “servidores web”, en general con los diferentes protocolos (HTTP, FTP, etc.). En este capítulo intentaremos ver los conceptos generales de cada uno de los tipos principales de servidores y qué aspectos debemos tener en cuenta a la hora de implantarlos en nuestro sistema y ocuparnos de su administración.

En esta lección perseguimos los siguientes objetivos:

1

Aprender los principales conceptos sobre el funcionamiento de un sistema cliente-servidor.

2

Conocer los principales tipos de servidores y su utilidad en nuestra red, así como las técnicas de conexión remota más usuales.

3

Comprender la diferencia entre un servidor de archivos y un servidor de aplicaciones, y qué parámetros son importantes para tenerlos en cuenta al dimensionarlos.

Servidores en la red

Entre las tareas del administrador de la red está la de diseñar su configuración y, por tanto, definir cuántos y qué ordenadores serán los que funcionen como “servidores” y cuál será el servicio que presten a los demás equipos de la red.

Las características hardware de los equipos dedicados a ser servidores normalmente son de mayores prestaciones que las de un equipo “normal” de usuario, y el sistema operativo que montan suele ser específico para la función de servidor.

También podemos tener equipos que nos ofrezcan el servicio (sea uno u otro) desde el exterior de nuestra red, es decir, equipos a los que nos conectamos normalmente a través de Internet y que se encuentran “en la nube”, en lo que modernamente conocemos como “servicios cloud”.

La tendencia general es que ambas configuraciones, los servidores “locales” y los “cloud”, estén presentes en los sistemas modernos y los usuarios no tengan por qué preocuparse de dónde está la máquina que les está proporcionando el servicio.

Entre otras cosas, el número de servidores determina en parte la configuración de la red, puesto que habrá que asegurar que todos los equipos pueden comunicarse con el servidor que necesitan, y a este dotarlo de una conexión con suficiente ancho de banda para atender las peticiones de los clientes. En función de su número y la cantidad de información a intercambiar la arquitectura de red podrá ser diferente (además de, por supuesto, tener en cuenta otros factores como la seguridad, etc.).



Evidentemente, los servidores de la red serán equipos de atención prioritaria para los administradores del sistema, donde además se extremarán las medidas de seguridad y se mirará mucho el rendimiento. Si el número de servidores es elevado, también lo será el esfuerzo necesario para su administración y mantenimiento.

Si el servidor se encuentra “en la nube” y lo que tenemos no es un servidor físico sino un contrato de nivel de servicio con un proveedor externo, entonces la atención a su administración y rendimiento cambia, pero no desaparece, pues será necesario asegurar que se están cumpliendo los parámetros del acuerdo de servicio contratado.



Tipos de servidores

Aunque nos centraremos en los tres tipos de servidores más utilizados (de ficheros, de impresión y de aplicaciones), conviene que hagamos un repaso de la **terminología** utilizada en este entorno y de los muchos tipos de servidores existentes, esto es:

- **Servidor de correo:** almacena, envía y recibe los “*e-mails*” del usuario. Nos conectamos a él con un cliente de correo (tipo Outlook u otro) o bien a través de una interfaz web (en ese caso se combina con un servidor web y hablamos de “*web-mail*”).
- **Servidor de archivos:** un servidor de ficheros/archivos se encarga de almacenar ordenadamente un gran número de archivos (que pueden ser de diferente tipo) y permitir el acceso a ellos a los usuarios que los solicitan, de forma que estos no tienen por qué conocer la ubicación física del archivo y trabajan con él como si estuviera en una unidad de almacenamiento más del sistema.
- **Servidor web:** principalmente almacena ficheros (documentos) HTML que nos muestra en lo que llamamos “páginas web” (nos los devuelve como respuesta a una petición nuestra vía HTTP).

- **Servidor proxy:** más que un servidor es un “intermediario” entre nuestro equipo y la red exterior, a la que se oculta la estructura interna y en el que se pueden establecer filtros, reglas, etc.
- **Servidor de base de datos:** proporciona servicio de gestión y almacenamiento utilizando un software de base de datos (p. ej. Oracle, MySQL, ...). Puede almacenar gran cantidad de información que está organizada en tablas y a las que se accede con un procedimiento propio de cada gestor de base de datos.
- **Servidores en clúster:** normalmente están formados por conjuntos de varios servidores trabajando de forma coordinada para almacenar una gran cantidad de información y al mismo tiempo para asegurar su disponibilidad y aumentar la seguridad.
- **Servidores dedicados:** simplemente es cualquier servidor que NO es compartido, es decir, es utilizado por una sola organización, para una aplicación concreta, etc.
- **Servidores de imágenes y multimedia:** se trata de servidores especializados en almacenar y suministrar archivos de imágenes o multimedia.

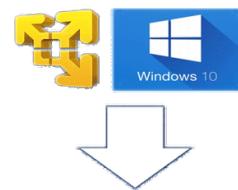
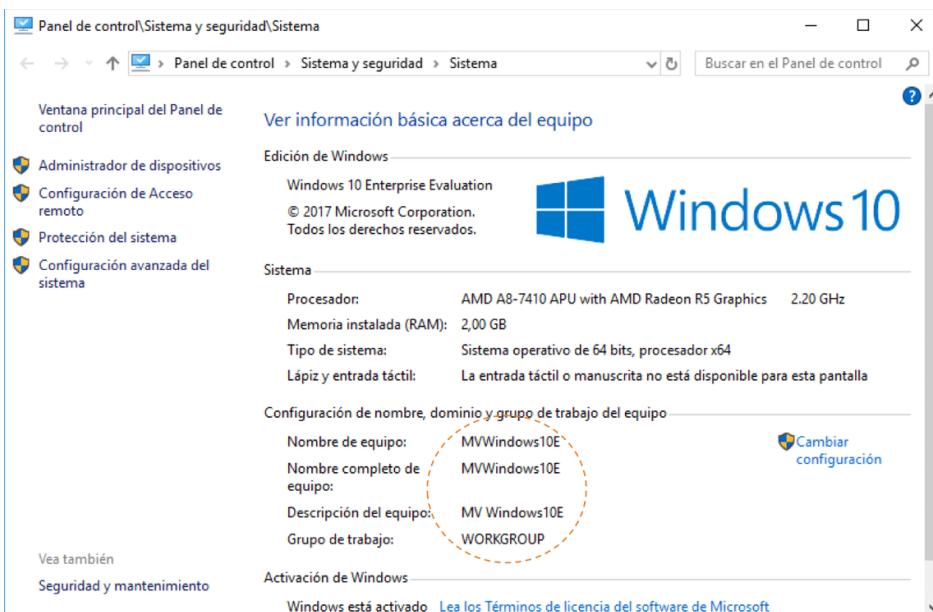
Ejemplo: compartir carpetas en Windows

Sin necesidad de instalar un servidor como tal, podemos hacer que nuestro sistema Windows permita el acceso a recursos locales (carpetas, directorios, impresoras), configurando las opciones de "compartición de recursos en red" que trae el propio Windows 10.

Esta opción te puede resultar muy útil, por ejemplo, para permitir que en la red local de tu domicilio (o un pequeño negocio) varios ordenadores puedan acceder a la información de uno de ellos con mayor capacidad de disco, etc.

Te mostramos la secuencia en imágenes y verás que es sencilla:

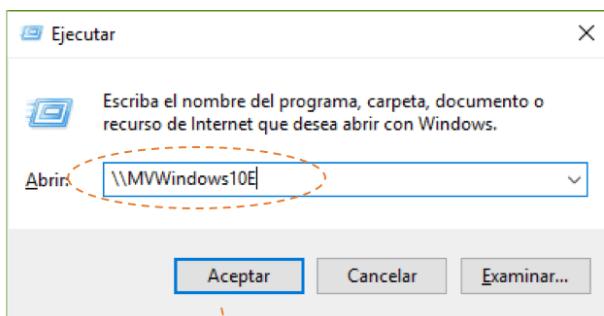
**Ejemplo: compartir archivos
en red local con Windows**



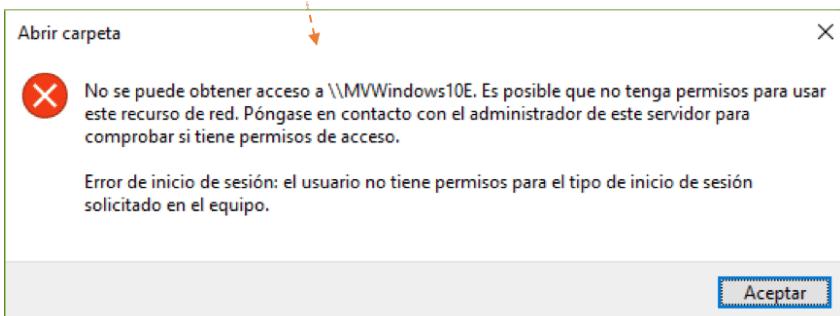
Si visualizamos la información del equipo en nuestra MV Windows 10, vemos que le hemos dado un nombre al "equipo" (MVWindows10E) y que el grupo de trabajo hemos dejado el que trae por defecto (WORKGROUP).

Para poder compartir carpetas necesitaremos básicamente:

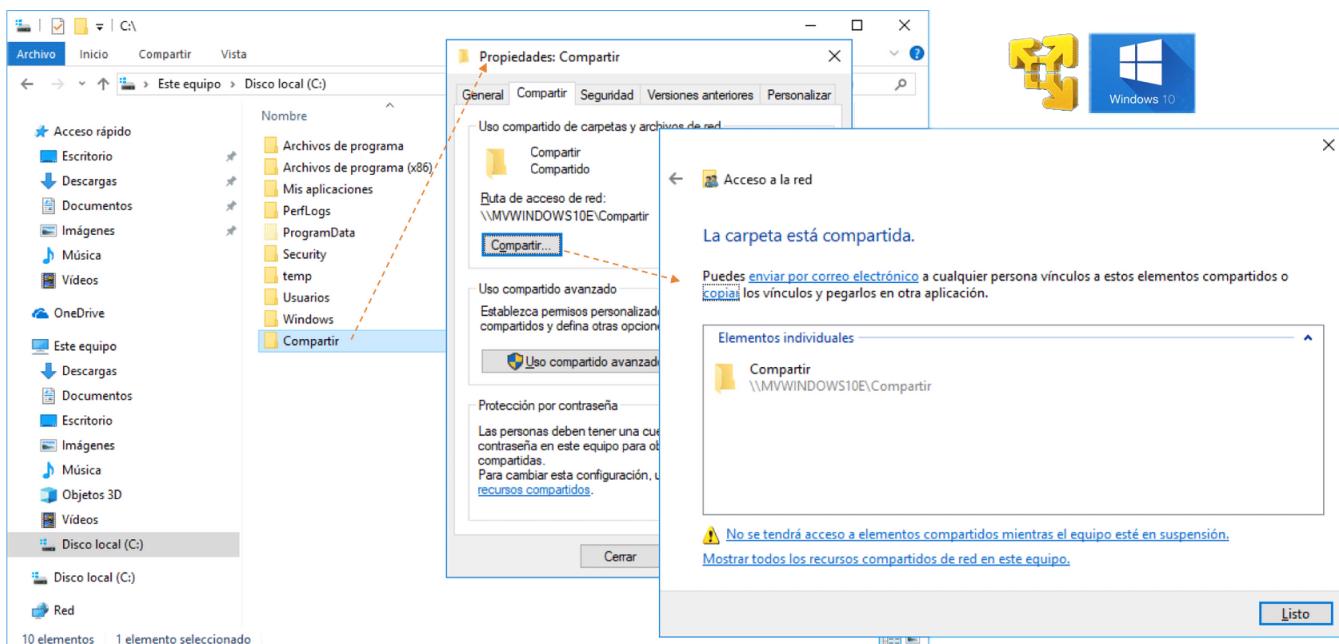
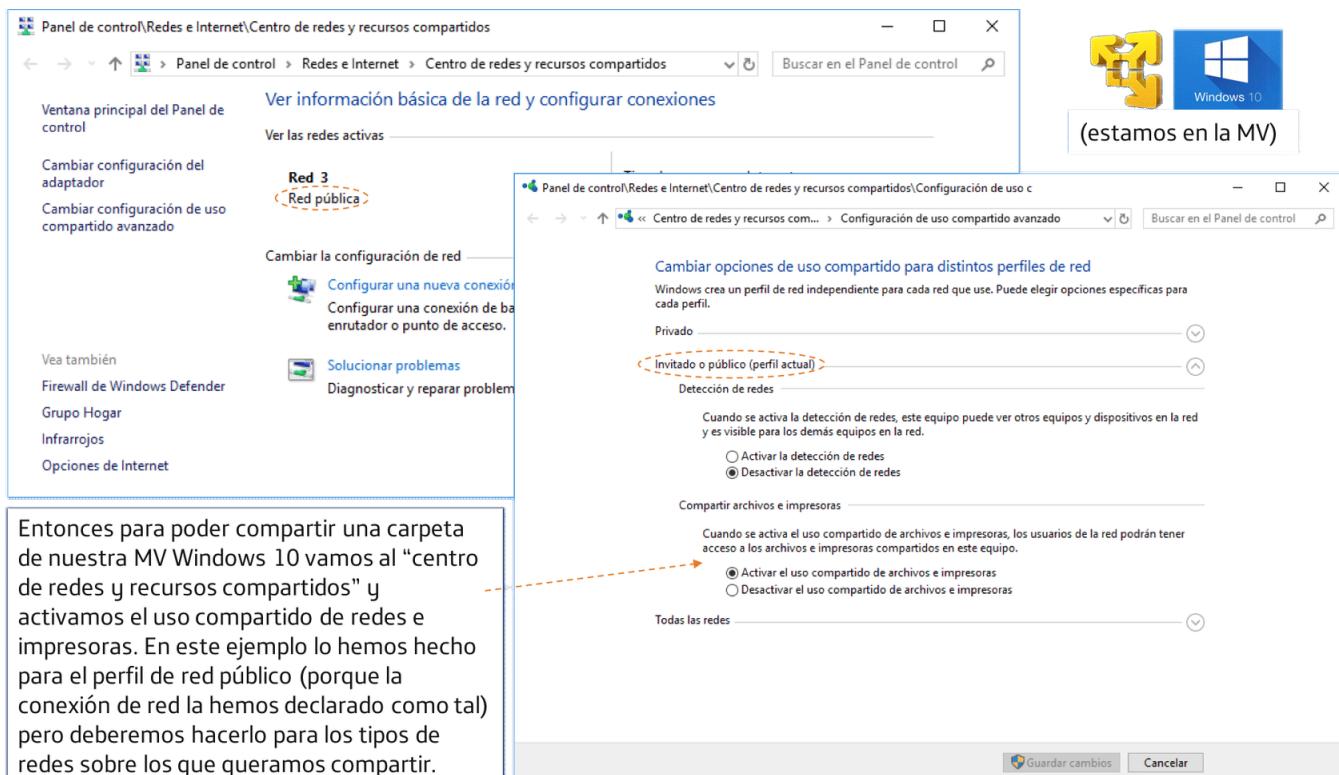
- Estar en el mismo grupo de trabajo.
- Habilitar la compartición de recursos a través de nuestra conexión de red.
- Señalar la carpeta que queremos compartir.



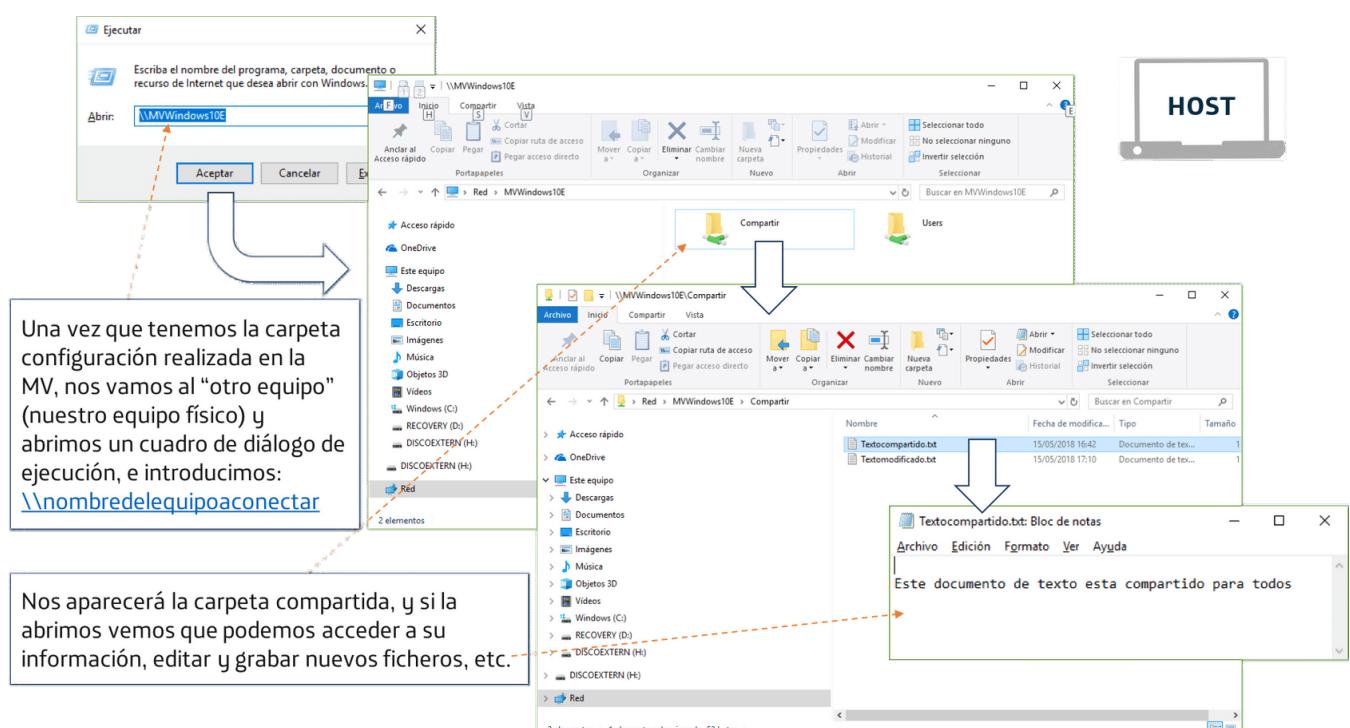
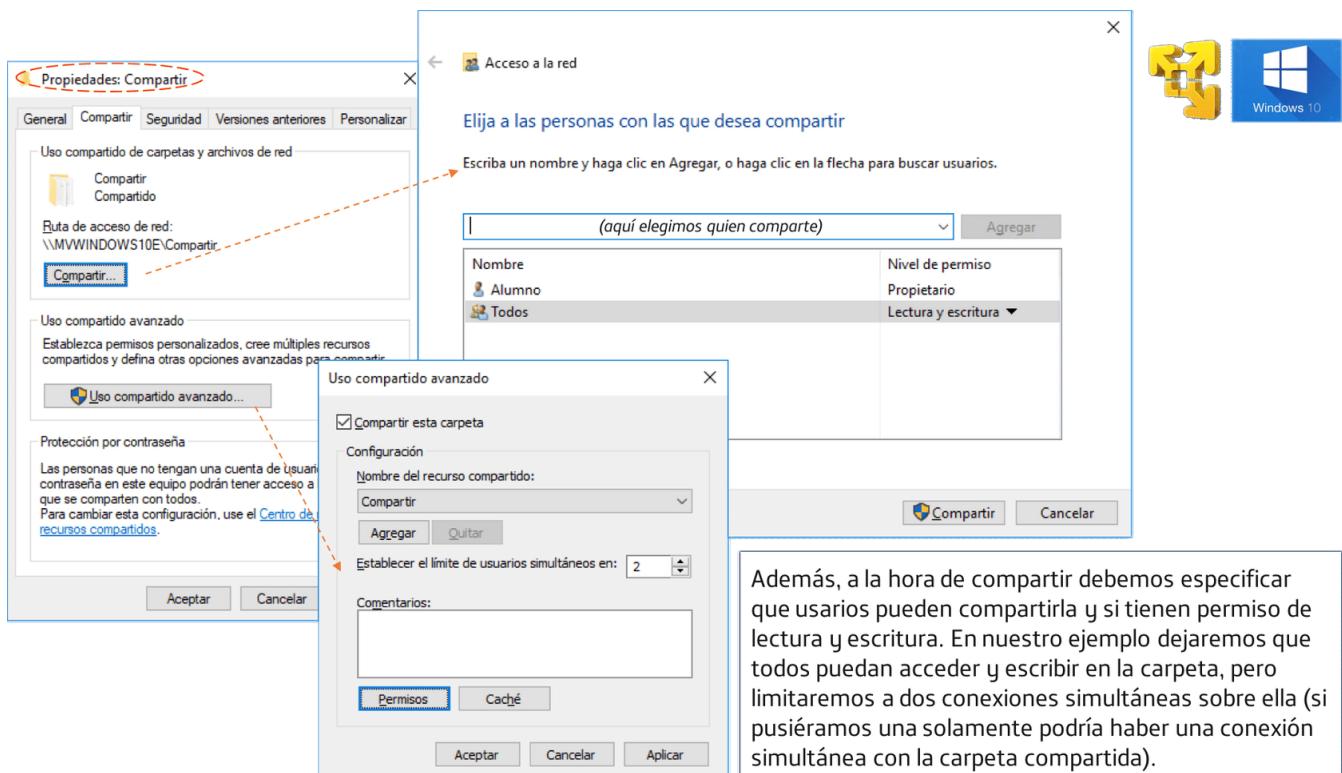
(estamos en el equipo físico)



Aunque conocemos el nombre del equipo, si desde nuestro host (equipo físico en el que tenemos corriendo la MV) intentamos conectarnos con él sin haberlo habilitado, nos saldrá un mensaje de que no podemos conectarnos.



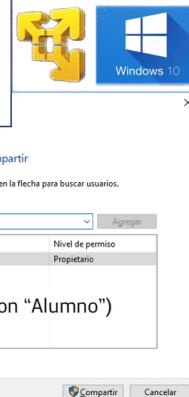
En nuestra MV creamos una carpeta que llamamos "Compartir" y en sus propiedades la configuramos como recurso compartido.



Una vez que tenemos la carpeta configuración realizada en la MV, nos vamos al "otro equipo" (nuestro equipo físico) y abrimos un cuadro de diálogo de ejecución, e introducimos: \\nombredelequipoaconectar

Nos aparecerá la carpeta compartida, y si la abrimos vemos que podemos acceder a su información, editar y grabar nuevos ficheros, etc.

En algunas ocasiones puede ser interesante compartir archivos pero solamente con un usuario (un grupo, varios usuarios, etc. pero no con todos) y entonces podemos configurar la carpeta compartida para que solamente él tenga acceso y sea necesario identificarse para poder ver su contenido. Un ejemplo con la misma carpeta anterior sería...



En la configuración de compartir la carpeta vemos como solamente le damos permiso al usuario "Alumno" sobre ella. Nadie más podrá acceder a ella remotamente.

Ahora cuando solicitamos la conexión desde el equipo remoto nos pide primero el usuario y la clave de acceso y luego nos aparece la carpeta compartida con su contenido.



Nombre	Fecha de modificación	Tipo	Tamaño
Documentos	16/05/2018 0:07	Carpetas de archivos	
shfrase.jpg	16/05/2018 0:07	Documento de texto	1 KB
Textocompartido.txt	16/05/2018 0:06	Archivo	91 KB
Texto	16/05/2018 0:06	Documento de texto	1 KB
Textomodificado.txt	15/05/2018 16:42	Documento de texto	1 KB
Textomodificado333.txt	15/05/2018 18:28	Documento de texto	1 KB
Textomodificado333.txt	17/05/2018 0:55	Documento de texto	1 KB

Ten en cuenta que no estamos montando un "servidor de ficheros" propiamente dicho, sino utilizando las características de compartición que nos brinda Windows 10, pero funcionalmente estaríamos accediendo a información de un sistema desde otro equipo remoto.

Servidores de ficheros

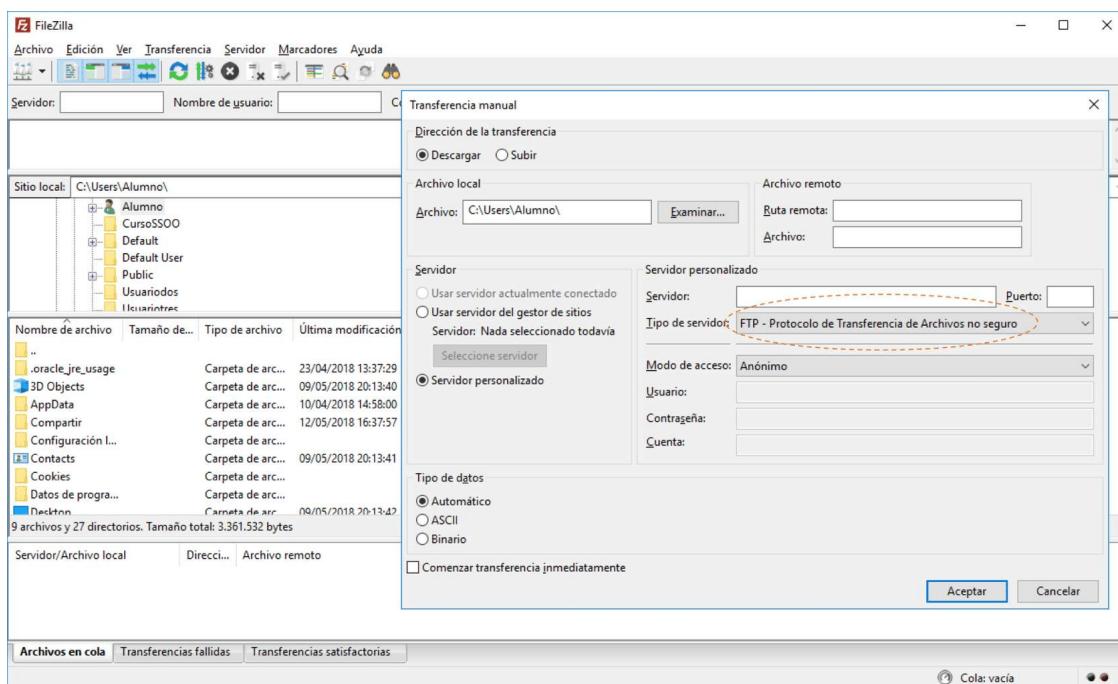
Como su nombre indica, un "servidor de archivos" es un equipo especialmente dedicado al almacenamiento de ficheros y a suministrar acceso a ellos (bajo petición) a aquellos "clientes" que lo soliciten.

Podemos montar un servidor de ficheros prácticamente sobre cualquier ordenador instalando el software de servidor, pero si el volumen de peticiones a soportar es muy alto será conveniente que sea un equipo con un hardware de altas prestaciones.

Evidentemente, entre los equipos clientes y los servidores debe existir una forma de comunicación (red) y un protocolo para intercambiar los archivos. Existen varios **protocolos** que pueden utilizarse para esto entre cliente y servidor, por ejemplo:

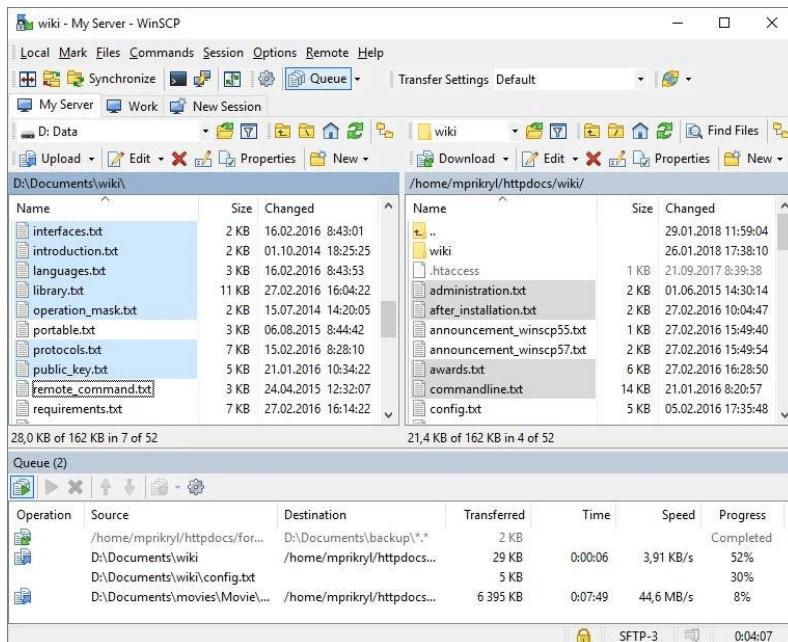
FTP

El protocolo de transferencia de archivos “**File Transfer Protocol**” (FTP) es bien conocido en el mundo de las redes TCP/IP para transferencia de ficheros, y puede utilizarse para implantar un servidor de ficheros en nuestra red. Existe software de servidor FTP para cualquier plataforma (sistema operativo) de las que nos encontramos hoy en día (Windows, Unix/Linux, Mac, ...) pues suele estar incluido en la capa de aplicación de la pila de protocolos IP. La transferencia de los ficheros entre el servidor y el cliente se realiza utilizando el **protocolo TCP** y suele utilizar los **puertos 20 y 21**. Se trata de un protocolo **eficiente y rápido, pero poco seguro**, ya que **toda la información (incluido el nombre de usuario y la clave)** se intercambia **sin cifrar**. Para mejorar la seguridad pueden usarse protocolos como SCP o SFTP, que utilizando SSH (para autenticación) permiten cifrar el tráfico.



SCP

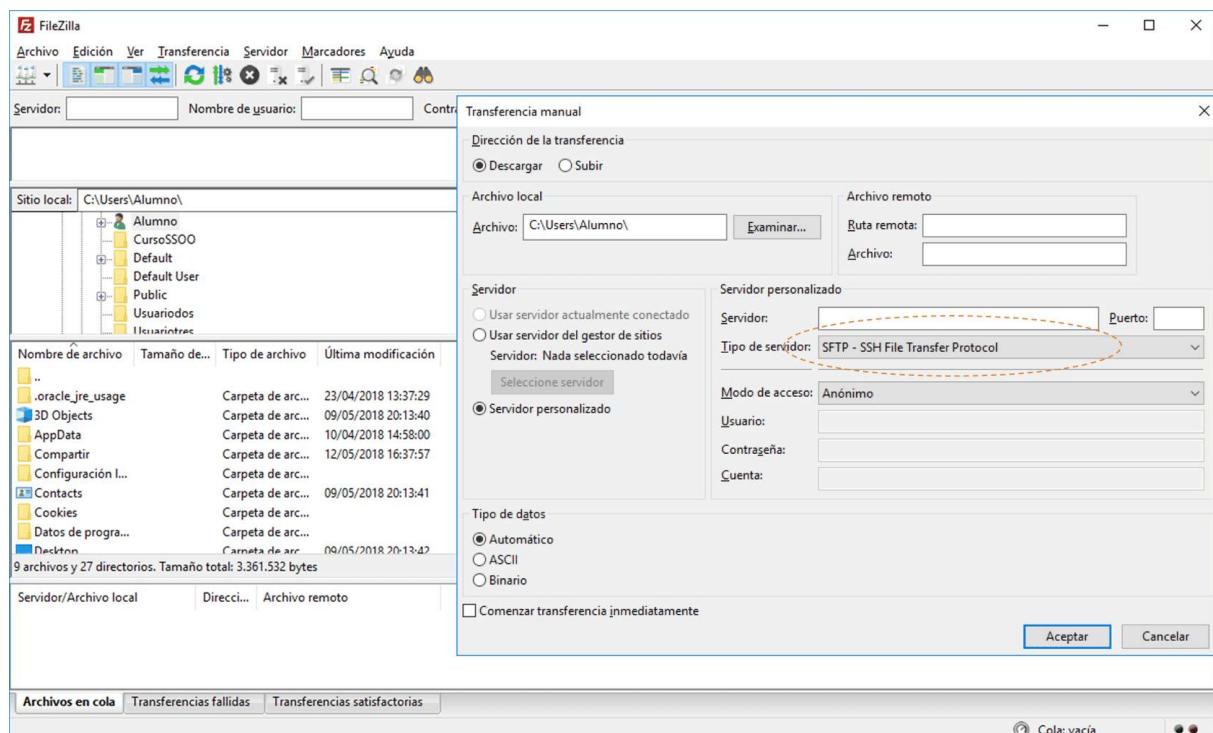
SCP (“Secure Copy”) permite cifrar los datos intercambiados mediante TCP entre el servidor y el cliente, y a su vez utiliza el protocolo SSH (“Secure Shell”) para la autenticación del usuario. La transferencia se realiza entre un programa SCP que corre en el equipo cliente y otro que corre en el equipo servidor. El manejo del programa SCP suele ser mediante comandos que pueden utilizarse en todos los clientes SCP, pero también algunos clientes SCP gráficos (GUI), como por ejemplo WinSCP, suelen incluir otras funcionalidades (que pueden ya no ser válidas en todos los entornos). Puedes ampliar la información en: <http://winscp.net/>.



SFTP

Las siglas SFTP significan “SSH File Transfer Protocol”, y se utiliza para la **transferencia segura de archivos sobre TCP** (usando el **puerto 22**), pero **no se trata de ejecutar FTP sobre SSH**, sino que es un protocolo independiente y, al igual que SCP, necesita de otro protocolo para realizar la autenticación del usuario y el servidor (típicamente usa SSH).

Nota: como ves en la figura, un mismo programa (como "Filezilla" por ejemplo) puede permitirnos usar diferentes protocolos para la transferencia de archivos.



SMB/CIFS

Originalmente llamado **“Server Message Block” (SMB)** y luego actualizado por Microsoft como **“Common Internet File System” (CIFS)**, es un protocolo que permite compartir recursos (ficheros, impresoras, puertos, ...) y de forma independiente de la plataforma utilizada, aunque es usado principalmente en entorno Windows. Existen extensiones de CIFS para Unix a partir de las versiones 2.6 de los servicios de ficheros del núcleo de Linux.

Los programas clientes se conectan a los servidores CIFS y, una vez establecida la conexión, envían comandos para ejecutar las operaciones sobre el servidor. Entre las posibilidades que ofrecen están:

- Establecimiento y control de las sesiones de trabajo.
- Negociación del dialecto de intercambio (SMB-2, SMB-2.1, ...).

- Exploración de la red para encontrar servidores SMB.
- Compartición de ficheros y directorios (lectura y escritura), con autentificación del usuario.
- Bloqueo de archivos y grabación.
- Notificación de cambios en archivos y directorios.
- Gestión de atributos extendidos de los archivos.
- Soporte Unicode.
- Independencia del protocolo de resolución de nombres.
- Impresión remota.
- Bloqueo de accesos oportunistas.

NFS

“Network File System” (NFS) o **Sistema de archivos en red** fue creado originalmente para los sistemas Unix de Sun Microsystems y posteriormente fue especificado en la “RFC 3530” de IETF (“Internet Engineering Task Force”). Es el sistema utilizado normalmente por Linux para compartir archivos y carpetas en red, aunque pueden usarse también los otros tipos de servidores.

NFS permite acceder a archivos en nodos remotos de la misma forma que si estuvieran localmente en nuestro equipo, de modo transparente al usuario e independiente de la arquitectura del servidor. Entre las funcionalidades que permite, además de compartir los archivos, están:

- Bloqueo del acceso a ficheros.
- Negociación de la seguridad.
- Listas de control de acceso (ACL).
- Interoperabilidad entre plataformas diferentes.

Para disponer de la función de servidor NFS en Linux será necesario tener instalado el paquete correspondiente en el servidor y en el equipo cliente, y por supuesto arrancar el servicio en el “servidor”. Previamente deberemos configurar el servidor NFS indicando qué carpetas deseamos compartir y con qué permisos (lectura/escritura), o desde qué equipos se permite la conexión.

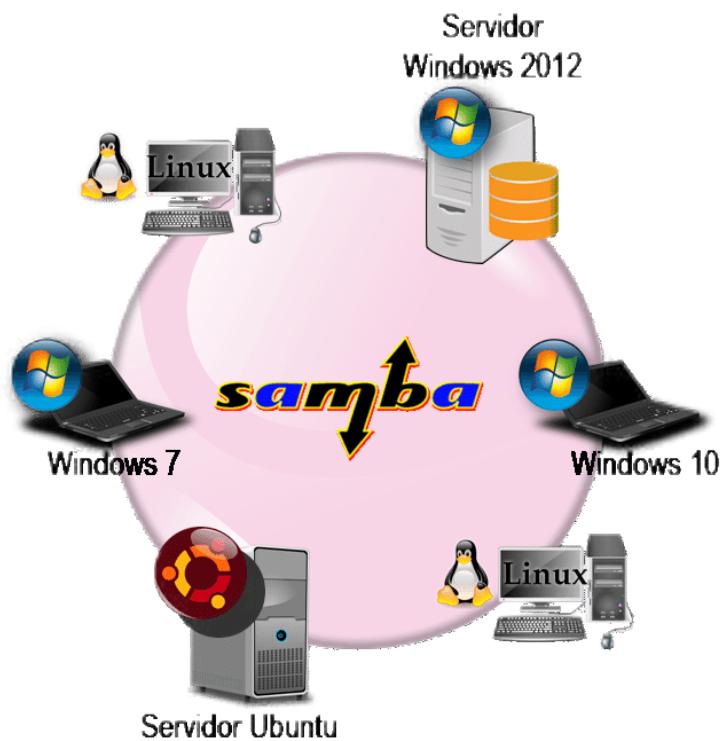
Un detalle importante: recuerda que en caso de existir conflicto entre los permisos establecidos en la configuración del servidor NFS y los del propio sistema Unix/Linux, siempre prevalecen los más restrictivos. Es decir, aunque por NFS demos permiso de escritura sobre una carpeta, si en Linux está como solo lectura no podrá escribirse sobre ella.

SAMBA

SAMBA está formado por un **conjunto de programas** que implementan sobre Linux/Unix el protocolo SMB utilizado por los sistemas Windows para compartir carpetas e impresoras, es decir, nos permite que desde equipos con sistema operativo Linux podamos conectarnos a carpetas en ordenadores con sistemas Windows.

Al utilizar SAMBA podremos tener en la red equipos con Windows y otros con Linux que intercambien información como si todos trabajasen en entorno Windows. Por supuesto, no es la única solución, pues por ejemplo también podríamos emplear protocolos del tipo FTP que ambos sistemas pueden utilizar.

En realidad SAMBA proporciona más funcionalidad que la compartición de archivos y carpetas, permitiendo que un equipo con Linux pueda actuar como “controlador de dominio” de una red Windows.



Ejemplo de servidor FTP en Linux y conexión desde Windows 10

Te mostraremos ahora un ejemplo de conexión con un servidor FTP desde un sistema Windows.

Para hacerlo usaremos nuestras dos máquinas virtuales del curso (Windows 10 y Ubuntu), pero podrías hacerlo igualmente desde sistemas reales en equipos físicos.

Te lo mostraremos en imágenes de ambos sistemas. La secuencia seguida es la siguiente:

1. Instalar el servidor FTP sobre el S.O. Linux (usaremos el programa "vsftpd").
2. Instalar un programa cliente en el S.O. Windows 10 (usaremos "Filezilla client").
3. Comprobaremos la conexión entre ambas máquinas con "ping".
4. Arrancaremos el servidor en Ubuntu (en este caso ya se arranca durante la instalación, si no habría que hacerlo).
5. Nos conectamos con el cliente desde Windows y realizamos acciones de intercambio de archivos.

Instalar servidor FTP en Ubuntu y conexión desde cliente FTP en sistema Windows 10

```
usuario@ubuntu:~$ sudo apt-get install vsftpd
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  liblvm3.8 libmircmons libqmi-glib1 linux-headers-4.4.0-31 lin
  linux-image-4.4.0-31-generic linux-image-extra-4.4.0-31-generic
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes NUEVOS:
  vsftpd
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 0 no ac
Se necesita descargar 115 kB de archivos.
Se utilizarán 336 kB de espacio de disco adicional después de est
Des:1 http://es.archive.ubuntu.com/ubuntu xenial/main amd64 vsft
Descargados 115 kB en 0s (129 kB/s)
Preconfigurando paquetes ...
Selecciónando el paquete vsftpd previamente no seleccionado.
(Leyendo la base de datos ... 245679 ficheros o directorios insta
Preparando para desempaquetar .../vsftpd_3.0.3-3ubuntu2_amd64.deb
Desempaquetando vsftpd (3.0.3-3ubuntu2) ...
Procesando disparadores para systemd (229-4ubuntu21.2) ...
Procesando disparadores para ureadahead (0.100.0-19) ...
Procesando disparadores para man-db (2.7.5-1) ...
Configurando vsftpd (3.0.3-3ubuntu2) ...
Procesando disparadores para systemd (229-4ubuntu21.2) ...
Procesando disparadores para ureadahead (0.100.0-19) ...
usuario@ubuntu:~$
```

Para disponer de un servidor FTP en Ubuntu/Linux instalaremos "vsftpd", que es un "demonio" que se ejecutará en segundo plano (una vez instalado) y estará a la escucha del puerto 20 de las solicitudes de conexión via FTP.

El programa vsftpd admite un buen número de personalizaciones a través de su fichero de configuración (/etc/vsftpd.conf) pero a nosotros de momento simplemente nos interesa ver su funcionamiento básico.

Una vez arrancado el servidor FTP nos conectaremos desde un sistema remoto (Windows 10).

```
usuario@ubuntu:~$ cat /etc/vsftpd.conf | grep -v "#"
listen=YES
listen_ipv6=YES
anonymous_enable=NO
local_enable=YES
dirmessage_enable=YES
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
secure_chroot_dir=/var/run/vsftpd/empty
pam_service_name=vsftpd
rsa_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
rsa_private_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
ssl_enable=NO
usuario@ubuntu:~$
```

Dentro del fichero de configuración repasamos las opciones activas (tiene muchas más pero filtramos con "grep" las que no empiezan con un "#" que serán comentarios).

Nos aseguramos de que en la instalación los parámetros tienen los valores que aparecen en la lista.

C:\ Administrador: Símbolo del sistema

```
C:\Windows\system32>ping 192.168.0.164

Haciendo ping a 192.168.0.164 con 32 bytes de datos:
Respuesta desde 192.168.0.164: bytes=32 tiempo=1ms TTL=64

Estadísticas de ping para 192.168.0.164:
  Paquetes: enviados = 4, recibidos = 4, perdidos = 0
            (% perdidos),
  Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 0ms, Máximo = 1ms, Media = 0ms

C:\Windows\system32>
```

ANTES DE EMPEZAR A HACER CONEXIONES POR PRECAUCIÓN COMPROBARES MOS que desde la MV de Windows10 con dirección IP 192.168.0.161, vemos la MV Ubuntu (192.168.0.164).

Evidentemente las IP son de nuestro ejemplo y cambiarán seguramente si lo pruebas en tu equipo.

usuario@ubuntu: ~

```
usuario@ubuntu:~$ ping 192.168.0.161
PING 192.168.0.161 (192.168.0.161) 56(84) bytes of data.
64 bytes from 192.168.0.161: icmp_seq=1 ttl=128 time=0.863 ms
64 bytes from 192.168.0.161: icmp_seq=2 ttl=128 time=0.760 ms
64 bytes from 192.168.0.161: icmp_seq=3 ttl=128 time=0.678 ms
64 bytes from 192.168.0.161: icmp_seq=4 ttl=128 time=0.847 ms
^C
--- 192.168.0.161 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 0.678/0.787/0.863/0.074 ms
usuario@ubuntu:~$
```

Comprobamos que desde la MV Ubuntu (192.168.0.164) vemos la MV de Windows10 (192.168.0.161)

Ubuntu 16.04 - VMware Workstation 14 Player (Non-commercial use only)

Player Archivo Editar Ver Ir Marcadores Ayuda

usuario@ubuntu: ~

```
usuario@ubuntu:~$ cat /etc/vsftpd.conf | grep -v "#"
listen=YES
listen_ipv6=YES
anonymous_enable=NO
local_enable=YES
dirmessage_enable=YES
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
secure_chroot_dir=/var/run/vsftpd/empty
pam_service_name=vsftpd
rsa_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
rsa_private_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
ssl_enable=NO
usuario@ubuntu:~$
```

Contenido del fichero de configuración del servidor FTP (programa vsftpd), **que es un demonio corriendo en el sistema**, a la espera de solicitud de conexiones.

Carpeta personal

Recientes Carpeta personal Escritorio Descargas Documentos Escritorio Imágenes Música Plantillas Público Videos VMware Ejemplos fichero.txt miclave.pub shfrase.jpg shfrase.jpg.gpg

Contenido del directorio personal del usuario, al que accederemos al realizar la conexión vía FTP desde el equipo remoto (se podría configurar para que accediésemos a otro directorio del sistema)

```
Administrator: Símbolo del sistema
C:\Windows\system32>
C:\Windows\system32>netstat -a

Conexiones activas

  Proto  Dirección local        Dirección remota      Estado
  TCP    0.0.0.0:135           MVWindows10E:0       LISTENING
  TCP    0.0.0.0:445           MVWindows10E:0       LISTENING
  TCP    0.0.0.0:49664          MVWindows10E:0       LISTENING
  TCP    0.0.0.0:49665          MVWindows10E:0       LISTENING
  TCP    0.0.0.0:49666          MVWindows10E:0       LISTENING
  TCP    0.0.0.0:49667          MVWindows10E:0       LISTENING
  TCP    0.0.0.0:49668          MVWindows10E:0       LISTENING
  TCP    0.0.0.0:49671          MVWindows10E:0       LISTENING
  TCP    192.168.0.161:139     MVWindows10E:0       LISTENING
  TCP    192.168.0.161:49878   db5sch101102019:https ESTABLISHED
  TCP    192.168.0.161:49886   a104-126-93-104:https CLOSE_WAIT
  TCP    192.168.0.161:49903   192.168.0.164:ftp    ESTABLECIDO
  TCP    [::]:135              MVWindows10E:0       LISTENING
  TCP    [::]:445              MVWindows10E:0       LISTENING
  TCP    [::]:49664             MVWindows10E:0       LISTENING
  TCP    [::]:49665             MVWindows10E:0       LISTENING
  TCP    [::]:49666             MVWindows10E:0       LISTENING
  TCP    [::]:49667             MVWindows10E:0       LISTENING
```

Podemos comprobar la conexión establecida con el comando "netstat", donde vemos que entre las demás presentes nos aparece la que tenemos con la dirección de nuestra MV Ubuntu, con protocolo ftp y sobre TCP.

```
usuario@ubuntu: ~
udp6   0      0 :::5353          ::::*                  -
usuario@ubuntu:~$ netstat -putna
(No todos los procesos pueden ser identificados, no hay información de propiedad del proceso
no se mostrarán, necesita ser superusuario para verlos todos.)
Conexiones activas de Internet (servidores y establecidos)
Proto Recib Enviad Dirección local        Dirección remota      Estado      PID/Program name
tcp    0      0 127.0.0.1:631         0.0.0.0:*          ESCUCHAR    -
tcp    0      0 127.0.1.1:53         0.0.0.0:*          ESCUCHAR    -
tcp    0      0 0.0.0.0:22          0.0.0.0:*          ESCUCHAR    -
tcp6   0      0 ::1:631            ::.*                  ESCUCHAR    -
tcp6   0      0 ::1:21             ::.*                  ESCUCHAR    -
tcp6   0      0 ::1:22             ::.*                  ESCUCHAR    -
tcp6   0      0 192.168.0.164:21    192.168.0.161:49907 ESTABLECIDO
udp    0      0 0.0.0.0:39767      0.0.0.0:*          -
udp    0      0 127.0.1.1:53         0.0.0.0:*          -
udp    0      0 0.0.0.0:68          0.0.0.0:*          -
udp    0      0 0.0.0.0:631         0.0.0.0:*          -
udp    0      0 0.0.0.0:5353        0.0.0.0:*          -
udp    0      0 0.0.0.0:41712       0.0.0.0:*          -
udp6   0      0 ::1:53662          ::.*                  -
udp6   0      0 ::1:5353           ::.*                  -
```

De igual manera, si nos vamos al sistema Ubuntu podemos comprobar que también podemos listar las conexiones establecidas y entre ellas la que tenemos con la IP del sistema Windows 10.

Para que veas lo sencillo que es trabajar con FTP a través de un cliente como Filezilla te lo mostramos en un [vídeo](#). Fíjate que podemos arrastrar archivos y directorios de un sitio a otro, como si estuviésemos trabajando con el administrador de archivos.

Ejemplo de conexión con servidor SAMBA

Como ya habrás leído, SAMBA es un conjunto de programas que permite compartir directorios y archivos entre sistemas Linux y Windows.

Bien, seguro que también has leído que es más que eso y tiene más funcionalidades, pero en este ejemplo nos limitaremos a mostrarte cómo se lleva a cabo esta conexión y lo cómoda que resulta, pues ambos sistemas acceden al espacio compartido como a cualquier otro directorio de cada uno de ellos.

Conexión Windows – Ubuntu a través de servidor SAMBA

```
C:\ Símbolo del sistema

Adaptador de LAN inalámbrica Wi-Fi:
Sufijo DNS específico para la conexión. . . :
Vínculo: dirección IPv6 local. . . : fe80::5dd0:3fc0:8893:ab55%10
Dirección IPv4. . . . . : 192.168.43.113
Máscara de subred . . . . . : 255.255.255.0
Puerta de enlace predeterminada . . . . : 192.168.43.1

Adaptador de túnel Conexión de área local* 12:
Sufijo DNS específico para la conexión. . . :
Dirección IPv6 . . . . . : 2001:0:9d38:953c:82c:1d18:3f57:d48e%
Vínculo: dirección IPv6 local. . . : fe80::82c:1d18:3f57:d48e%
Puerta de enlace predeterminada . . . . : 

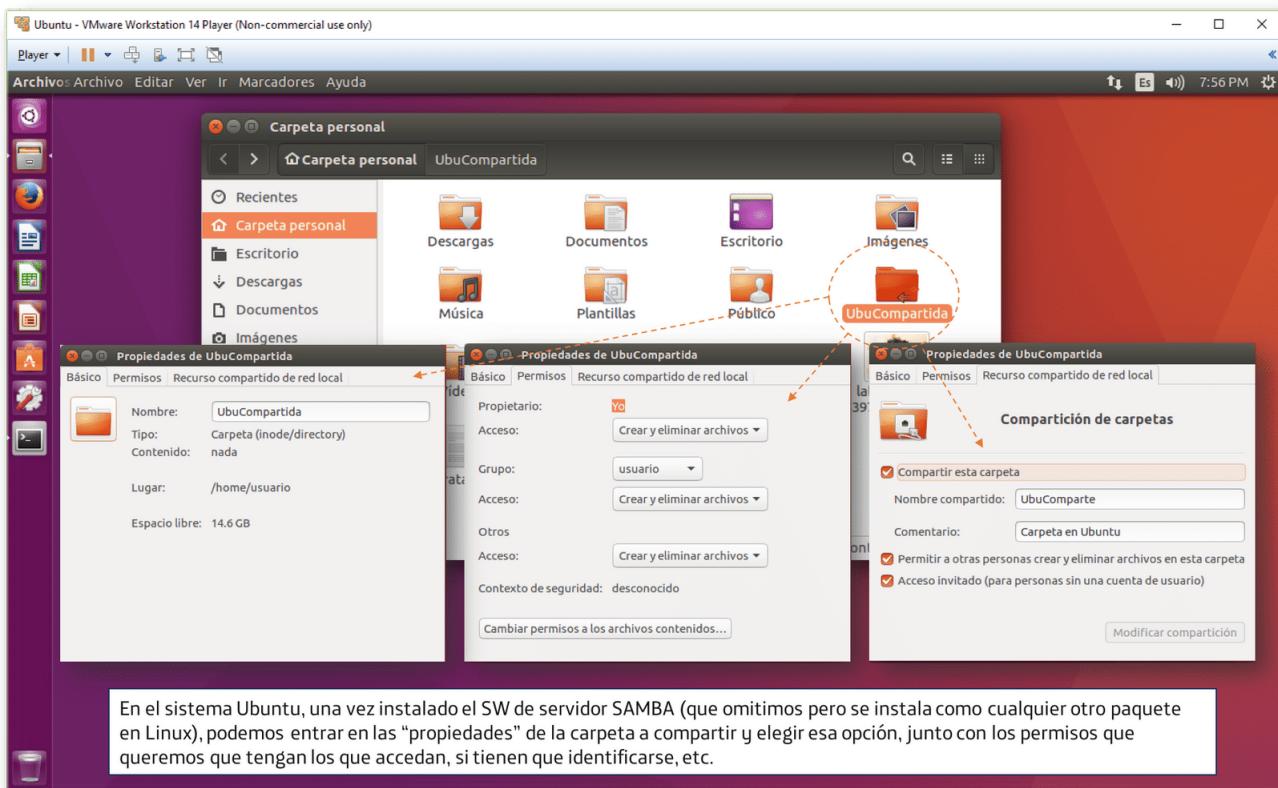
C:\Users\PC>ping 192.168.43.117

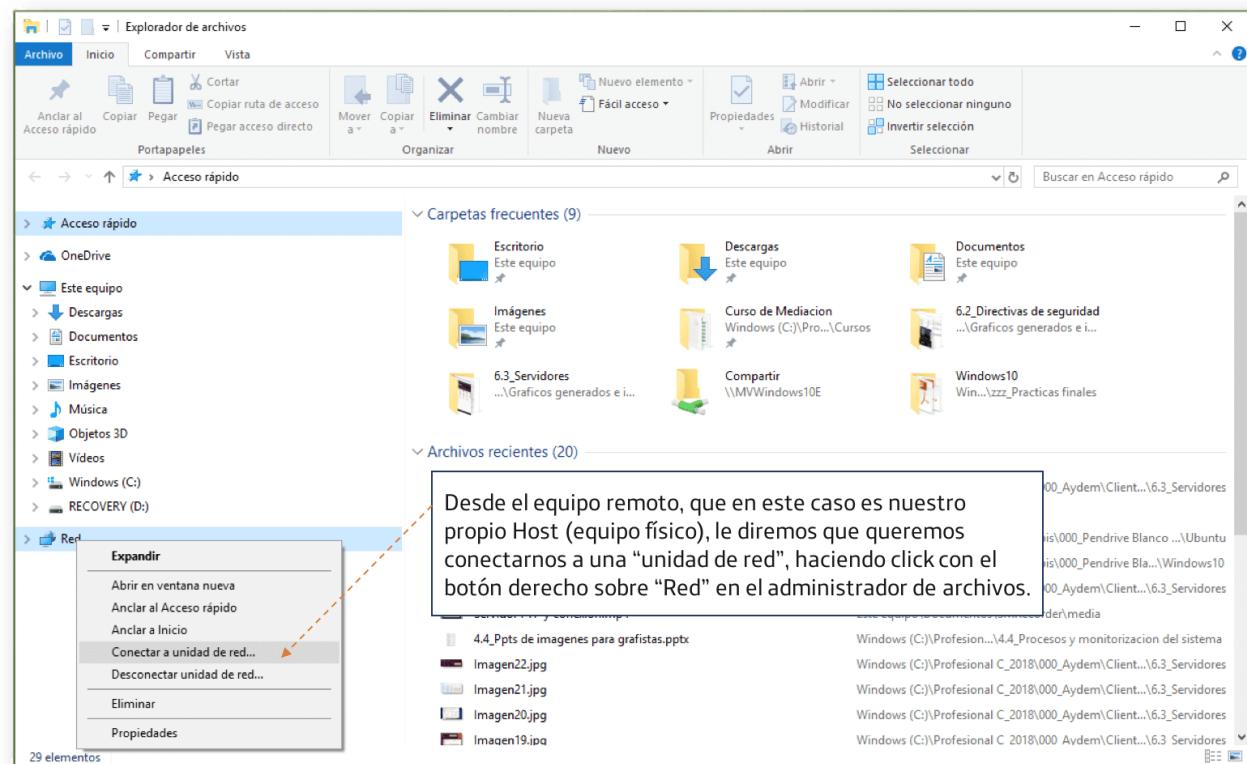
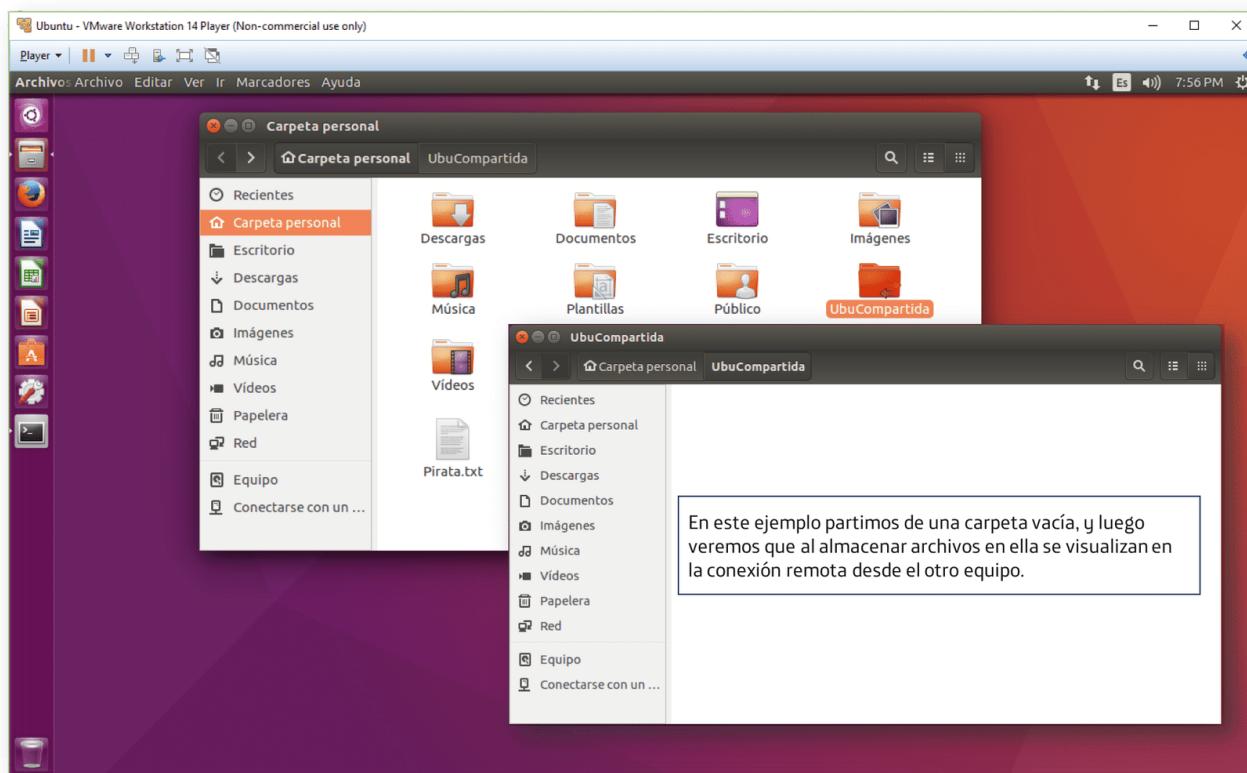
Haciendo ping a 192.168.43.117 con 32 bytes de datos:
Respueta desde 192.168.43.117: bytes=32 tiempo<1m TTL=64

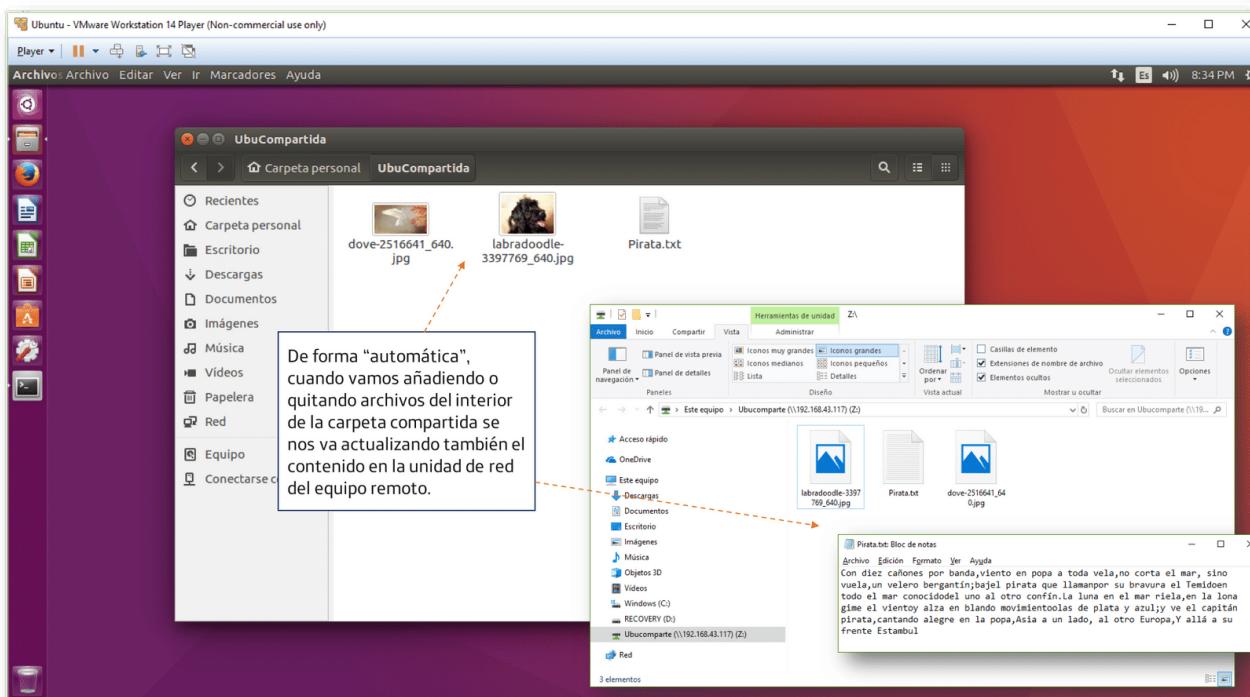
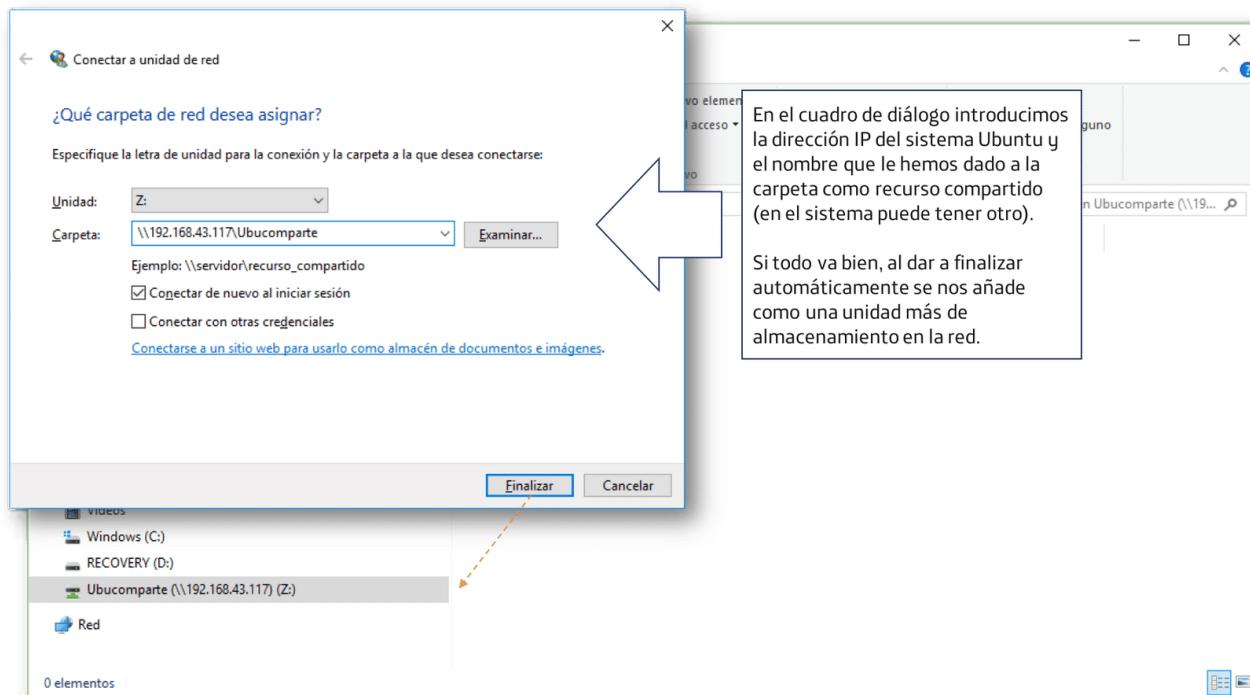
Estadísticas de ping para 192.168.43.117:
Paquetes: enviados = 4, recibidos = 4, perdidos = 0
(0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
```

Lo primero como siempre (en nuestras prácticas al menos) es ver que dirección IP tenemos y comprobar que "vemos" al otro sistema.

En este ejemplo vamos a conectarnos desde nuestro HOST con Windows 10 y con IP=192.168.43.113, y hacemos "ping" con una MV Ubuntu con dirección IP=192.168.43.117







Ten en cuenta que el proceso de instalación de SAMBA en Linux es como cualquier otro paquete y se puede hacer por comandos o a través del entorno gráfico, en función de la versión y el sistema que utilices.

Servidores de impresión

Aunque cada vez tendemos más al uso de documentación electrónica, la impresión de documentos sigue siendo una necesidad, tanto en un entorno doméstico como en cualquier puesto de trabajo.

Hasta hace unos años, cada equipo de usuario tenía conectada su propia impresora. Pero el trabajo en red hace posible compartir recursos, de modo que **el uso de impresoras compartidas en la red es la práctica más extendida**, incluso en pequeñas organizaciones.

El “servidor de impresión” es un **paquete SW (instalado en un equipo)** que permite que los ordenadores de la red puedan hacer uso compartido de las impresoras conectadas a él. El servidor centralizará las tareas de impresión, almacenará temporalmente los documentos y gestionará la utilización de los diferentes tipos de impresoras instaladas en la red.

Antes de continuar, vamos a repasar algunos **conceptos básicos**:

IMPRESORA

Si estamos trabajando “dentro” de un sistema operativo, una “impresora” definida en el sistema se refiere al **software de interfaz** que lo comunica con el dispositivo físico, al que también llamamos impresora. Lo habitual en entorno de trabajo de oficina, es referirse a la impresora para señalar al **equipo físico** que realiza la impresión de los documentos, y que puede ser “**local**” y estar conectada a un ordenador, o ser una “**impresora en red**”, las cuales disponen de interfaz de conexión a la red de la oficina, bien a través de cable Ethernet y un conector RJ45 como cualquier otro equipo, o también mediante interfaz inalámbrica wifi.



CONTROLADOR DE IMPRESORA

Es el **software** encargado de adaptar los datos del documento (proporcionado por la aplicación de usuario) al formato adecuado que entiende el dispositivo físico (impresora). La aplicación de usuario le enviará el documento en su formato (.doc, .pdf, etc.) y el controlador lo “traducirá” y adaptará al “lenguaje” de la impresora.

COLA DE IMPRESIÓN

Es el **software** que **se encarga de organizar la impresión** de los documentos, controlando los puertos de impresión, el estado y la configuración de la impresora, **gestionando las prioridades** de los distintos tipos de documentos a imprimir, preparar el documento que va a imprimirse y, cuando le toca el turno, enviarlo a la impresora. Cuando hay varios documentos solicitando ser imprimidos, este software se encarga de almacenarlos temporalmente en orden antes de enviarlos a la impresora. Durante el tiempo que el documento está esperando, en la **cola** tendremos la copia almacenada y otro documento de control generado por el propio software con los datos administrativos sobre el documento a imprimir. Al igual que gestiona las esperas de los documentos “en cola”, almacena información sobre los trabajos ya realizados y genera informes, estadísticas, etc.

PUERTO DE IMPRESORA

Es la **interfaz** a través de la cual el ordenador se comunica con la impresora física. Los sistemas soportan varios tipos de puertos, como por ejemplo:

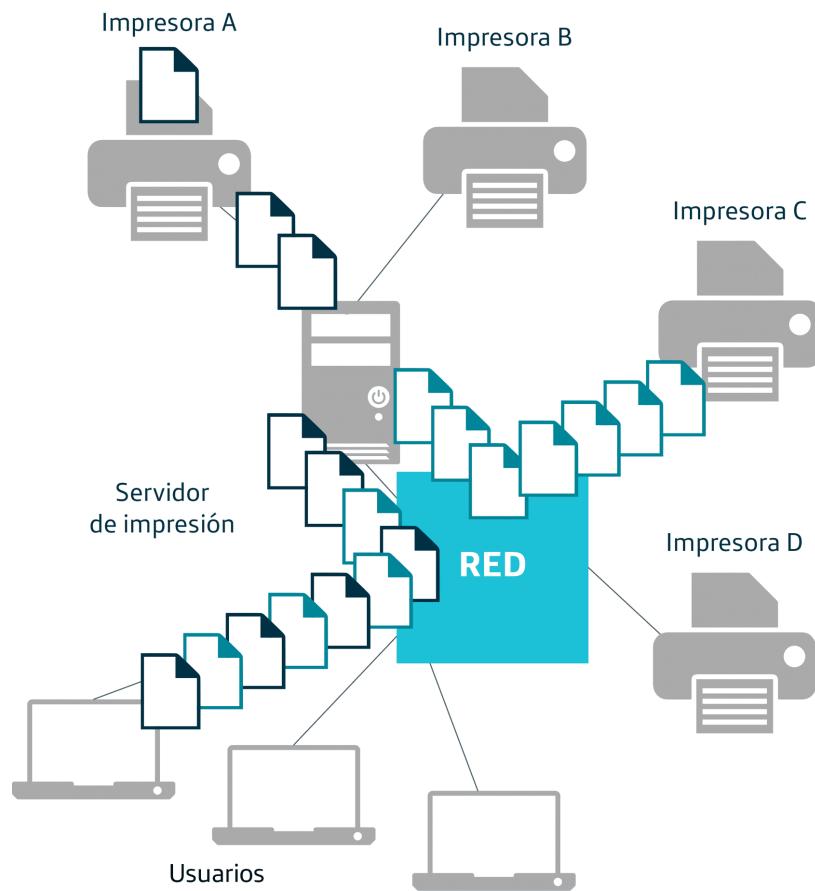
- "**Local Port**": puerto local en el equipo, por ejemplo un puerto paralelo. La impresora conectada puede ser compartida a través de la red. En aquellas impresoras que se conectan vía USB o IEEE 1394 y son dispositivos “*plug & play*” el puerto se crea de forma automática.
- "**LPR Port**": “Line Printer Remote” (LPR) permite imprimir a través de un servidor de impresión Unix/Linux instalando el servicio LPD (“Line Printer Daemon”).
- "**TCP/IP Port**": para la comunicación de impresoras a través de una red IP. Lo tienen la mayoría de las impresoras que disponen de conexión directa a la red.

¿Y qué parámetros podemos configurar en un servidor de impresión?

En general podremos configurar todo lo que tiene que ver con la ejecución de los trabajos de impresión. Según el sistema podremos hacerlo de una u otra forma, pero generalmente a través de una opción de la interfaz gráfica.

Por ejemplo, podremos:

- Administrar las distintas colas de impresión (local y de red).
- Verificar los puertos definidos, modificarlos o añadir otros nuevos.
- Visualizar los controladores instalados y gestionar su actualización.
- Definir la carpeta de almacenamiento temporal de los trabajos de impresión.
- Crear formularios de impresión definiendo tamaños de papel, formatos para sobres o etiquetas, plantillas de márgenes, etc.
- Configurar el registro de eventos del servidor y alertas ante ciertos sucesos.
- Programar el envío de notificaciones cuando se terminan de imprimir los documentos, o la información a enviar en los mensajes de error.



Servidores de impresión en Windows y Linux

Los servidores de impresión pueden ser equipos dedicados o bien ordenadores de usuario que comparten su impresora local.

Si queremos convertir un equipo en servidor de impresión, solo tendremos que compartir la impresora instalada localmente y configurar en los equipos clientes que se trata de una impresora en red a la que se accede a través de ese ordenador.

Servidores de impresión en Windows

Como de costumbre, las posibilidades varían de si estamos en un S.O. de red (servidor) o en una versión "desktop". Incluso dentro de la familia Windows, podemos tener varios tipos de herramientas, como son:

- **Administrador del servidor:** se usa para agregar en el equipo la función de "servidor de impresión", que lo configura como servidor e instala la consola de administración. Se puede usar además para administrar el servidor local. En el servidor tendremos que activar el uso compartido de impresoras y, por otra parte, arrancar el servicio LPD ("Line Printer Daemon"), que instala e inicia el servicio de servidor de impresión TCP/IP (LPDSVC) que permite que equipos Linux/Unix u otros que utilicen el servicio LPR impriman sobre las impresoras compartidas en el servidor.

- **Administrador de impresión:** proporciona información y detalles sobre las impresoras y los servidores de impresión que existen en la red. Se usa para supervisar las colas de impresión, configurar conexiones con equipos clientes en grupo, ejecutar *scripts*, conocer el estado de puertos, establecer filtros para la selección de impresoras, etc.

Por ejemplo, en entorno Windows Server las impresoras conectadas a un servidor se pueden administrar de forma remota y pueden guardarse como un elemento más del conjunto del sistema, de forma que los usuarios pueden localizarlas fácilmente utilizando los servicios de búsqueda de Windows.

Cuando las impresoras se conectan a través de TCP/IP la gestión se facilita mucho, pues se puede imprimir desde varios sistemas. En muchos casos el servidor proporciona una interfaz web a través de la cual los usuarios pueden actuar sobre sus trabajos de impresión pendientes en el servidor.

Servidores de impresión en entorno Linux

En los sistemas Unix/Linux existen varios sistemas (protocolos) de configuración de servidores de impresión. Por ejemplo, podremos definir como tipos de impresoras (hay más) los siguientes:

- 1 **Impresora local:** conectada directamente al equipo a través de un puerto paralelo o USB.
- 2 **Conexión por CUPS (IPP):** se puede acceder a ella a través de red TCP/IP y el protocolo de impresión en Internet (IPP).
- 3 **Conexión UNIX (LPD):** impresora conectada a un sistema Unix y accesible a través de la red vía LPD.
- 4 **Conexión Windows (SMB / SAMBA):** impresora compartida accesible por red conectada a un sistema Windows.
- 5 **Conexión Novell (NCP):** impresora compartida en un sistema empleando tecnología de red Novell Netware.



Linux

De todos ellos, quizás los principales y los que más nos interesan son, por un lado, el mecanismo de conexión de impresoras utilizando **CUPS** (“*Common Unix Printing System*”), y la posibilidad de compartir impresoras con **SAMBA**.

Servidores CUPS

“*Common Unix Printing System*” (**CUPS**) es un sistema de impresión modular, libre (licencia GPL) y portable, que permite configurar servidores de impresión y que prácticamente se ha convertido en un estándar “de facto” para las distribuciones Unix/Linux.

CUPS se encarga de **gestionar la planificación de la cola de trabajos** y tareas de impresión. Dispone de filtros de conversión de los datos a los formatos apropiados para imprimirlas, soportando una amplia gama de impresoras, desde matriciales hasta láser. CUPS también soporta *PostScript Printer Description* (PPD) y autodetección de impresoras de red. Para la gestión de las colas de impresión utiliza el protocolo IPP (*Internet Printing Protocol*), los comandos de impresión de Unix y algunas operaciones bajo protocolo *Server Message Block* (SMB).

En las versiones modernas de Ubuntu ya viene instalado el sistema CUPS y está disponible de forma automática (puedes abrir el navegador y acceder a "localhost:631"), pero en caso de que la distribución empleada no lo tenga, siempre puede instalarse como un paquete más.

The screenshot shows a web browser window titled "Inicio - CUPS 1.4.3" at "localhost:631". The menu bar includes "Inicio", "Administración", "Clases", "Ayuda en línea", "Trabajos", "Impresoras", and "Search Help". The main content area has three columns:

- CUPS para usuarios**: Includes links like "Descripción de CUPS", "Impresión desde la línea de comandos y opciones", "Qué hay de nuevo en CUPS 1.4", and "Foro de usuarios".
- CUPS para administradores**: Includes links like "Añadiendo impresoras y clases", "Gestionando políticas de funcionamiento", "Contabilidad básica de impresora", "Seguridad del servidor", "Usando autenticación Kerberos", "Usando impresoras de red", and "Referencia de cupsd.conf".
- CUPS para desarrolladores**: Includes links like "Introducción a la programación de CUPS", "La API de CUPS", "Programación de filtros y programas de conexión", "Las APIs HTTP e IPP", "La API PPD", "La API Raster", "Referencia del archivo de información del compilador de controladores PPD", and "Foro de desarrollo".

A large "UNIX PRINTING SYSTEM" logo is visible on the right side of the page.

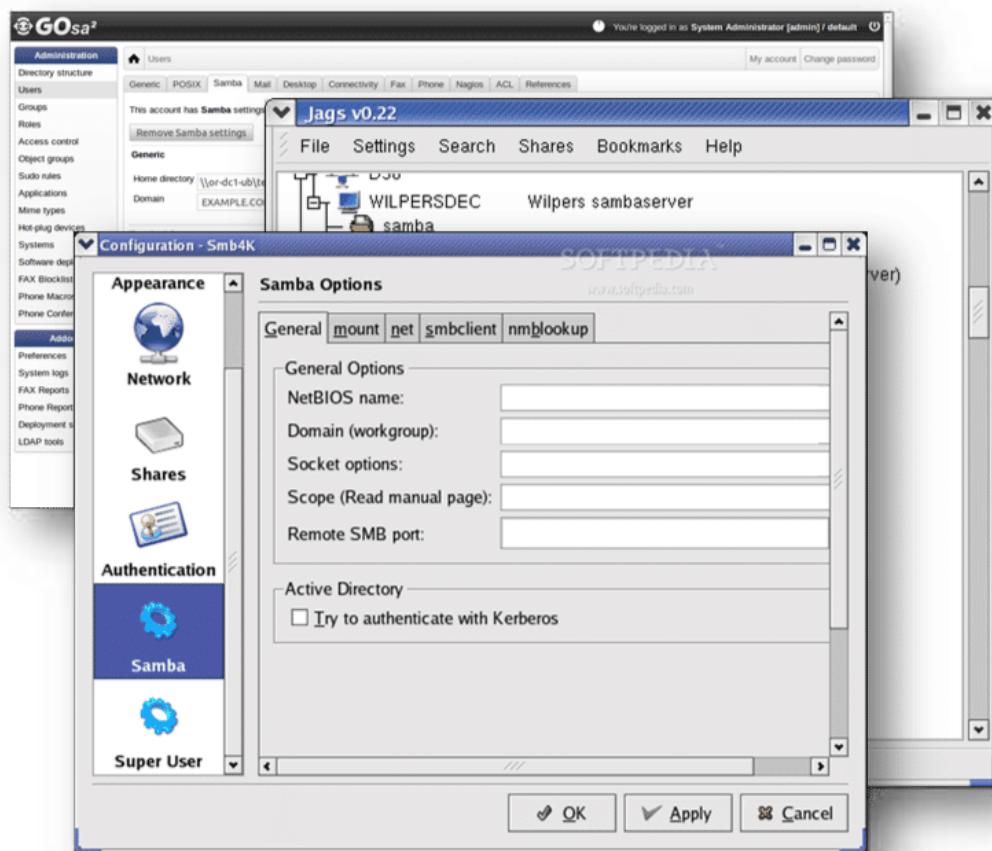
Servidores SAMBA

SAMBA es una reformulación moderna a partir del protocolo SMB/CIFS que nos **facilita la compartición de recursos entre sistemas Linux y Windows**.

Así pues, para poder instalar servidores de impresión gestionados con SAMBA, normalmente deberemos seguir los siguientes pasos:

1. Instalar SAMBA en nuestro sistema.
2. En el archivo de configuración de SAMBA declararemos las impresoras compartidas.
3. Reiniciar el servicio para aplicar la nueva configuración.
4. Dar de alta en el servidor Linux a los usuarios que queremos que tengan acceso a la impresora (por ejemplo creando un grupo e incluyéndolos).
5. Dar la orden de compartir las carpetas, directorios e impresoras a través de SAMBA.

Existen varias aplicaciones de administración de SAMBA a través de interfaz gráfica, por ejemplo: Gosa, Smb4K, Jags, Komba, etc.

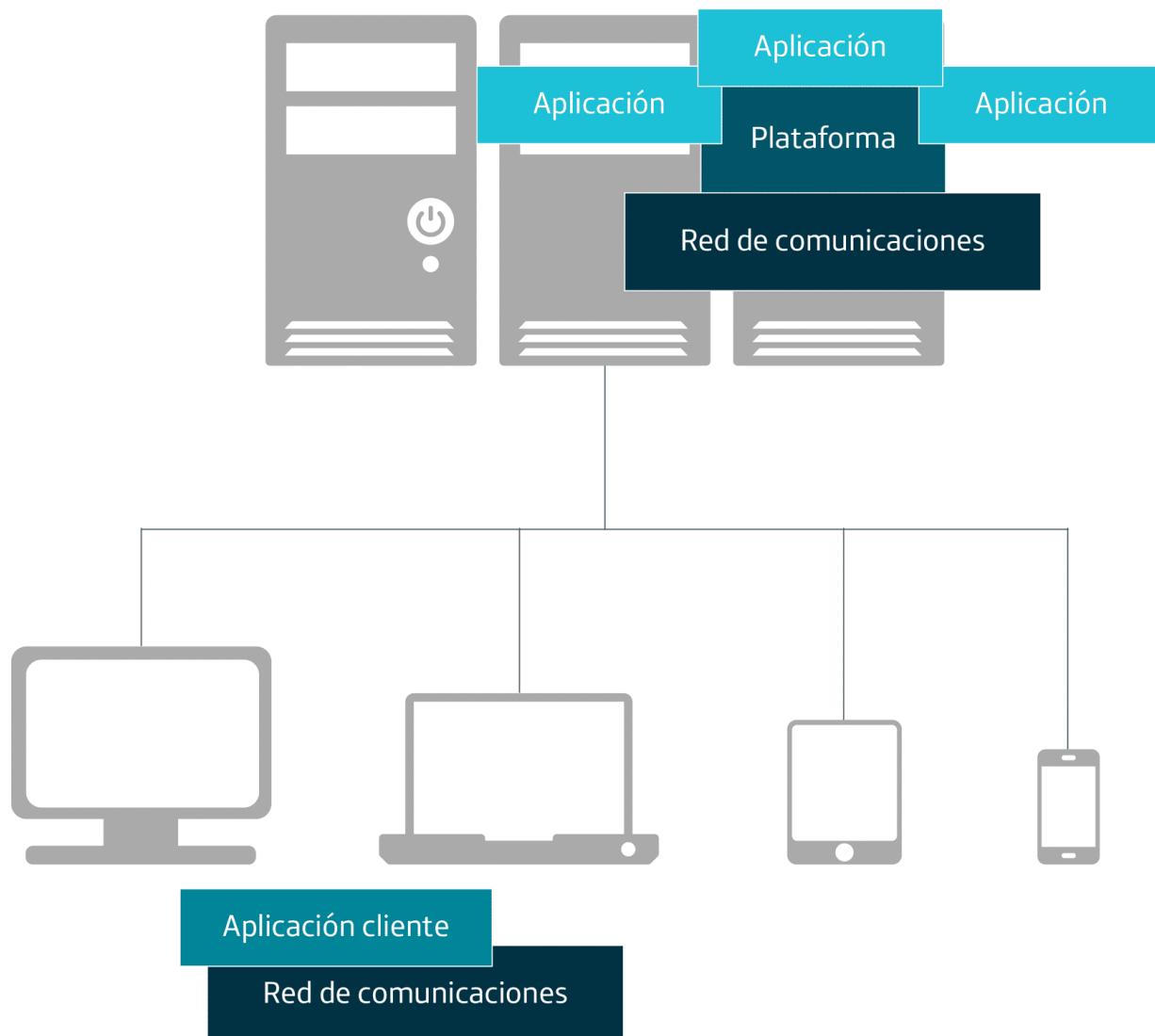


Qué es un servidor de aplicaciones

Un servidor de aplicaciones, como su nombre indica, es un tipo de equipo servidor que tiene la capacidad de ejecutar ciertas aplicaciones y al que nos conectaremos a través de una red y desde una "aplicación cliente".

A nivel de servidor, las diferentes plataformas existentes suelen ofrecer una **interfaz de programación de aplicaciones (API)**, que facilita a los desarrolladores la generación de aplicaciones a través de varias interfaces (modernamente las interfaces web principalmente) e independizando en gran parte el desarrollo del sistema operativo que hay debajo.

A nivel del equipo del usuario será necesario tener una **"aplicación cliente"** que conecte con el servidor por algún medio de comunicación, típicamente una red TCP/IP en la actualidad. Este cliente **solicitará la ejecución de la aplicación** (la que deseamos que ejecute la tarea para el usuario) en el servidor, **y recibirá la información resultado** de su ejecución. Evidentemente la comunicación puede ser interactiva, de forma que durante la ejecución de la aplicación el usuario intercambie datos desde el cliente (en el equipo de usuario).



Entre la aplicación cliente y el servidor se establecen además mecanismos de seguridad para garantizar la autenticación del usuario y sus derechos de ejecución sobre el servidor.

Existen múltiples plataformas tecnológicas en el mercado para soluciones de servidores de aplicaciones, y sobre cada una de ellas variadas soluciones de productos propietarios y libres.

Por ejemplo, a raíz del éxito del lenguaje de programación Java se desarrollan y extienden los servidores de aplicaciones Java EE, entre los cuales podemos encontrar como productos propietarios: Weblogic de Oracle (antes BEA Systems), WebSphere de IBM, EAServer de Sybase Inc., y entre los productos de SW libre; JOnAS, JBoss, Gerónimo y TomEE (Apache), etc.



SYBASE EA Server



Apache TomEE



Microsoft proporciona, a través de sus sistemas Windows Server, la posibilidad de configurar el rol de servidor de aplicaciones para ejecutar aplicaciones desarrolladas en su entorno .NET, y de forma integrada con su “*Internet Information Server*” (IIS, el servidor web integrado en Windows Server).

¿Por qué utilizar un servidor de aplicaciones?

Aunque la respuesta siempre estará en función de las necesidades de la organización y la funcionalidad esperada de los sistemas, podemos apuntar ciertas ventajas en el uso de un servidor de aplicaciones:

- **Alta disponibilidad:** los servidores de aplicaciones normalmente son equipos diseñados para trabajar en un régimen de 24 x 7 x 365 días, y con mecanismos de redundancia, distribución de carga, recuperación ante fallos y seguridad apropiados para ello.
- **Escalabilidad del sistema:** si las necesidades de la organización aumentan (número de usuarios, carga de procesos, etc.) podemos aumentar la potencia (equipamiento y configuración) de los servidores, o bien aumentar su número para atender las nuevas demandas.
- **Gestión y mantenimiento:** la actualización de las aplicaciones, su gestión y mantenimiento se realiza de forma más centralizada que cuando las tenemos “dispersas” instaladas en cada equipo de usuario. Cuando se hace un cambio es efectivo para todos los usuarios que se conectan al servidor.

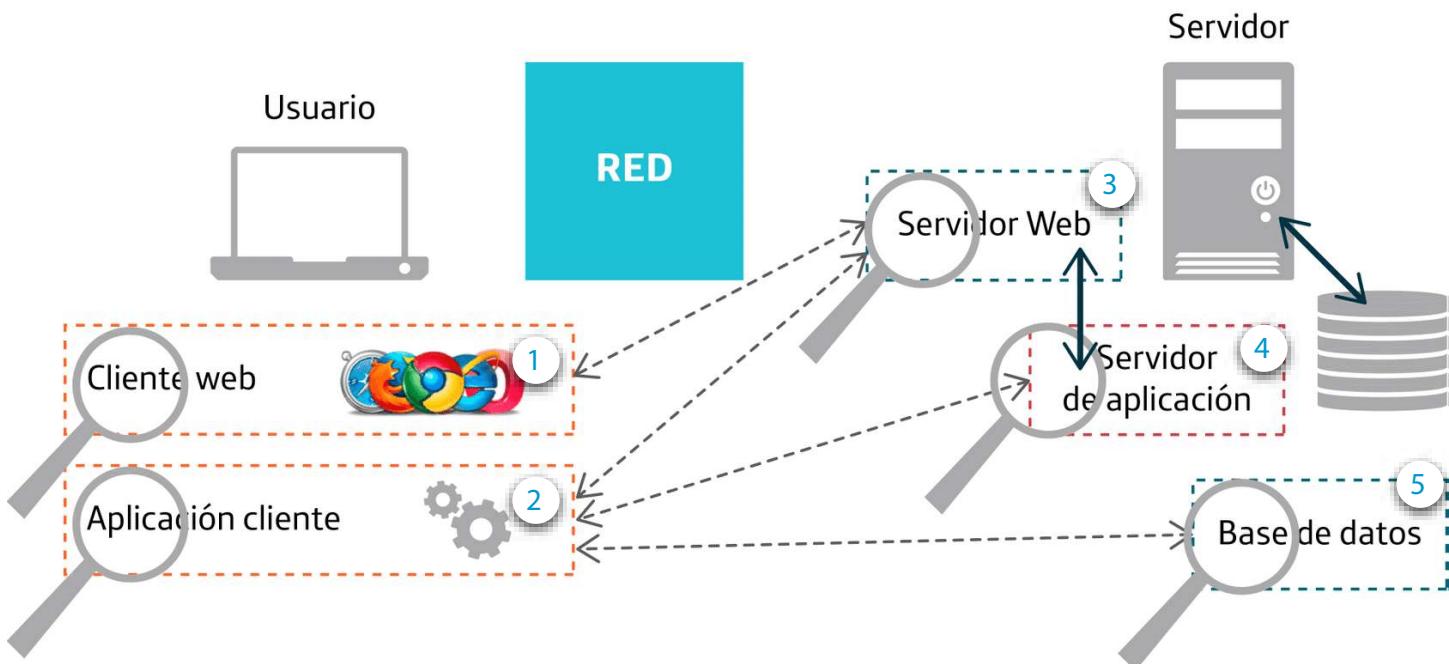
Evidentemente, es probable que no todas las aplicaciones de una empresa vayan a estar soportadas por un servidor, en muchos casos convendrá más instalarlas localmente en cada equipo que las vaya a utilizar.

Es el administrador del sistema quien debe definir la arquitectura de funcionamiento y, con ello, qué aplicaciones de utilización masiva conviene que funcionen bajo el modelo cliente-servidor.

Estructura de un servidor de aplicaciones

En la actualidad, la mayoría de los servidores de aplicaciones contemplan la interacción con los programas clientes a través de interfaces web, aunque esto no es algo obligatorio necesariamente.

En esta arquitectura de funcionamiento conviene tener claros algunos elementos.



1. Cliente web

Un navegador que interactúa con el servidor web vía HTTP normalmente. Recibe las páginas HTML/XML y puede ejecutar applets y código Javascript.

A menudo se le conoce como “cliente ligero”, básicamente un navegador web que envía una petición HTTP al servidor web. Si lo que pide son datos estáticos le responde directamente el servidor web, y si en cambio es necesario realizar alguna operación o ejecutar alguna aplicación, el servidor web pasa los datos al servidor de aplicación, que ejecuta la operación (p. ej., servlet, JSP) y devuelve la respuesta vía HTTP.

2. Aplicación cliente

Programa que corre en el equipo cliente, independiente del navegador, y que puede comunicarse a través de la red con el servidor web o directamente con el servidor de aplicación.

Se suele considerar un “cliente pesado”; puede ser, por ejemplo, una aplicación Java con interfaz gráfica o en modo consola que se comunica vía HTTP. Puede tener una interfaz gráfica y se encarga del registro de eventos, delegando la mayor parte de las funciones en componentes remotos (por ejemplo la resolución de nombres).

3. Servidor web

También llamado “contenedor web” o “motor web”, viene a ser la parte “visible” del servidor de aplicaciones, que se comunica vía HTTP o con protocolos seguros (SSL) con los clientes.

Esta arquitectura se ocupa de la parte “estática”, es decir, de disponer de una estructura de páginas web para mostrar al cliente, entorno gráfico de imágenes, etc., y de atender las peticiones que le llegan desde el cliente vía HTTP.

4. Servidor de aplicación

Conjunto de programas que realizan la “ejecución” de la aplicación propiamente dicha, proporcionando los diferentes servicios de negocio.

El servidor de aplicación se ocupa de la parte dinámica, es decir, de ejecutar las operaciones de la aplicación, acceder a los datos necesarios para realizarlas, elaborar un resultado y enviarlo al cliente.

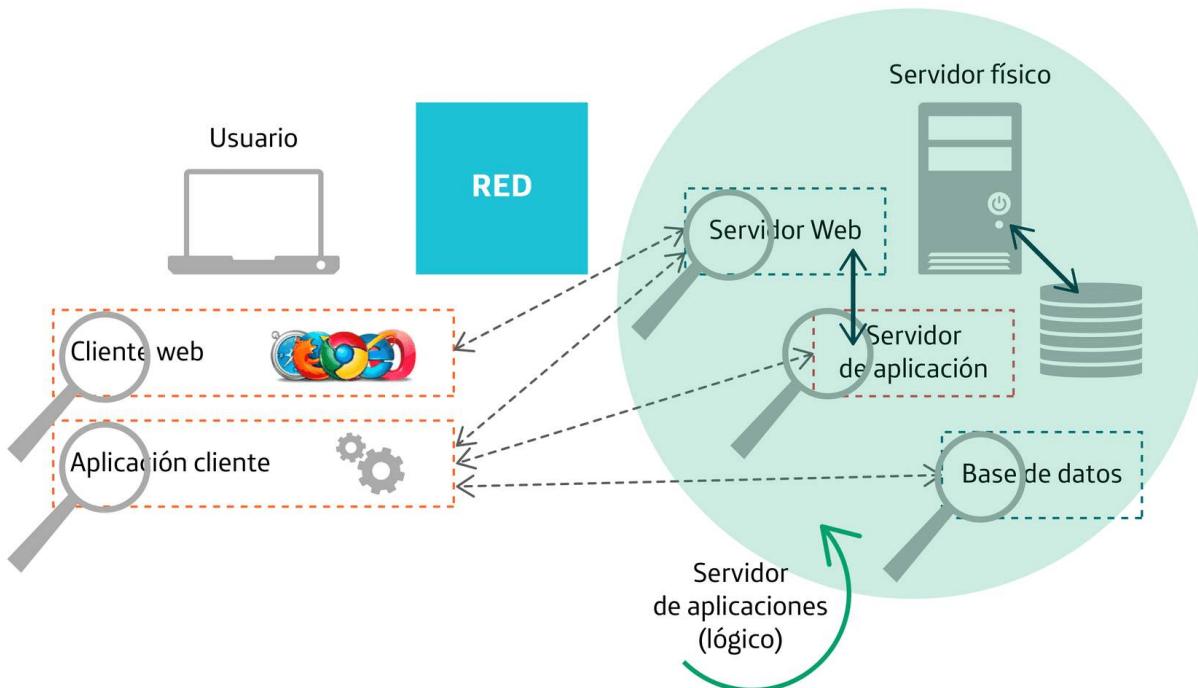
5. Base de datos

Sistema de almacenamiento de los datos empleados por la aplicación. En función de la implementación, puede dar servicio al servidor solamente o ser accedido también desde los programas clientes.

Instalación de varios servidores de aplicación

Debemos tener en cuenta que sobre un mismo servidor de aplicación físico (equipos físicos) pueden ser instalados varios servidores de aplicación funcionales (lógicos), es decir, lo que tenemos es un “servidor de aplicaciones” (en plural).

Cada servidor de aplicación instalado (lógico) puede a su vez contener los dos elementos, es decir; el servidor web y el servidor de aplicación propiamente dicho.



Además de esto, un “servidor de aplicación” constará de un **conjunto de procesos y servicios** necesarios para el funcionamiento global del sistema. Un ejemplo puede ser un servicio de nombres para conocer la ubicación física de los recursos, un gestor de transacciones, la gestión de disponibilidad de las aplicaciones, servicios de acceso a bases de datos, etc.

¿En qué consiste la conexión remota?

Cuando hablamos de “conexión remota” normalmente nos referimos a realizar acciones sobre un equipo informático sin trabajar directamente sobre él, sino estableciendo una conexión a través de una red de comunicaciones y enviando las órdenes desde otro equipo lejano.

Es decir, operamos desde un ordenador sobre los programas de otro, o accedemos a sus recursos, conectándonos a él a través de una red y un protocolo de comunicaciones.

Podemos establecer diferentes tipos de conexiones remotas y usar un variado número de protocolos, pero en general hablaremos de tres formas de conexión:

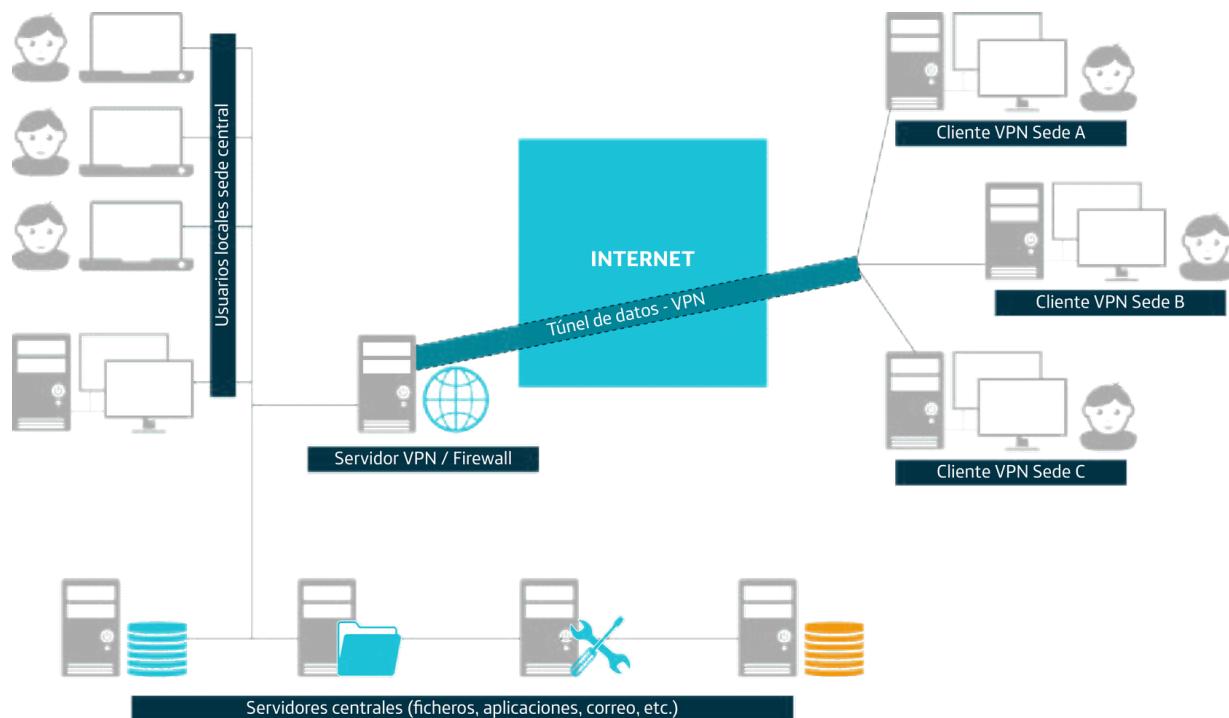
- **Conexión cliente-servidor:** un programa “cliente” accede a un programa “servidor” y a los servicios que le ofrece. Es quizás la forma más utilizada de conexión remota.
- **Conexión inversa o “back connect”:** el SW “cliente” del equipo de origen se conecta con un servidor. Este servidor se enlaza con un “administrador”, que conecta con el equipo que originó la “llamada.”
- **Conexión VPN:** se establece un túnel/canal de comunicación entre los equipos. A través de la conexión pueden incluso “hablar” varios equipos o aplicaciones simultáneamente.

¿Qué es una red privada virtual (VPN)?

Una **red privada virtual (VPN – “Virtual Private Network”)** consiste en la extensión de una red de datos privada (p. ej. una LAN) utilizando infraestructura de red pública (típicamente Internet, aunque existen otras), de forma que para los usuarios el efecto es como si estuvieran conectados “dentro” del entorno de la red privada, lo cual permite el **intercambio de información y la conexión de equipos de forma segura**.

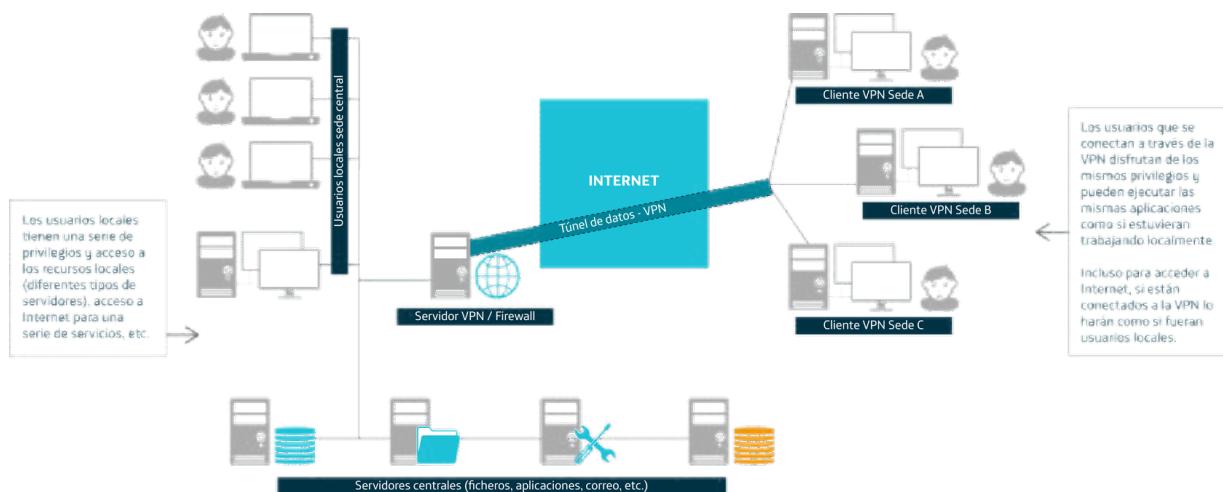
Entre sus características están:

- Identificación y autenticación de usuarios.
- Control de acceso: los usuarios pueden acceder a los datos a los que están autorizados.
- Integridad de los datos mediante funciones *hash* (algoritmos MD2, MD5 y SHA).
- Cifrado de los datos (algoritmos DES, 3DES y AES).
- Procedimientos para firmar los mensajes y asegurar el “no repudio”.
- Auditoría y registro de las actividades.
- Normalmente tienen un gran rendimiento y una alta calidad de servicio.



- ⓘ Existen muchos protocolos utilizados para construir VPN (IPSEC, PPTP, L2F, L2TP, SSH, SSL/TLS, ...) y productos específicos que incorporan hardware dedicado o implementaciones por software.

Ejemplo de VPN



¿Qué es el “tunelling”?

Cuando hablamos de hacer “tunelling” o de establecer una conexión a través de un túnel, lo que sucede es que se crea un túnel virtual que une el dispositivo de origen con el destino.

Este túnel virtual es similar a un túnel real, ya que se crea un túnel que une el dispositivo de origen con el destino. El tráfico se envía a través de este túnel y se desvanece en el destino. Al final del túnel, se vuelve a “desenvolver” para recuperar la PDU original.

La técnica de tunelling se puede emplear para redireccionar el tráfico a través de caminos fijados en la red, pero sobre todo para aportar seguridad a las comunicaciones empleando protocolos seguros que impidan el acceso a la información que viaja por el túnel a agentes externos.

Protocolos y aplicaciones para conexión remota

Dentro de cada uno de los tipos de conexión remota podemos encontrar diversos programas y aplicaciones que utilizan los protocolos de las redes que tienen por debajo.

Nos centraremos en las redes TCP/IP y veremos algunos ejemplos de los protocolos empleados y algunas aplicaciones que los utilizan.

Verás que te hablamos de bastantes protocolos y aplicaciones, pero no tenemos la intención de que te los aprendas ahora, simplemente que te suenen y que puedas fijarte en algunas diferencias, como por ejemplo sobre qué protocolos establecen las conexiones, si son cifradas o no, etc.

¡Es importante que te des cuenta de que no es lo mismo a nivel de seguridad utilizar un protocolo u otro!

Protocolos en modo cliente-servidor

Telnet

TElecommunication **N**ETwork = Protocolo de **emulación de un terminal remoto para ejecución de órdenes** por línea de comandos (como si estuviéramos en un terminal/consola local).

Para poder utilizarlo debemos tener un “**cliente Telnet**” en el equipo remoto que solicita el servicio a un programa “**servidor Telnet**” funcionando en el equipo al que nos conectamos. Una vez establecida la conexión el usuario puede iniciar una sesión con su usuario y clave de acceso. Incluso en algunos sistemas se permite la conexión de un usuario “invitado” o “*anonymous*” sin clave, pero no es recomendable desde el punto de vista de la seguridad.

La conexión a través de Telnet se suele llamar “**sesión en terminal virtual**” (VTY). Podemos usar Telnet sobre Unix/Linux y Windows (la mayoría de los S.O. incluyen por defecto un cliente Telnet, aunque no todos el programa servidor). También ha tenido un gran uso sobre routers y equipos de red.

Características:

- Funciona en modo **cliente-servidor**.
- Funciona sobre una conexión **TCP**.
- Existen clientes Telnet para cualquier sistema operativo. Pueden emular la mayoría de terminales (el más extendido es el VT100) y también el teclado.
- Utiliza el **puerto 23**.
- Intercambio bidireccional de información en formato texto (ASCII de 8 bits).

Para iniciar sesión simplemente tecleamos en un terminal: *telnet + (Dir. IP o nombre del servidor)*, y al conectar nos pedirá usuario + *password*. Los comandos introducidos son los propios del sistema operativo al que nos conectamos (como si estuviéramos en un terminal local), y siempre se ejecutan en el *host* remoto, que devuelve el resultado.

Su gran desventaja es la seguridad, pues todos los nombres de usuario y contraseñas se intercambian en texto plano sin cifrar.

RSH

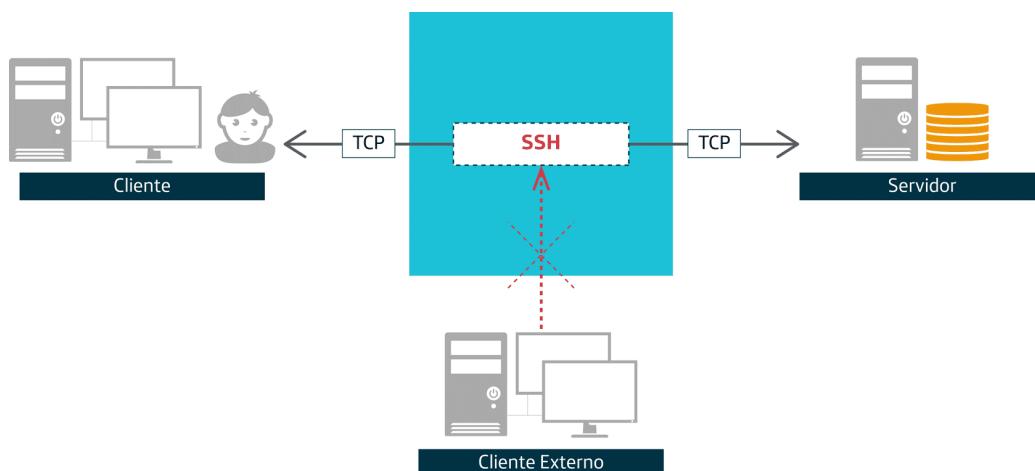
RSH (Remote SHell) = Protocolo de ejecución de comandos desde equipos remotos en entornos Unix, basado a su vez en el protocolo **rlogin** (aplicación que abre una sesión de terminal remoto sobre TCP/IP) y **rlogind** (proceso “demonio” servidor que recibe peticiones RSH por el **puerto TCP 513**). Con RSH podemos ejecutar comandos sobre un equipo distante con un nombre de usuario o sin hacer “*login*” (iniciar sesión) en el sistema. Para emplearlo teclearemos en sistemas Unix/Linux: *rsh + nombre_servidor + comando*. Normalmente los administradores no permitirán la ejecución de comandos remotos usando este comando sin hacer autenticación en el sistema. Además, al igual que Telnet, tiene el problema de que la **información se transmite sin cifrar**, por lo que tiende a utilizarse poco. En sistemas Windows el comando es “*rexec*” y en Windows Server es “*rshsvc*”.

SSH

SSH (Secure SHell). Común en sistemas Unix/Linux, aunque también existe para Windows. Proporciona la posibilidad de intercambio de **información cifrada** mediante mecanismos de clave pública entre el cliente y el servidor. Además realiza proceso de **autentificación mediante usuario y clave**, que también van cifrados. Al proceso de verificación entre cliente y servidor se le suele llamar “*handshake*” (apretón de manos) entre ambos, de forma que el cliente puede saber que se está conectando al servidor correcto y a lo largo de la comunicación comprobar que sigue conectado a él.

Permite copiar datos (archivos) **de forma segura** (utiliza algoritmos de encriptación de 128 bits), gestiona claves RSA e intercambia datos de otra aplicación estableciendo un canal seguro “tunelizado” entre ambos equipos. Tiene la posibilidad de intercambiar información de entorno gráfico X11, por lo que se puede usar para gestionar aplicaciones gráficas en una red.

Utiliza el **puerto 22 TCP**. A pesar de su seguridad, es posible que un tercero intente atacar estas comunicaciones, generalmente con la técnica “*man-in-the-middle*” (poniéndose en medio entre origen y destino).



FTP

FTP (File Transfer Protocol) = Como hemos visto al hablar de servidores de ficheros, FTP es un protocolo de capa de aplicación **utilizado para la transferencia de archivos** en un modelo **cliente-servidor**. De esta forma, FTP también viene a constituir un protocolo de conexión remota para la transferencia de archivos. En el servidor FTP se encontrará corriendo un programa (FTPD = FTP daemon) que primero atenderá las solicitudes de conexión del cliente a través del **puerto 21 TCP** para realizar el tráfico de control (comandos pedidos por el cliente y respuestas del servidor), y luego a través del **puerto 20 TCP** se establecen conexiones para la transferencia de los ficheros, una para cada archivo enviado. Esta transferencia de archivos puede darse en las dos direcciones, es decir, el cliente puede descargarse ficheros desde el servidor o bien “subir” (enviar) archivos hacia él y almacenarlos allí. Existe también otro protocolo, que citamos solamente para que lo conozcas: el **TFTP** (Trivial File Transfer Protocol), que utilizando **datagramas UDP** permite el intercambio de archivos, pero sin servicios de directorio o autenticación de usuario, por lo que no es muy usado en la actualidad.

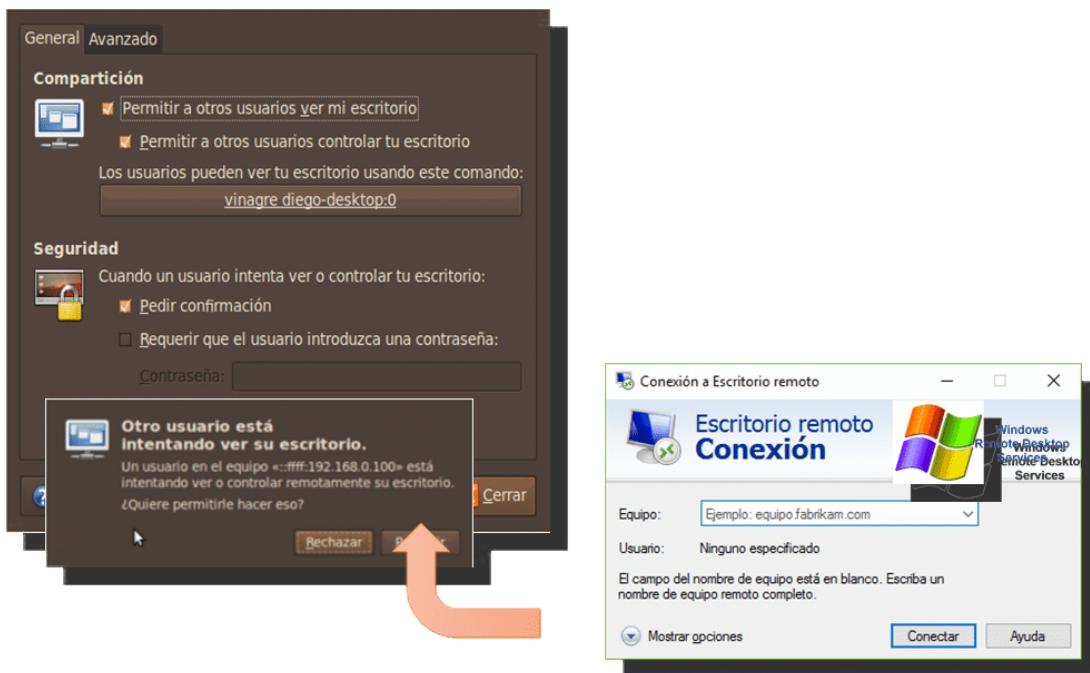
RDP

RDP (Remote Desktop Protocol). RDP es un protocolo desarrollado por Microsoft para la ejecución de aplicaciones en modo gráfico, estableciendo comunicación entre un cliente y un servidor Windows. La información gráfica se convierte a un formato RDP propio de Microsoft que se usa para regenerar el entorno gráfico en el equipo cliente. También permite gestionar los ficheros del ordenador remoto en una ventana local, compartir el portapapeles o ejecutar un programa con audio en el ordenador remoto y escuchar el audio en el equipo local. Cuando se inicia una sesión sobre un servidor RDP este mostrará la pantalla de bienvenida de Windows, sin aparecer lo que está ejecutando el usuario remoto.

Como características tiene:

- Utiliza el puerto TCP 3389 en el servidor.
- Cifrado de 128 bits.
- Seguridad a nivel de transporte.
- Posibilidad de compartir el portapapeles.

Por supuesto, otros sistemas operativos (p. ej. Linux) también ofrecen la opción de gestionar remotamente el escritorio con procedimientos similares; un equipo autoriza y el otro se conecta remotamente.



RCP

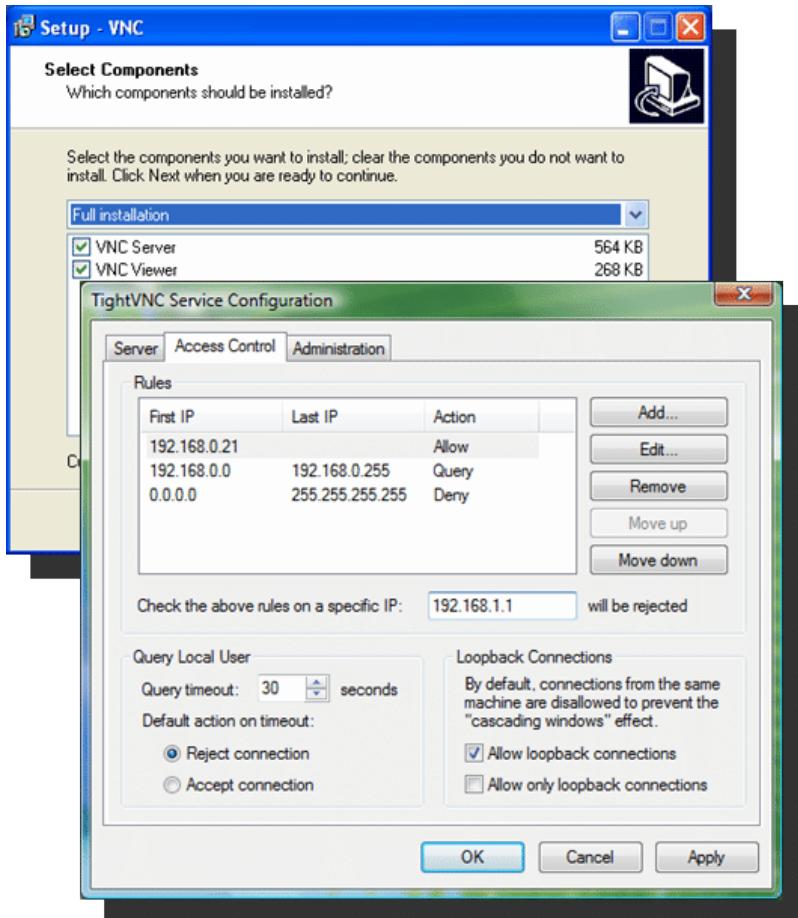
RCP (Remote CoPy) = En sistemas Unix RCP es la utilidad (y comando) para **realizar copias remotas de archivos** de un equipo a otro. Funciona utilizando **TCP** y envía la información **sin encriptar**, por lo que no resulta muy seguro y ha ido siendo sustituido por su equivalente, el SCP basado en SSH y por lo tanto más seguro.

Herramientas y aplicaciones en modo cliente-servidor

VNC

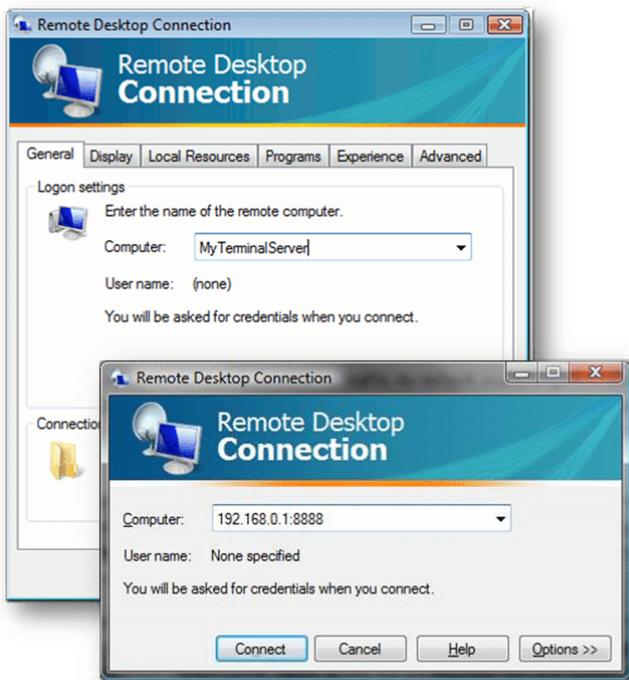
VNC (Virtual Network Computing) = Se trata de una herramienta para el control remoto de ordenadores, basada también en el modelo **cliente-servidor**, con interfaz gráfica y pudiendo manejar teclado + ratón sobre el escritorio remoto como si estuviésemos trabajando localmente. Utiliza el **protocolo RFB** (“Remote Frame Buffer”) y el cliente utiliza el puerto 5900 para conectarse. Varios clientes pueden conectarse a un mismo servidor VNC, por lo que un solo administrador puede gestionar varios equipos. Además, es independiente de la plataforma, lo que permite que se utilice con sistemas operativos diferentes. Su inconveniente radica en que no es un protocolo seguro. Aunque utiliza claves de seguridad (contraseñas), estas pueden ser averiguadas con facilidad si no son muy largas. Una posible opción es utilizar VNC sobre conexiones seguras basadas en SSH o en una VPN (red privada virtual). Ejemplos de aplicaciones que utilizan este protocolo:

- <https://www.realvnc.com/>.
- <http://www.tightvnc.com/>.
- <https://remotevnc.net/>.



Terminal Server

Terminal Server es una herramienta, similar a VNC, proporcionada por Microsoft en sus sistemas de tipo "server". Permite utilizar y gestionar de forma remota aplicaciones que estén instaladas en el servidor. Con esta herramienta se pueden utilizar aplicaciones complejas y "pesadas" en equipos con pocos recursos. Además resulta apropiada para labores de mantenimiento informático. Al funcionar en modo cliente-servidor, antes de utilizarlo será necesario arrancar el servicio en el servidor y, por supuesto, tener instalado el SW cliente en el equipo remoto. Las opciones de "Terminal Server" han recibido varios nombres según el entorno (S.O.) Microsoft, e incluyen servicios de escritorio remoto, acceso web, ejecución remota de aplicaciones, acceso remoto a impresoras, etc.



PcAnyWhere

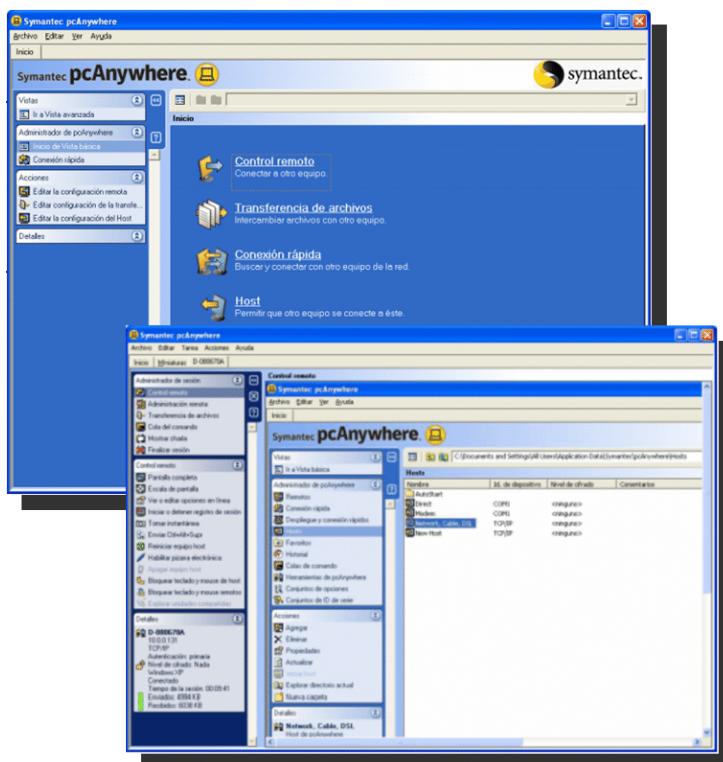
PcAnyWhere es una **suite** (conjunto de herramientas) propietaria de Norton Symantec que permite el acceso remoto a equipos y servidores de forma segura, por ejemplo para:

- Administración remota de equipos.
- Soporte y mantenimiento de servidores.
- Transferencia de archivos entre equipos.
- Trabajo desde ubicación remota.

Funciona en modo cliente-servidor y bajo varias plataformas: Windows, Linux o Mac OS X. Puede operar sobre conexiones a través de red de área amplia, vía módem en red telefónica y conexiones punto a punto (a través de puerto paralelo o serie).

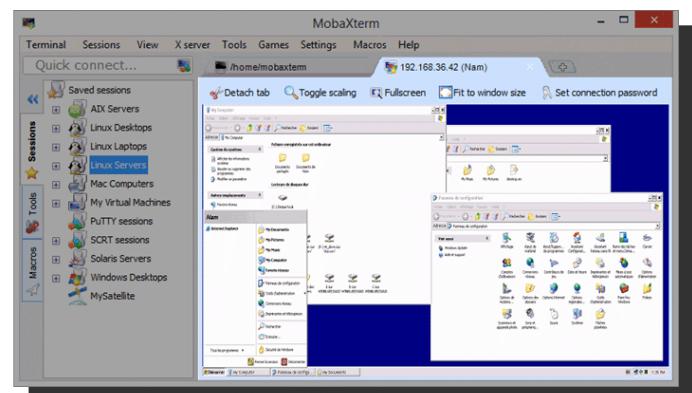
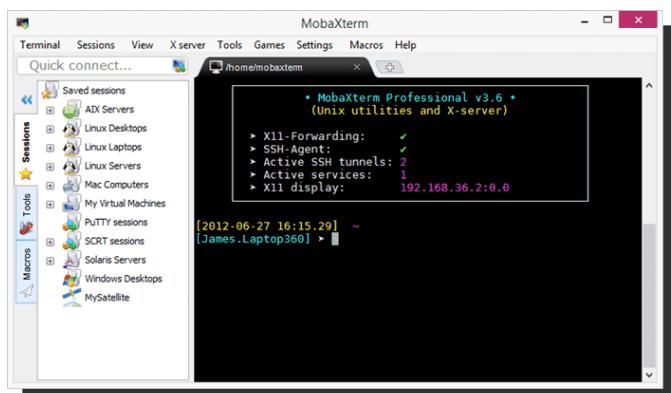
La conexión es cifrada utilizando una combinación de criptografía de clave pública y algoritmos de cifrado simétrico, que generan claves aleatorias en cada conexión.

La aplicación permite monitorizar las actividades y registrar los sucesos de la sesión, agregar información al registro de eventos de Windows y grabar las sesiones para reproducirlas posteriormente.



MobaXterm

Se trata de una **aplicación Windows** que, entre otras cosas, proporciona las principales opciones de conexión remota (SSH, X11, RDP, VNC, FTP, MOSH, ...) y la introducción de comandos Unix sobre el escritorio Windows (`bash`, `ls`, `cat`, `sed`, `grep`, `awk`, `rsync`, ...). Con ella podemos disponer de interfaz gráfica para SFTP (FTP vía SSH), Terminal X, Telnet, etc., con algunas ventajas adicionales, como el hecho de poder ejecutar el mismo comando sobre diferentes servidores. Puedes ver una “demo” en: <http://mobaxterm.mobatek.net/demo.html>. Échale un vistazo :-).



Aplicaciones tipo back-connect

TeamViewer

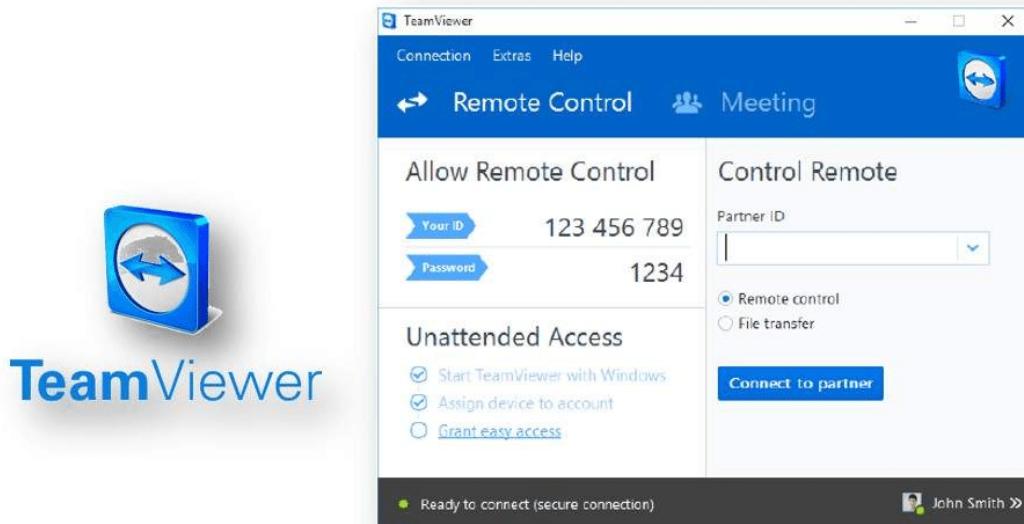
Programa útil y muy utilizado en **entorno Windows** que permite la administración de un equipo desde otro remoto. El usuario que va a ser “administrado” debe admitir la conexión y proporcionar un usuario y contraseña al administrador.

Entre otras funciones, TeamViewer puede permitir:

- Establecer conexión entre ordenadores con distintos sistemas operativos.
- Administrar servidores y estaciones de trabajo.
- Conexión desde dispositivos móviles.
- Compartir escritorio durante reuniones de trabajo y presentaciones remotas.

La aplicación permite establecer comunicaciones a través de *firewalls*, *routers* con NAT (traducción de direcciones) y *proxys* sin tener que modificar la configuración de enrutamientos de la red.

Los equipos que utilizan TeamViewer están identificados por una ID global y única, que se genera la primera vez que se inicia la aplicación. Todas las conexiones establecidas con TeamViewer están cifradas y protegidas frente a intentos de acceso de terceros.



Tiny Shell

Herramienta escrita en Python para **conectar dos máquinas Unix/Linux** (soportada por la mayoría de las plataformas de este entorno) intercambiando información en modo texto. Nos permite ejecutar comandos (mediante una interfaz básica) e intercambiar ficheros, emular terminales y establecer conexiones a menudo incluso a través de *firewalls* intermedios, al utilizar el protocolo SCTP (“Stream Control Transmission Protocol”). El programa soporta la emulación de terminales (PTY/TTY), utiliza cifrado de 8 bits y con fuerte encriptación AES de 128 bits.

Aplicaciones tipo VPN

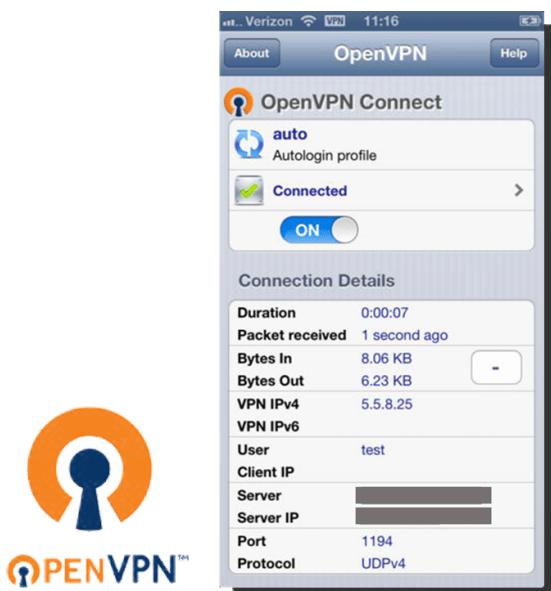
Open VPN

Open VPN es una solución de **conexión remota basada en SSL + VPN**. Es **multiplataforma**, con lo que admite conectar equipos Windows y Linux. Permite configurar VPN de forma más sencilla que otras opciones basadas en IPSec (que es prácticamente el estándar usado en VPN).

Como características podemos señalar:

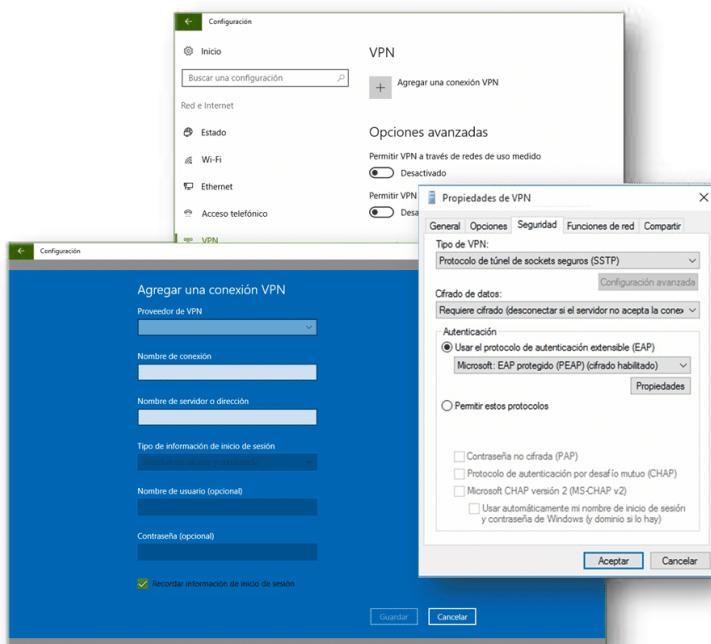
- Conexión cifrada.
- Utiliza SSL/TLS como protocolos para criptografía.
- Compatibilidad con *firewalls*, *proxys* y NAT.
- Necesita solamente un puerto del *firewall*.
- Sencilla configuración y manejo.
- Interfaces de red estandarizados.

Ver: <https://openvpn.net/>



Windows VPN

Windows nos ofrece una forma bastante sencilla de configurar una conexión VPN en el sistema operativo Windows 10 (y en algunos anteriores). A través del menú de configuración nos permite elegir la opción de crear una conexión VPN, asignarle nombre, determinar el usuario y contraseña, el protocolo de conexión (“tipo de VPN”) donde la más utilizada es PPTP (“Point to Point Tunelling Protocol”). La herramienta permite, por supuesto, el transporte de datos a través de **conexión segura** (cifrada) y autenticación de usuarios con claves que viajan encriptadas por la red. El tráfico PPTP utiliza el puerto 1723.



Resumen

Has finalizado esta lección. Hagamos un repaso de sus contenidos principales.

En esta lección has visto la gran importancia que tienen los servidores para facilitar el trabajo y aumentar el rendimiento de un conjunto de equipos conectados en red. También sabemos que hay muchos tipos de servidores, y que podemos dividirlos por la función que realizan. De esta forma hablamos de: servidores de impresión, de ficheros, de aplicaciones, etc.

Por otro lado, además de la conexión remota a un servidor, hemos visto que podemos establecer conexiones remotas con equipos de la red que no son servidores, y que tenemos varios métodos para ello y con varios niveles de seguridad. En cualquier caso, siempre es necesario que tengamos una arquitectura cliente-servidor o un software que soporte la conexión remota. Disponemos de múltiples aplicaciones y herramientas para ello.

El área de las redes y la conexión de sistemas es muy amplia y no te preocupes si te parecen muchos conceptos y conocimientos. Se trata de que tengas los conceptos generales claros y que vayas progresando poco a poco en tu aprendizaje. ¡Seguro que no te aburrirás!



PROEDUCA