UD5. Servicio de transferencia de ficheros

Protocolo SSH

Despliegue de Aplicaciones Web

2º DAW

íNDICE

1. INTRODUCCIÓN

- En la mayoría de los casos, los **servidores y máquinas** que **necesitamos gestionar** para **desplegar** nuestras **aplicaciones** no comparten el mismo espacio físico.
  - Diferentes **departamentos de una empresa**
  - Centros de **proceso de Datos (CPD)**
  - Proveedores de infraestructura
    - Computación **en la nube.**
    - **Servidores Dedicados.**
    - Servicios de **Hosting.**

1. INTRODUCCIÓN

- El despliegue de una aplicación implica la **realización y automatización** de una serie de **tareas** como son:
  - Gestión del **logs**.
  - **Copias** de **seguridad** de las base de datos.
  - **Compilación** de la **aplicación** (Lenguajes compilados).
  - Establecimiento de permisos

NECESITAMOS **HERRAMIENTAS** PARA
GESTIONAR EL **SERVIDOR REMOTO**

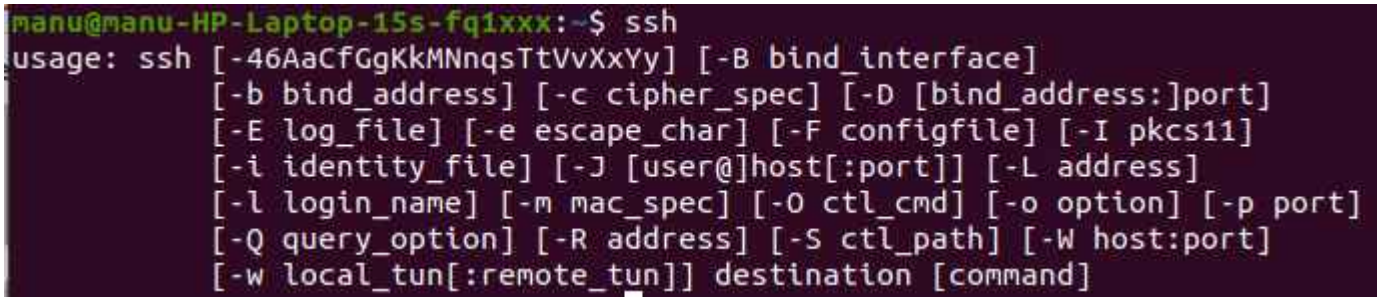
1. INTRODUCCIÓN

- El **protocolo FTP solo permite la transferencia de archivos** entre dispositivos remotos.
  - Además, presenta grandes **problemas** en cuanto a la **seguridad** de los datos y de **gestión** de las **conexiones** (Utiliza diferentes puertos para el control y la transmisión de información)

1. INTRODUCCIÓN

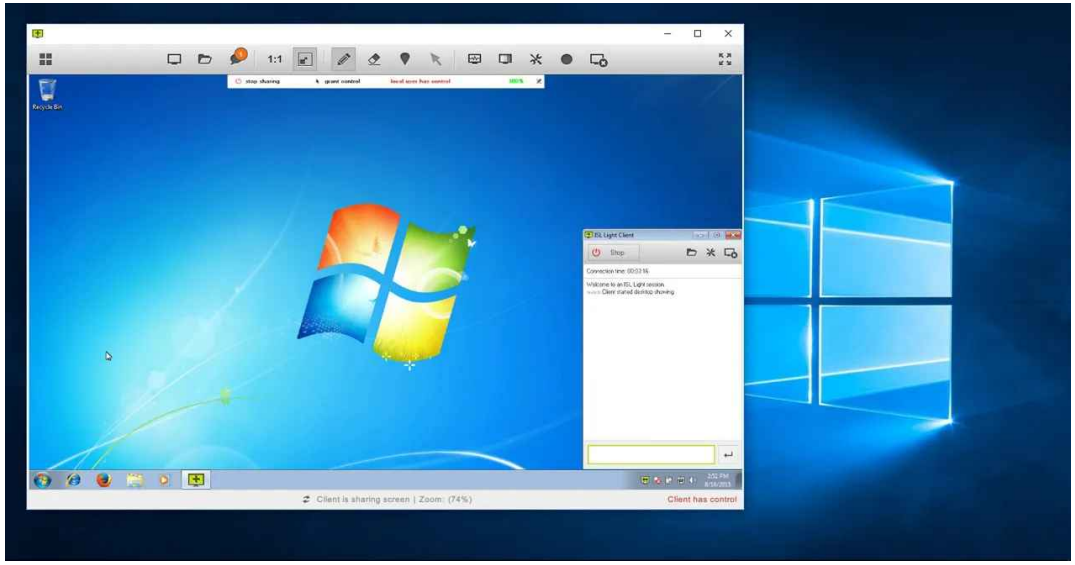
Podemos diferenciar **2 tipos de herramientas:**

- **Modo Texto:** permiten administrar el servidor mediante una consola
  - Telnet, rlogin, **Secure Shell (SSH)**



1. INTRODUCCIÓN

- **Modo gráfico:** permiten administrar el servidor mediante interfaz gráfica
  - VNC en entornos Unix **Gnu/Linux** y los servicios de **Terminal Server** en Windows.



1. INTRODUCCIÓN

- Las **conexiones remotas** mediante **línea de comandos** son la **opción más versátil** y que **menos recursos** necesita.
- Las principales herramientas son aquellas que hacen uso del protocolo **telnet** y del protocolo **SSH**
  - El protocolo **telnet** no realiza ningún tipo de cifrado, por lo que **ha dejado de utilizarse** en favor del **protocolo SSH**

2. PROTOCOLO SSH

- Permite establecer **conexiones seguras** entre **equipos conectados** mediante una **red insegura como puede ser Internet**
  - Basado en arquitectura **cliente-servidor**

2. PROTOCOLO SSH

Características principales

- Utiliza el **puerto 22** TCP **para** el **establecimiento** de las **conexiones**.
- Una vez **establecida la conexión** podremos ejecutar las órdenes como si se tratara de una **terminal local**.
- Está implementado para la mayoría de sistemas operativos existentes; Windows, linux, OSX,...
- Proporciona mecanismo para asegurar la **autenticación**, **integridad** y **confidencialidad de la información**

2. PROTOCOLO SSH

Características principales

- **Autenticación:** Después de la primera conexión, el cliente puede conocer que se está conectando al mismo servidor en futuras sesiones: conexiones de confianza.
- **Confidencialidad:** Todos los datos que se envían y se reciben durante la conexión son cifrados.
- **Integridad:** Proporciona mecanismos para asegurar que la información enviada no es manipulada por un tercero

2.1 FUNCIONAMIENTO

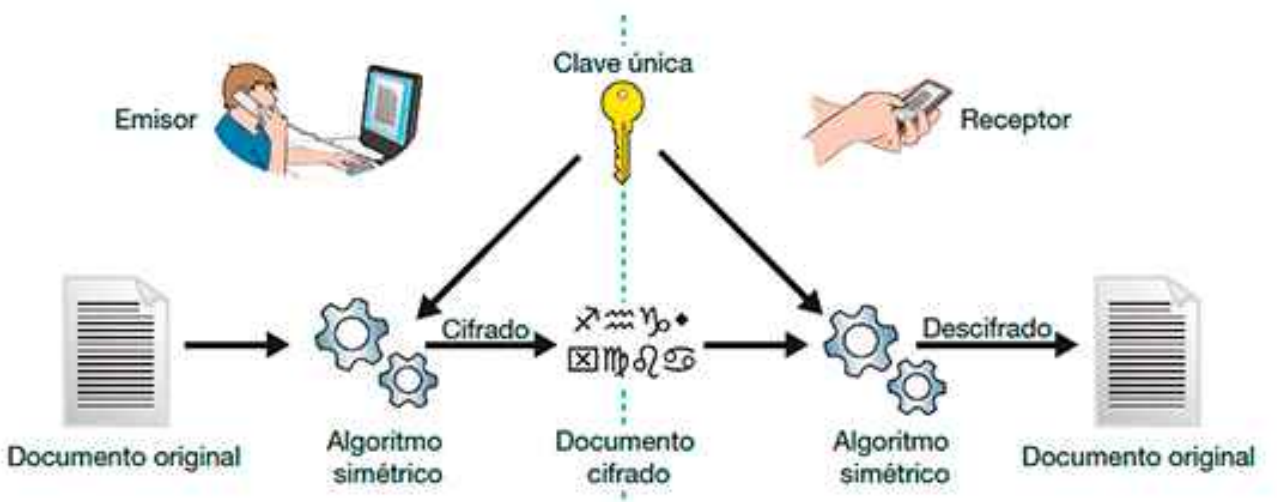
¿Qué es la Criptografía?

- Técnica utilizada para **convertir un texto claro** en otro cuyo **contenido** es igual al anterior pero **solo puede ser decodificado** por personas autorizadas.
- SSH utiliza **varios algoritmos** de encriptación y autenticación.
  - Para **establecer la conexión** con la máquina remota utiliza algoritmos de **encriptación asimétrica**.
  - Para **la transferencia de datos** utiliza algoritmos de **encriptación simétrica**, que son más rápidos.

Similar al **SSL/TLS** estudiado en la unidad anterior

2.2 CRIPTOGRAFÍA SIMÉTRICA

- **Los** algoritmos de criptografía simétrica **son los que utilizan la misma clave** tanto para el **proceso de cifrado** como para el descifrado del mensaje.
  - Los más utilizados: **DES, 3DES, AES, IDEA y Blowfish**

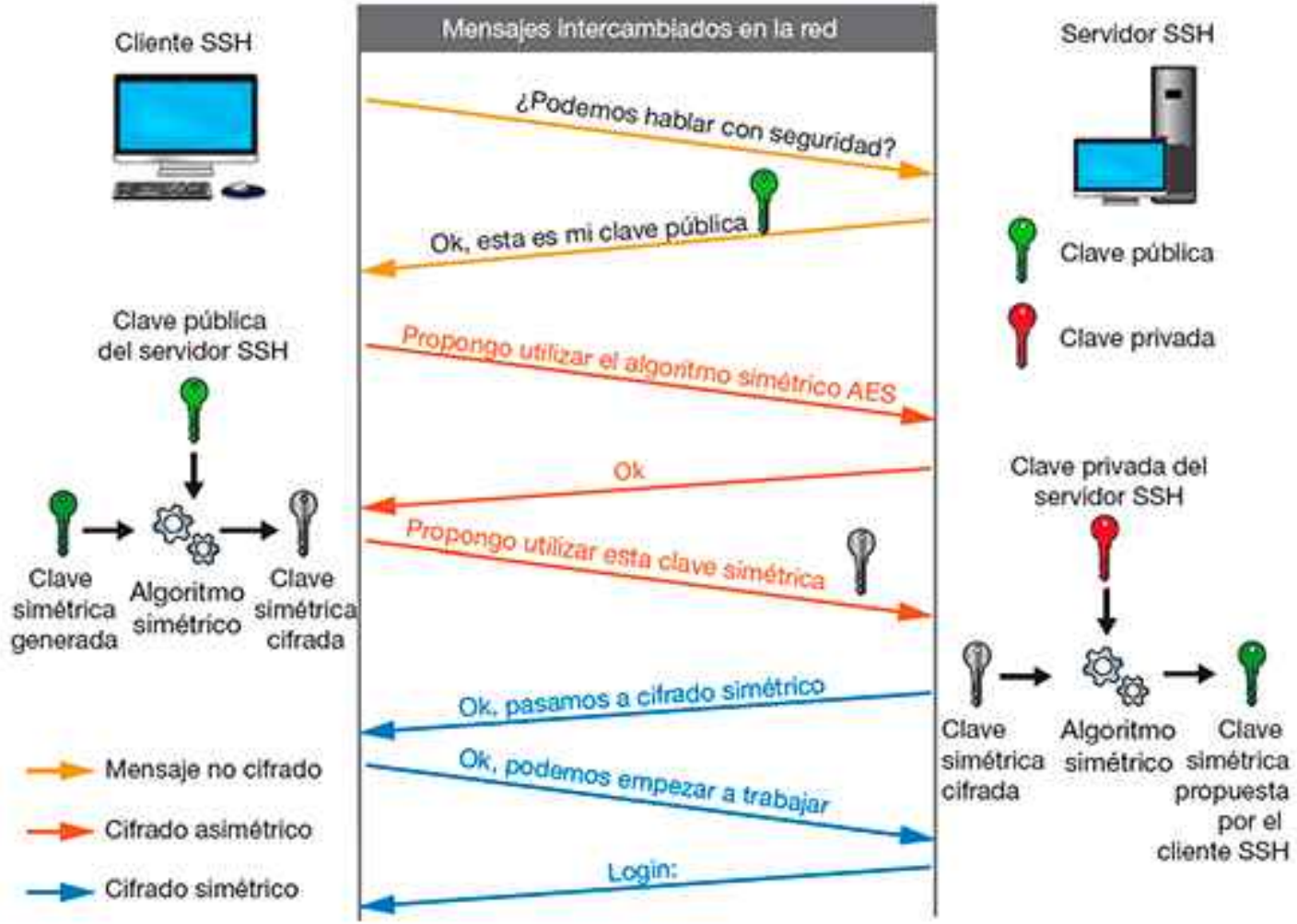


Problema: Intercambio de claves

2.3 CRIPTOGRAFÍA ASIMÉTRICA

- Utiliza **dos claves** matemáticamente **relacionadas** de manera que **lo que ciframos con una** (clave pública) **sólo puede descifrarse con la segunda** (clave privada).
  - Algunos **algoritmos** representativos son: RSA, y DSA

2.4 ESTABLECIMIENTO DE LA CONEXIÓN



2.4 ESTABLECIMIENTO DE LA CONEXIÓN

ies severo ochoa

1. El cliente abre una conexión **TCP** en el **puerto 22** del servidor.
2. Cliente y servidor negocian qué **versión SSH** van a utilizar y determinan el algoritmo de **criptografía simétrica** a utilizar.
3. El **servidor envía** su **clave pública** al cliente:
   - ➔ Si es la primera vez la guardará para futuras conexiones.
   - ➔ En caso contrario la compara con la que ya tenía guardadas para la IP de conexión **(Autenticación del servidor)**

2.4 ESTABLECIMIENTO DE LA CONEXIÓN

4. El **cliente** genera **una clave de sesión** aleatoria mediante el **algoritmo seleccionado** en el **primer paso y** la **envía al servidor** cifrando con la clave pública del mismo.
5. **El resto de comunicaciones** se hará **utilizando** esta **clave compartida** y será indescifrable.
6. A partir de aquí se llevará a cabo **la autenticación del usuario** y la **transmisión de información/órdenes** a ejecutar en la máquina remota

3. AUTENTICACIÓN DEL CLIENTE

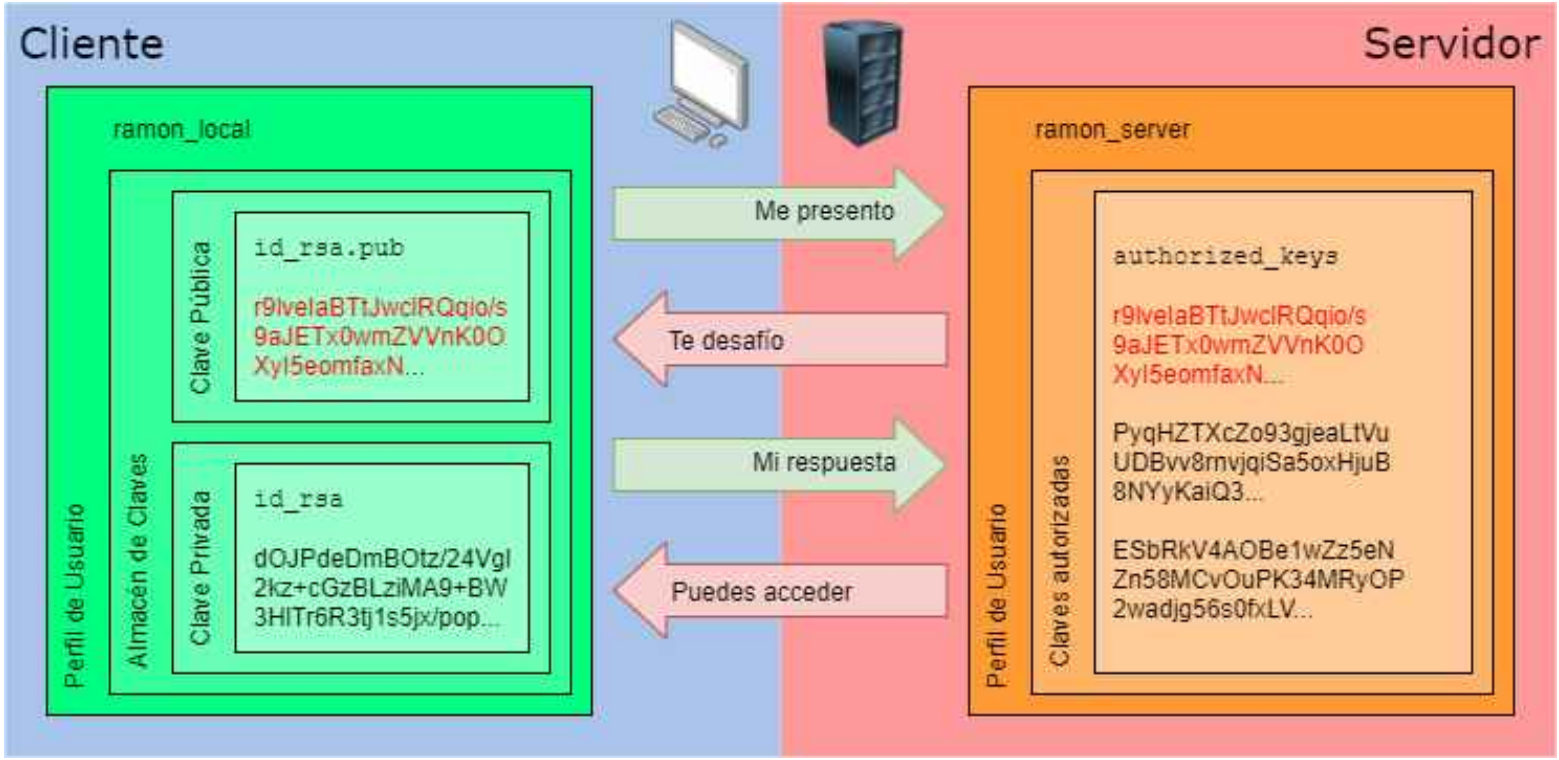
- Puede llevarse a cabo de **2 formas**:
  - **Usuario y contraseña:** Aunque las credenciales van cifradas, es menos seguro y menos recomendado ya que es más propenso a ataques de fuerza bruta.
    - Tendremos que utilizar sistemas de detección de intrusos como **fail2ban**.

3. AUTENTICACIÓN DEL CLIENTE

- Clave **pública/privada:**
  - Cuando el cliente se conecta al servidor, le informa de **la clave pública que va a utilizar**.
  - Si el servidor la tiene en **su almacén de claves autorizadas** (*authorized_keys*), enviará un mensaje aleatorio cifrado con la clave pública que, si el cliente es capaz de descifrar, quedará autenticado.
  - El **cliente decodificará el mensaje** con la clave privada **y lo enviará al servidor** decodificado.

3. AUTENTICACIÓN DEL CLIENTE

- Clave **publica/privada:**



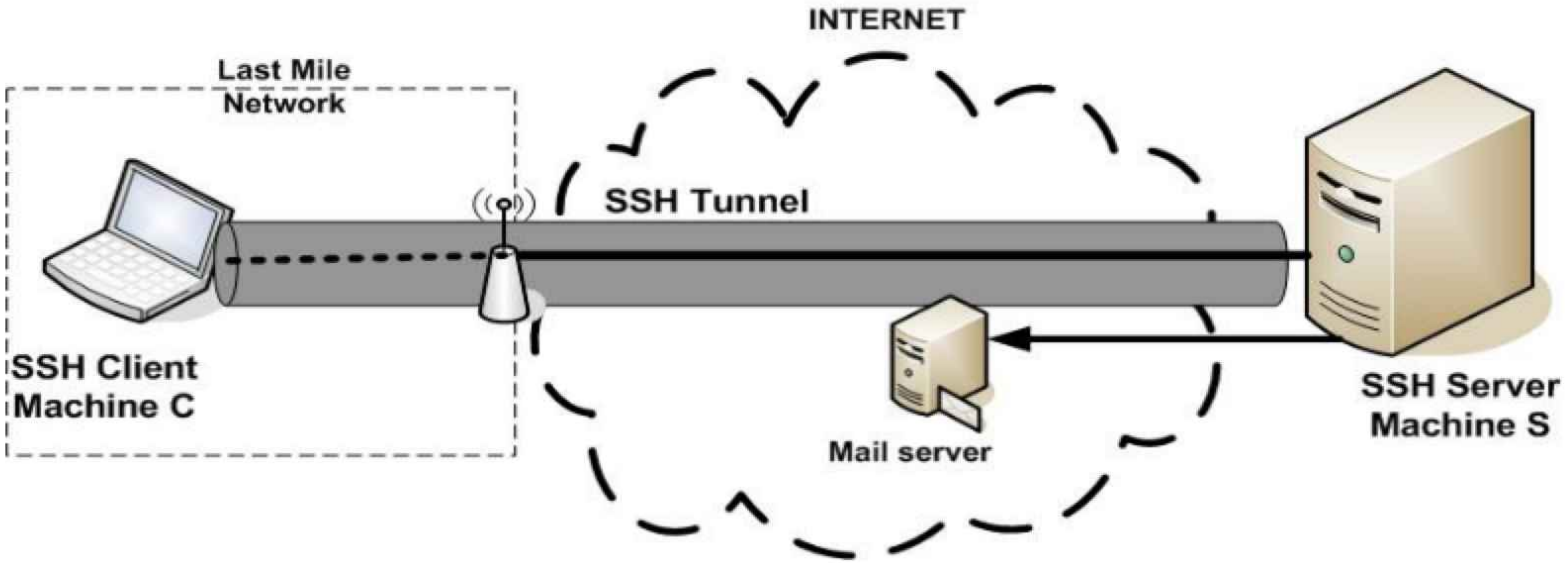
4. ¿QUÉ SON LOS TÚNELES SSH?

ies severo ochoa

- La **mayoría de protocolos** que se emplean en las comunicaciones están **basados en diseños** de hace **más de 30 años**, cuando la seguridad en redes no era un problema.
  - Telnet, FTP, POP3, SMTP
- **SSH** es que permite el establecimiento de **conexiones** ofreciendo **soporte seguro** a **cualquier protocolo que funcione sobre TCP.**

4. ¿QUÉ SON LOS TÚNELES SSH?

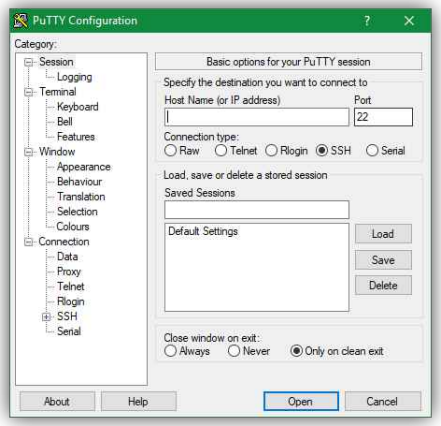
- Hace uso del **port-forwarding:** toma los datos que el cliente envía en un extremo del túnel y los reenvía por el canal seguro.
- En el **otro extremo** donde se recogen los datos y se envían al servicio de destino



5. CLIENTES SSH

- Podemos classificarlos en: **clientes gráficos** y de **consola.**
  - Putty
  - Kitty
  - CMDer
  - Los sistemas operativos LIKE-UNIX disponen de un cliente **ssh** en **modo consola pre-instalado**

```
manu@manu-HP-Laptop-15s-fq1xxx:~$ ssh
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]
           [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
           [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
           [-i identity_file] [-J [user@]host[:port]] [-L address]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
           [-w local_tun[:remote_tun]] destination [command]
```



5. CLIENTES SSH

Ejemplos de uso cliente consola

- Conexión a una máquina remota:

```
ssh usuario@ejemplo.servidor.es
```

- Ejecutar una orden sin conectarse:

```
ssh ejemplo.servidor.es comando
ssh serdis.dis.ulpgc.es ls ./
```

6. IMPLEMENTACIONES

OpenSSH

- Se trata de un proyecto de **código abierto** y **licencia libre** para su utilización para cualquier propósito.
- Es compatible con los protocolos **SSH1** (No seguro) y **SSH2**.
- Está disponible para plataformas **Gnu/linux** y **Windows**, así como **Unix, Mac, Solaris** o **AIX**

6. IMPLEMENTACIONES

Funcionalidades adicionales

- Permite la **copia de archivos** de la máquina local a la máquina remota o a la inversa mediante securecopy (**SCP).**
  - **Transferir un archivo local a un sistema remoto:**

    ```
    scp archivo_local usuario@servidor: / archivo_remoto
    ```
  - **Transferir un archivo remoto a un sistema local:**

    ```
    scp usuario@servidor:/archivo_remoto /archivo_loal
    ```
  - **Especificar múltiples archivos**

    ```
    scp/dir_local/* usuario@servidor:/dir_remoto/
    ```

6. IMPLEMENTACIONES

Funcionalidades adicionales

- Incluye **soporte completo para SFTP (SSHftp):** Sin embargo, se trata de un protocolo totalmente diferente a FTP, y está disponible a partir de la versión 2.5.

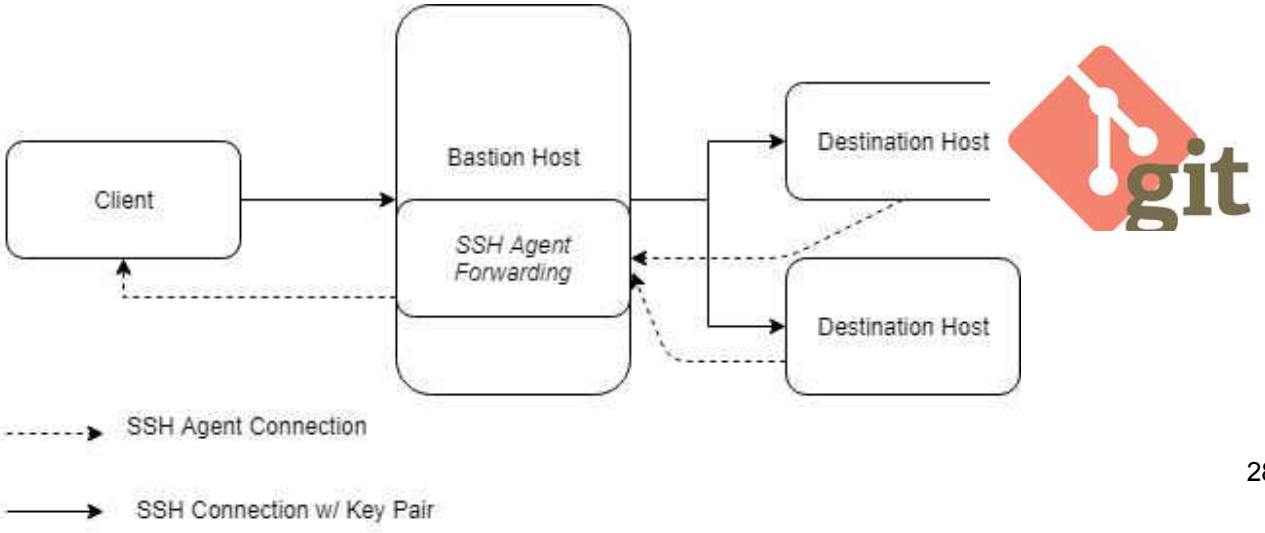
```
sftp usuario1@servidor.dominio.es
```

- Los comandos interactivos son similares a los estudiados en el protocolo **ftp**
  - **put, get, rm, ls, lpwd, ...**

6. IMPLEMENTACIONES

Funcionalidades adicionales

- **Agent Forwarding.** Consiste en el hecho de reenviar las claves **al agente de autenticación**, del cliente que ha iniciado la conexión, de forma que no es necesario guardar las claves de autenticación en ninguna máquina de la red (exceptuando la máquina del usuario).



6. IMPLEMENTACIONES

Funcionalidades adicionales

- Permite el **portforwarding:** envío de conexiones de **TCP / IP** a una máquina remota por un canal cifrado mediante el mapeado de un puerto local del cliente a un puerto remoto del servidor.

```
ssh -L local-puerto: remote-hostname:
remote-puerto username @ hostname
```

- **Compresión de datos**. OpenSSH comprime los datos antes del cifrado lo que mejora los resultados en los enlaces con redes lentas.

6. IMPLEMENTACIONES

Otras implementaciones

- Existen otras implementaciones para windows como son:
  - **FreeSSHd**
  - **Bitvise**

7. BIBLIOGRAFÍA / WEBGRAFÍA

- Velaz. R. “Accés a Sistemes Remots”. https://bit.ly/3jsczSp https://bit.ly/3jsczSp. **Institut Obert de Catalunya (IOC)**
- C. Lonvick. (2006). "RFC4253- The Secure Shell (SSH)". **Internet Engineering Task Force (IETF)**