



1º DAM/DAW Sistemas Informáticos

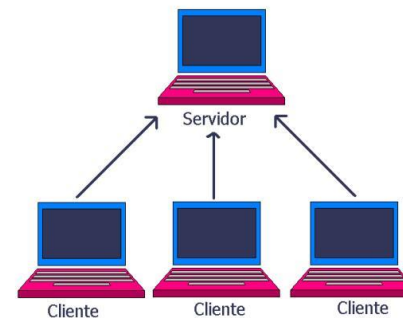
U5. Redes

4 - Servicio SSH



Modelo cliente-servidor

- La mayor parte de las aplicaciones que funcionan en un entorno de red, siguen el **modelo cliente-servidor**.
- Este modelo está formado por **dos componentes que se comunican entre sí**, la parte cliente y la parte servidor.
- Habitualmente **el cliente realiza una petición y el servidor proporciona una respuesta o resultado**.



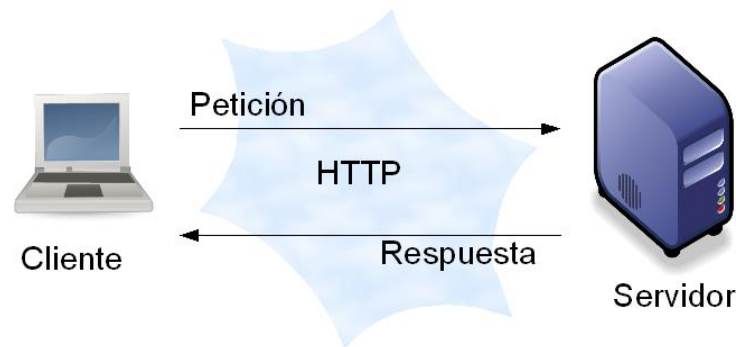
Modelo cliente-servidor

- La parte **cliente** se ejecuta en el **host o dispositivo que realiza la petición de un servicio**. Puede iniciarla un usuario o una aplicación y finaliza cuando termina el servicio.
- La parte **servidor** se ejecuta **en otro host, de la LAN o de Internet, que recibe la petición y puede ofrecer servicio a múltiples clientes**.
- Habitualmente, el objetivo es que el servidor ofrezca sus servicios de manera ininterrumpida y continuada.



Modelo cliente-servidor

- Por ejemplo, nosotros podemos tener un navegador web (Chrome, Firefox, Safari, etc.) que actúa como cliente mediante el protocolo HTTP, realizando peticiones a servidores web en ubicaciones geográficas dispares. Estos servidores web, además de procesar respuestas para nuestro cliente, también se comunican con otros clientes, ubicados también de forma dispar, que solicitan peticiones a estos sitios web.



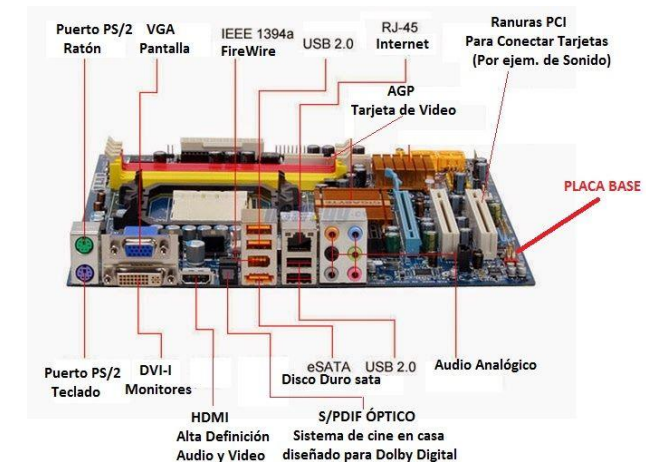
Modelo cliente-servidor. Puertos

- Se denomina **puerto** a cada uno de los tipos específicos de interfaces a través de las que se puede **enviar y/o recibir** diferentes tipos de **datos**.
- Una **interfaz** es un concepto más general que **describe cómo interactúan** dos entidades: sistemas, programas, dispositivos, componentes, ..., o un sistema y un usuario.



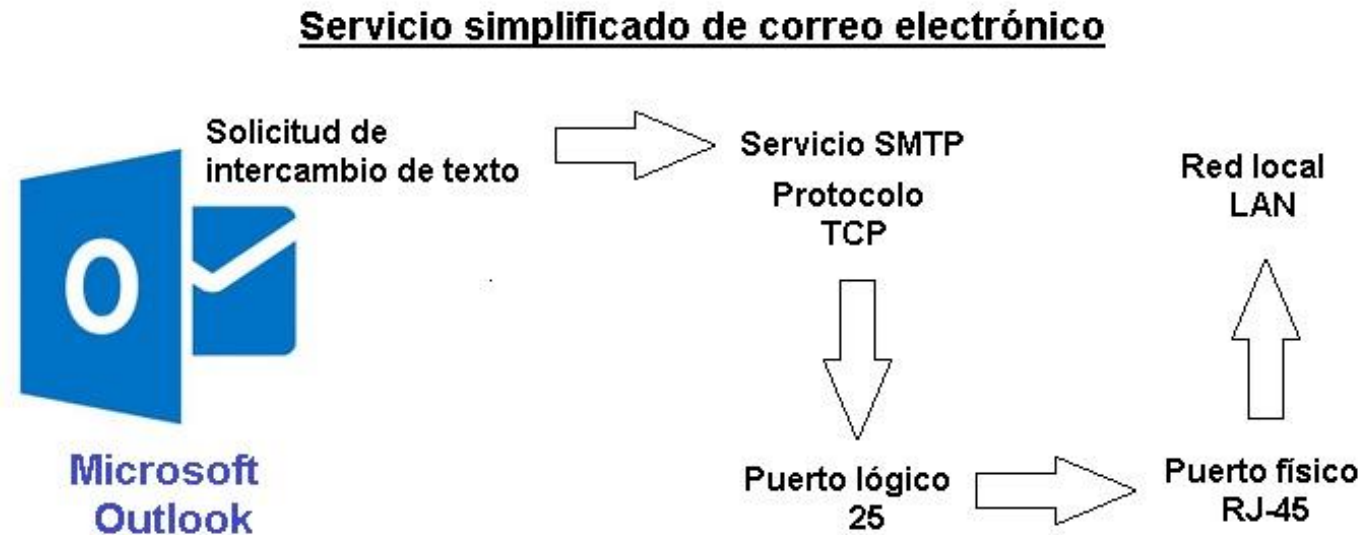
Modelo cliente-servidor. Puertos

- Los puertos pueden ser de 2 tipos:
 - Puertos físicos:** todo aquel hardware (cableado o inalámbrico) que permite una conexión física de entrada y/o salida de datos en un dispositivo.
 - Por ejemplo: USB, RJ45 (Red o LAN), PCI, HDMI,...



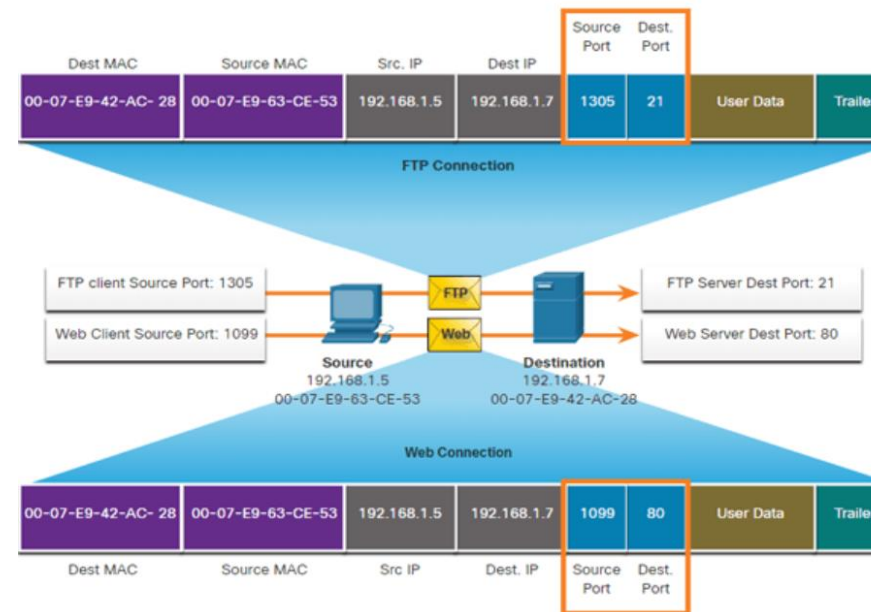
Modelo cliente-servidor. Puertos lógicos

- **Puertos lógicos:** son unas **zonas de memoria que gestiona el sistema operativo** y se usan **para el intercambio de información**. Serán de los que hablemos en esta unidad.



Modelo cliente-servidor. Puertos lógicos

- Cada uno de los diferentes procesos, petición y respuesta, que se está ejecutando en los hosts cliente y servidor, tiene asignado un número de puerto.



Modelo cliente-servidor. Puertos lógicos

- Los puertos comprendidos **entre el 0 y el 1023** se denominan **puertos conocidos**, y se asignan desde el sistema operativo, por convenio, a ciertas aplicaciones particulares o servicios de carácter universal. IANA (Internet Assigned Numbers Authority) determina dichas asignaciones.
- Los puertos comprendidos **entre el 1024 y el 49151** son **puertos accesibles** por otros procesos y usuarios.
- En el ámbito de redes, los puertos son valores que se integran en cada paquete, en la capa de transporte del modelo TCP/IP.

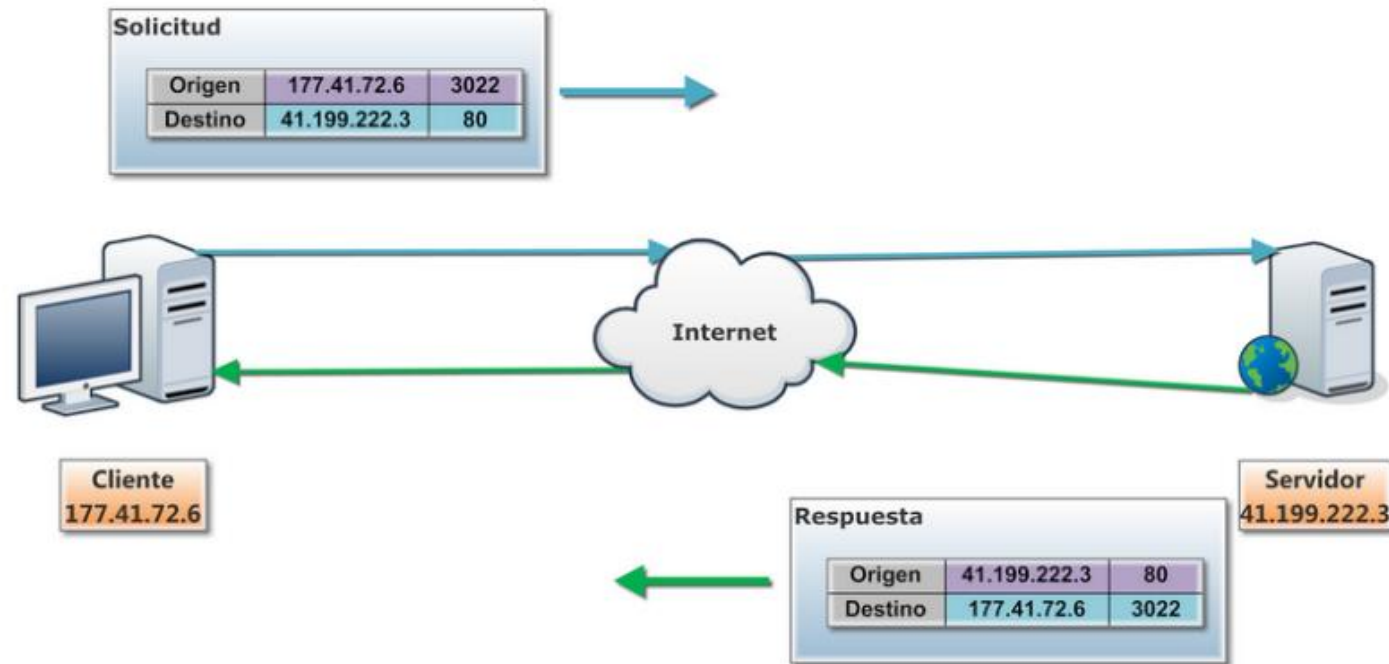
Modelo cliente-servidor. Puertos lógicos

- Unido al concepto de puerto, nos referimos como **socket de un proceso** al **par formado por la dirección IP del host donde se está ejecutando dicho proceso y el puerto del proceso**.
- Los sockets **permiten establecer conexiones virtuales** entre un proceso en ejecución en un host cliente y un proceso en ejecución en un host servidor.
- Los sockets favorecen el **intercambio de datos de forma fiable y eficiente**.

Socket = IP:puerto

Modelo cliente-servidor. Puertos lógicos

- En la imagen se ve un ejemplo, en el que tenemos un host cliente con la IP 177.41.72.6 que realiza una petición a un host servidor que tiene la IP 41.199.222.31



Modelo cliente-servidor. Puertos lógicos

- La petición HTTP se envía al número de puerto conocido 80, que es el puerto reservado para el servicio HTTP en el servidor.
- Por tanto, **el socket de este servicio es 41.199.222.3:80**
- Cuando la petición HTTP llega al host servidor, se entrega en el puerto 80, y, por tanto, será tratada por el servidor HTTP.

Modelo cliente-servidor. Puertos lógicos

- La aplicación de navegación en el host cliente tiene asignado también otro puerto de origen, en este caso, por ejemplo, el 3022.
- Cuando el servidor termina y envía la respuesta, en última instancia será entregada por el router correspondiente al host cliente en el puerto 3022.
- Por tanto, **el socket origen de esta petición es 177.41.72.6:3022**

Servicios de la capa de aplicación - FTP

- Ya hemos conocido algunos de los **servicios de la capa de aplicación del modelo TCP/IP**, como son el **servicio DNS** y el **servicio DHCP**.
- Además de estos dos servicios, existen un conjunto de servicios adicionales que usamos frecuentemente y que conviene conocer.
- El **servicio FTP (File transfer protocol)** permite a los clientes **enviar y recibir ficheros de un servidor** que esté ejecutando este servicio. Este servicio no es dependiente de ningún sistema operativo, por lo que **permite el intercambio entre distintas plataformas**.

Servicios de la capa de aplicación - FTP

- **La pila de protocolos TCP/IP incluye una utilidad FTP** que permite el uso de comandos para este servicio.
- El protocolo FTP consta principalmente y por defecto de dos puertos, **el puerto 21, utilizado para conectarse** de forma remota a un servidor y autenticarse en él y **el puerto 20**, que se utiliza para las transferencias de archivos una vez autenticado.



Servicios de la capa de aplicación - HTTP

- El protocolo **HTTP (Hypertext transfer protocol)** es el que gestiona la mayor parte del tráfico de Internet.
- Cuando un usuario solicita un recurso web, esta solicitud se realiza mediante HTTP.
- Por ejemplo, cuando accedemos a una URL, el servicio DNS resuelve la IP, y después, se envía una solicitud “get” al servidor web, el cuál devuelve un “send”, ambas operaciones usando HTTP.
- **HTTP hace uso del protocolo TCP y por defecto opera en el puerto 80.**

Servicios de la capa de aplicación - HTTP

- El protocolo **HTTPS (Hypertext transfer protocol secure)** se utiliza para realizar transacciones de datos seguras vía web.
- Este protocolo utiliza una tecnología basada en certificados digitales para asegurar una autenticación mutua entre cliente y servidor.
- Además, **HTTPS encripta todos los paquetes de datos**, lo que garantiza su confidencialidad.
- **HTTPS utiliza también TCP y por defecto opera en el puerto 443.**

Servicios de la capa de aplicación - POP3, IMAP, SMTP

- **POP3 (Post office protocol v3)** es un servicio de recepción de correo electrónico que proporciona al usuario el acceso a su carpeta de mensajes entrantes. POP3 se encarga de contactar con un servidor de correo y **descargar en un dispositivo local los mensajes recibidos** en el servidor de correo, en la cuenta del usuario.

POP3 utiliza TCP y opera por defecto en el puerto 110.

- **IMAP (Internet message access protocol)** permite acceder al servidor de correo electrónico desde cualquier dispositivo. **Mediante este protocolo los mensajes no se descargan en un dispositivo local.**

IMAP utiliza TCP y el puerto 143 por defecto.

- **SMTP (Simple mail transport protocol)** se encarga de gestionar el envío de correo electrónico. Los mensajes se envían desde un servidor SMTP a otro. Para ello, cada servidor utiliza el servicio DNS.

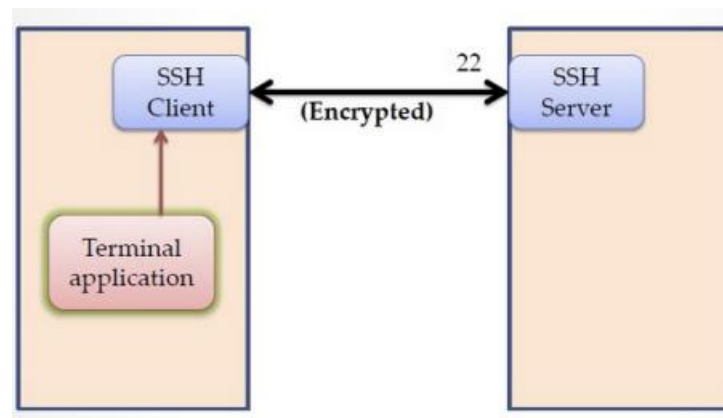
SMTP utiliza TCP y por defecto opera en el puerto 25.

Servicios de la capa de aplicación - TELNET

- **TELNET (Teletype network) permite conectar un cliente con un servidor remoto.** Este protocolo proporciona comunicación bidireccional entre cliente y servidor.
- **Nos permite acceder o iniciar sesión en otra máquina** con el objetivo de manejarla de forma remota.
- Mediante este servicio, **la información** viaja expuesta, es decir, **no viaja cifrada.**
- **Este protocolo también utiliza TCP y opera en el puerto 23 por defecto.**

Servicios de la capa de aplicación - SSH

- **SSH (Secure Shell) es un protocolo de red para establecer comunicaciones seguras entre dos hosts, siguiendo el modelo cliente-servidor.**
- Podríamos decir que es similar a TELNET, pero estableciendo una conexión segura, es decir, la información viaja cifrada.
- **Un cliente SSH se conecta con un servidor SSH a través del puerto 22.**



Servicios de la capa de aplicación - SSH

- **SSH ofrece confidencialidad e integridad de los datos** en redes inseguras, como puede ser Internet.
- La aplicación más común de este servicio es el **acceso remoto al shell de sistemas Linux**.
- Es decir, acceder a otras máquinas Linux a través de la red y trabajar con ellas como si estuviésemos en local.
- También ofrece más funcionalidades, como la transferencia de ficheros (SFTP).

Servicios de la capa de aplicación - SSH - Parte servidor

- Para usar este servicio, lo haremos mediante la herramienta OpenSSH.
- La podemos instalar mediante:

sudo apt-get install openssh-server

- La herramienta SSH se instala en **/etc/ssh**, donde podemos encontrar los siguientes ficheros:

sshd_config: archivo de configuración del servidor SSH

ssh_config: archivo de configuración del cliente SSH

ssh_host_*_key: clave privada de la máquina (* puede ser rsa, dsa o ecdsa)

ssh_host_*_key.pub: clave pública de la máquina (idem a anterior).

Servicios de la capa de aplicación - SSH - Parte servidor

- Al instalar OpenSSH en Ubuntu, el servicio SSH arranca automáticamente.
- Si el servicio SSH estuviera detenido, necesitaremos arrancarlo para que escuche las peticiones entrantes ejecutando:

sudo service ssh start

- Otras opciones que podemos utilizar sobre este servicio son:
 - **# sudo service ssh stop** (para pararlo)
 - **# sudo service ssh status** (para ver el estado)
 - **# sudo service ssh restart** (para reiniciarlo).

Servicios de la capa de aplicación - SSH - Parte servidor

- Además de “**# sudo service ssh status**”, podemos confirmar que el servicio SSH está escuchando peticiones de mediante la ejecución de:

\$ netstat -ltu

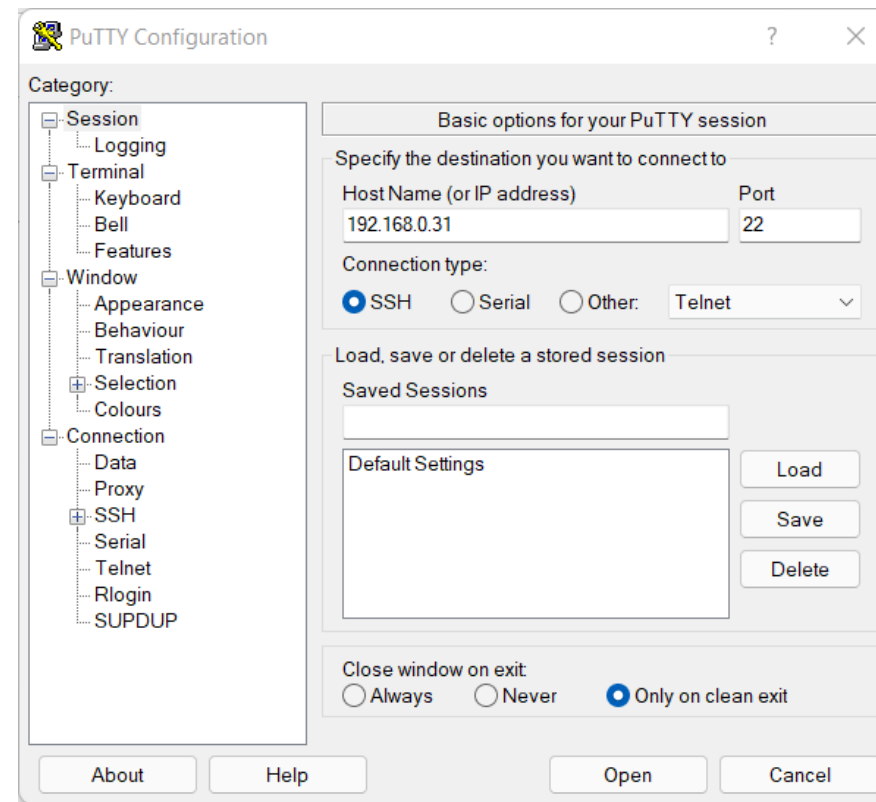
```
pmartinez@Ubuntu20:~$ netstat -ltu
Conexiones activas de Internet (solo servidores)
Proto  Recib Enviad Dirección local      Dirección remota      Estado
tcp     0      0 localhost:domain     0.0.0.0:*             ESCUCHAR
tcp     0      0 0.0.0.0:ssh          0.0.0.0:*             ESCUCHAR
tcp     0      0 localhost:ipp        0.0.0.0:*             ESCUCHAR
tcp6    0      0 [::]:ssh             [::]:*                ESCUCHAR
tcp6    0      0 ip6-localhost:ipp    [::]:*                ESCUCHAR
udp     0      0 localhost:domain     0.0.0.0:*             ESCUCHAR
udp     0      0 0.0.0.0:631          0.0.0.0:*             ESCUCHAR
udp     0      0 0.0.0.0:55974        0.0.0.0:*             ESCUCHAR
udp     0      0 0.0.0.0:mdns         0.0.0.0:*             ESCUCHAR
udp6    0      0 [::]:41507           [::]:*                ESCUCHAR
udp6    0      0 [::]:mdns            [::]:*                ESCUCHAR
pmartinez@Ubuntu20:~$
```


Servicios de la capa de aplicación - SSH

- Es interesante destacar que, a nivel práctico, podemos entender que el **servicio SSH es multiplataforma**. Es decir, podríamos tener cualquier plataforma tanto en el host servidor como en el host cliente.
- El host cliente puede acceder al host servidor mediante el uso de consola y comandos, aunque también existen aplicaciones en modo gráfico.
- Inicialmente vamos a pensar en este esquema:
 - **Servidor: máquina virtual Linux.**
 - **Cliente: nuestra máquina física.** Modo gráfico (Windows, Mac, etc.).

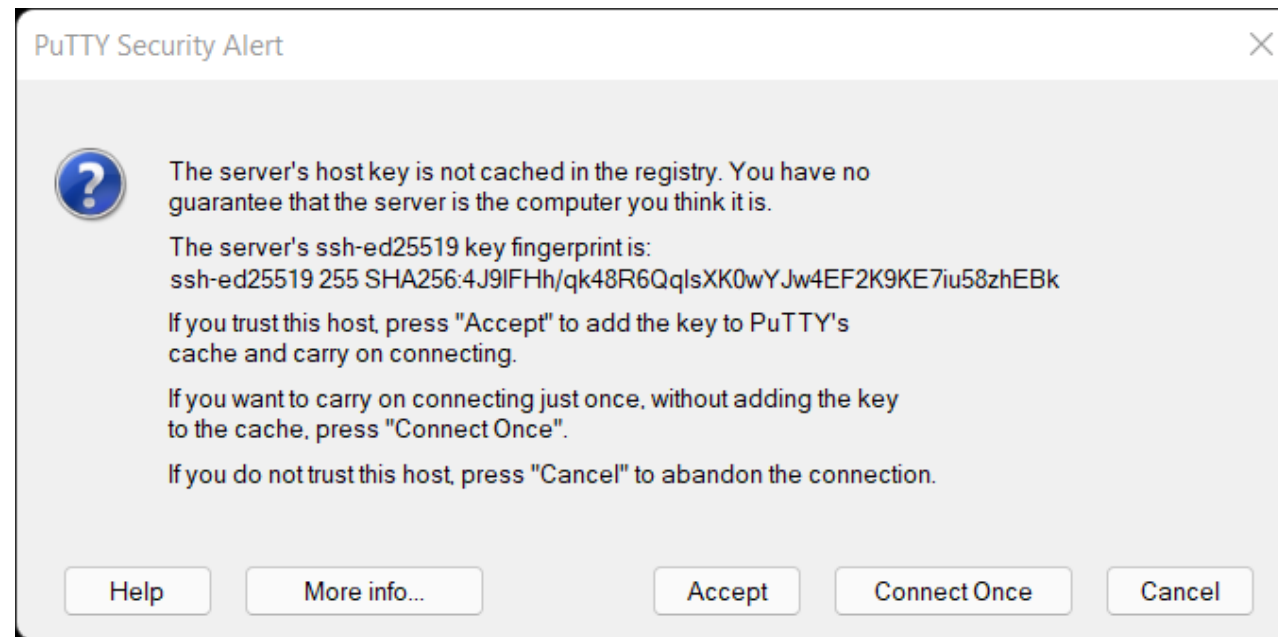
Servicios de la capa de aplicación - SSH - Parte cliente

- Por ejemplo, podemos usar la aplicación **PuTTY** para establecer la conexión desde el host cliente:



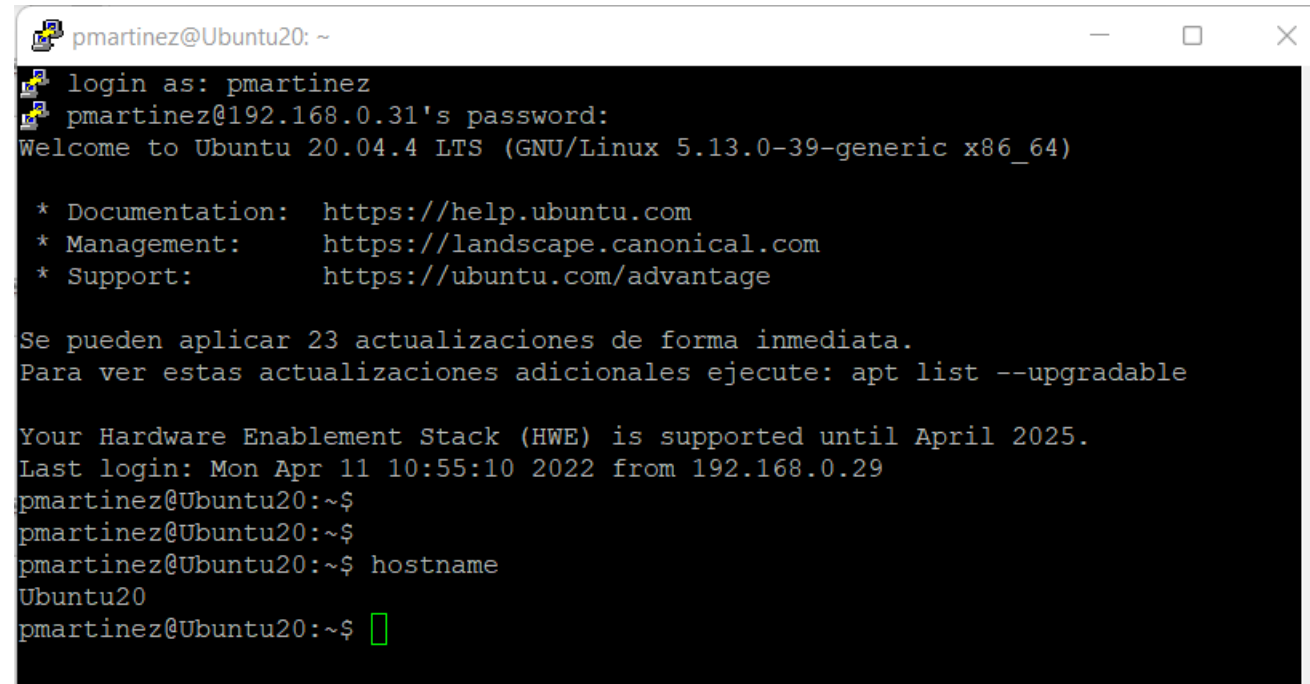
Servicios de la capa de aplicación - SSH - Parte cliente

- Nos mostrará una alerta de seguridad indicando que no tenemos registrada la clave de seguridad del servidor. Si aceptar la clave quedará registrada.



Servicios de la capa de aplicación - SSH - Parte cliente

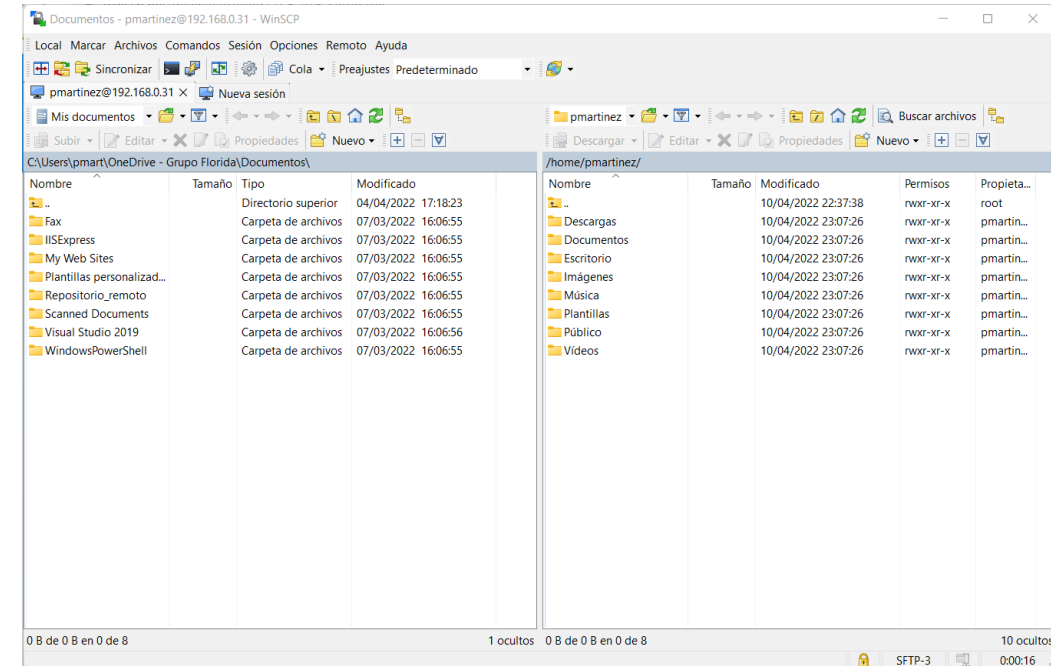
- Después de validarnos en el sistema remoto, nos da acceso a una sesión de terminal, desde la que podemos operar en un host servidor, independientemente de dónde esté instalado.



```
pmartinez@Ubuntu20: ~  
login as: pmartinez  
pmartinez@192.168.0.31's password:  
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-39-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
Se pueden aplicar 23 actualizaciones de forma inmediata.  
Para ver estas actualizaciones adicionales ejecute: apt list --upgradable  
  
Your Hardware Enablement Stack (HWE) is supported until April 2025.  
Last login: Mon Apr 11 10:55:10 2022 from 192.168.0.29  
pmartinez@Ubuntu20:~$  
pmartinez@Ubuntu20:~$  
pmartinez@Ubuntu20:~$ hostname  
Ubuntu20  
pmartinez@Ubuntu20:~$
```

Servicios de la capa de aplicación - SSH - Parte cliente

- **WinSCP** es una utilidad, entre muchas otras, que nos ofrece la cómoda posibilidad de **manipular o copiar archivos desde un host cliente a un host servidor y viceversa**, en función de nuestros permisos y previa conexión SSH.
- Realmente se trata de una capa gráfica, que ejecuta una serie de comandos internamente.



Servicios de la capa de aplicación - SSH - Comando conexión

- Podemos realizar las funciones descritas anteriormente en una conexión SSH, mediante el uso de la **línea de comandos**, por ejemplo:

- **Iniciar conexión ssh:**

\$ ssh [usuario@]IP_ServidorSSH

- El usuario es opcional y si no se indica, el sistema intenta acceder con el usuario de la sesión activa en el cliente.
- Nos pedirá la contraseña del usuario.
- **Se nos abre una sesión de consola en el host servidor.**

Servicios de la capa de aplicación - SSH - Comando conexión

- **Ejemplo:** datos del host servidor SSH

```
pmartinez@Ubuntu20:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.0.27  netmask 255.255.255.0  broadcast 192.168.0.255
    inet6 fe80::bdef:6b9d:bdc5:2ead  prefixlen 64  scopeid 0x20<link>
    ether 08:00:27:bd:d2:bc  txqueuelen 1000  (Ethernet)
    RX packets 233  bytes 122052 (122.0 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 240  bytes 38663 (38.6 KB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Bucle local)
    RX packets 194  bytes 17305 (17.3 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 194  bytes 17305 (17.3 KB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pmartinez@Ubuntu20:~$
```

Servicios de la capa de aplicación - SSH - Comando conexión

- **Ejemplo:** datos del host cliente

```
pmartinez@Ubuntu12: ~  
pmartinez@Ubuntu12:~$ ifconfig  
eth0      Link encap:Ethernet  direcciónHW 08:00:27:49:13:24  
          Direc. inet:192.168.0.28  Difus.:192.168.0.255  Másc:255.255.255.0  
          Dirección inet6: fe80::a00:27ff:fe49:1324/64 Alcance:Enlace  
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST  MTU:1500  Métrica:1  
          Paquetes RX:912 errores:0 perdidos:0 overruns:0 frame:0  
          Paquetes TX:510 errores:0 perdidos:0 overruns:0 carrier:0  
          colisiones:0 long.colaTX:1000  
          Bytes RX:462052 (462.0 KB)  TX bytes:206656 (206.6 KB)  
  
lo        Link encap:Bucle local  
          Direc. inet:127.0.0.1  Másc:255.0.0.0  
          Dirección inet6: ::1/128 Alcance:Anfitrión  
          ACTIVO BUCLE FUNCIONANDO  MTU:65536  Métrica:1  
          Paquetes RX:120 errores:0 perdidos:0 overruns:0 frame:0  
          Paquetes TX:120 errores:0 perdidos:0 overruns:0 carrier:0  
          colisiones:0 long.colaTX:0  
          Bytes RX:10612 (10.6 KB)  TX bytes:10612 (10.6 KB)  
  
pmartinez@Ubuntu12:~$
```

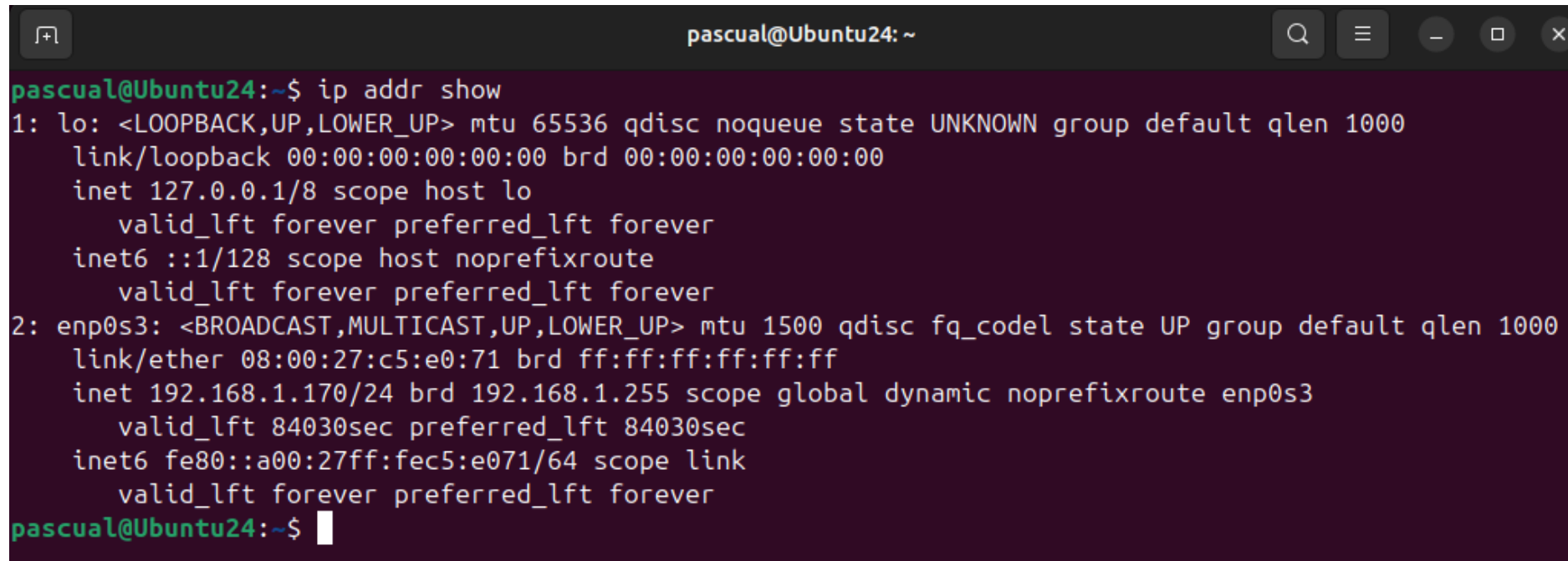

Servicios de la capa de aplicación - SSH - Comando conexión

- **Ejemplo:** conexión desde un cliente Linux a un servidor Linux, mediante el comando ssh (el prompt me indica el usuario y el host de la sesión activa en cada momento...)

```
pmartinez@Ubuntu20: ~  
pmartinez@Ubuntu12:~$ ssh pmartinez@192.168.0.27  
pmartinez@192.168.0.27's password:  
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-40-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
Se pueden aplicar 45 actualizaciones de forma inmediata.  
15 de estas son actualizaciones de seguridad estándares.  
Para ver estas actualizaciones adicionales ejecute: apt list --upgradable  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
Your Hardware Enablement Stack (HWE) is supported until April 2025.  
Last login: Fri Apr 29 16:20:54 2022 from 192.168.0.28  
pmartinez@Ubuntu20:~$
```

Servicios de la capa de aplicación - SSH - Comando conexión

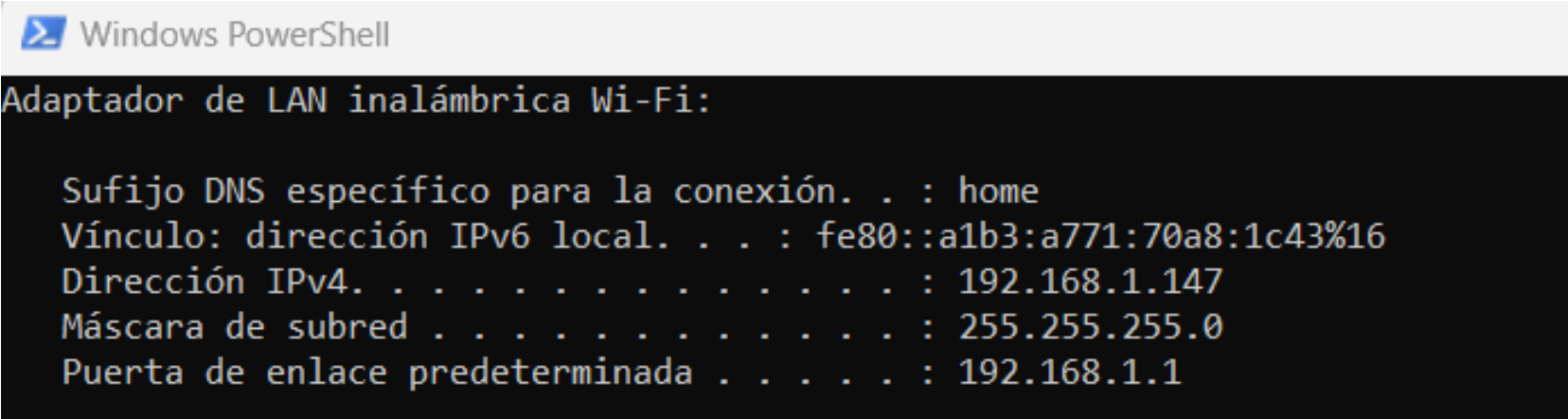
- **Otro ejemplo:** datos del host servidor SSH



```
pascual@Ubuntu24: ~  
pascual@Ubuntu24:~$ ip addr show  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
    inet 127.0.0.1/8 scope host lo  
        valid_lft forever preferred_lft forever  
    inet6 ::1/128 scope host noprefixroute  
        valid_lft forever preferred_lft forever  
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000  
    link/ether 08:00:27:c5:e0:71 brd ff:ff:ff:ff:ff:ff  
    inet 192.168.1.170/24 brd 192.168.1.255 scope global dynamic noprefixroute enp0s3  
        valid_lft 84030sec preferred_lft 84030sec  
    inet6 fe80::a00:27ff:fec5:e071/64 scope link  
        valid_lft forever preferred_lft forever  
pascual@Ubuntu24:~$
```

Servicios de la capa de aplicación - SSH - Comando conexión

- **Otro ejemplo:** datos del host cliente



```
Windows PowerShell
Adaptador de LAN inalámbrica Wi-Fi:

Sufijo DNS específico para la conexión. . : home
Vínculo: dirección IPv6 local. . . : fe80::a1b3:a771:70a8:1c43%16
Dirección IPv4. . . . . : 192.168.1.147
Máscara de subred . . . . . : 255.255.255.0
Puerta de enlace predeterminada . . . . . : 192.168.1.1
```

Servicios de la capa de aplicación - SSH - Comando conexión

- **Otro ejemplo:** conexión desde el cliente Windows al servidor Linux mediante el comando ssh

```
pascual@Ubuntu24: ~  
PS C:\Users\pmart> ssh pascual@192.168.1.170  
pascual@192.168.1.170's password:  
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-52-generic x86_64)
```

Servicios de la capa de aplicación - SSH - Comando scp

- **Comando scp:**
 - **Copia segura de ficheros** entre hosts. El **comando scp** se utiliza para copiar ficheros de forma segura entre diferentes hosts, conectados entre sí mediante el protocolo SSH.

\$ scp origen destino

*Donde, tanto origen como destino, pueden ser una **ruta local o remota**.*

Servicios de la capa de aplicación - SSH - Comando scp

- Para acceder a un origen o destino **local**, sólo tengo que indicar la **ruta** del sistema de archivos local.
- Para acceder a un origen o destino **remoto**, tengo que indicar un **usuario**, una **IP** o nombre de host y la **ruta** del sistema de archivos remoto.



Servicios de la capa de aplicación - SSH - Comando scp

- Descripción de una **ruta remota**:

[usuario_remoto@]IP_host_remoto:ruta_remota

Ejemplo para copiar un fichero desde local a remoto:

```
$ scp origen.txt usuario@192.168.1.31:/home/pepe/destino.txt
```

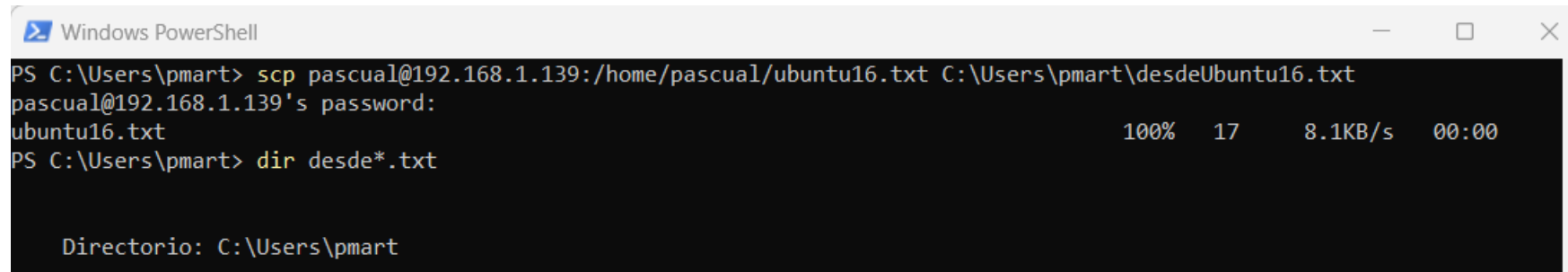
Ejemplo inverso, desde remoto a local:

```
$ scp usuario@192.168.1.31:/home/pepe/origen.txt destino.txt
```

Servicios de la capa de aplicación - SSH - Comando scp

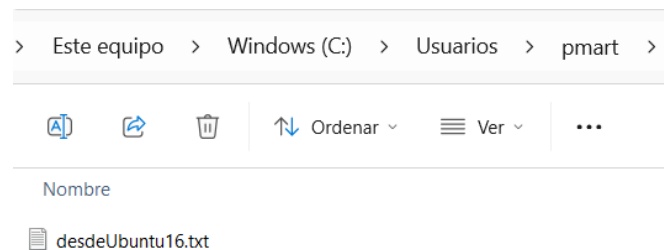
- Desde Windows como origen:
 - Desde una consola en Windows, por ejemplo, Powershell.

scp usuario@servidorSSH:/rutaOrigen/nombreOrigen.xxx c:\rutaDestino\ficheroDestino.xxx



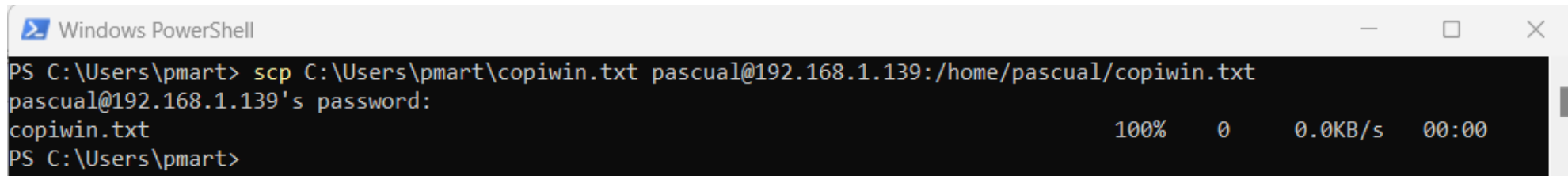
```
Windows PowerShell
PS C:\Users\pmart> scp pascual@192.168.1.139:/home/pascual/ubuntu16.txt C:\Users\pmart\desdeUbuntu16.txt
pascual@192.168.1.139's password:
ubuntu16.txt                                100% 17      8.1KB/s   00:00
PS C:\Users\pmart> dir desde*.txt

Directorio: C:\Users\pmart
```

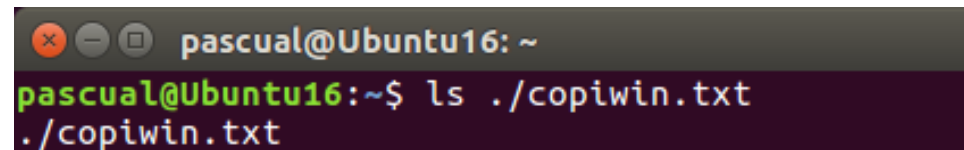


Servicios de la capa de aplicación - SSH - Comando scp

scp C:\rutaOrigen\ficheroOrigen.xxx usuario@servidorSSH:/rutaDestino/ficheroDestino.xxx



```
Windows PowerShell
PS C:\Users\pmart> scp C:\Users\pmart\copiwin.txt pascual@192.168.1.139:/home/pascual/copiwin.txt
pascual@192.168.1.139's password:
copiwin.txt                                100%   0   0.0KB/s   00:00
PS C:\Users\pmart>
```



```
pascual@Ubuntu16: ~
pascual@Ubuntu16:~$ ls ./copiwin.txt
./copiwin.txt
```