La Guía Definitiva de SSH y Certificados Digitales: De la Teoría a la Práctica en Windows y Ubuntu

Introducción: Construyendo Confianza en un Mundo Digital

En la infraestructura de TI moderna, la comunicación segura y la verificación de identidad no son lujos, sino necesidades fundamentales. Cada vez que accedemos a un servidor remoto, realizamos una compra en línea o enviamos información sensible, nos enfrentamos a preguntas críticas: ¿Cómo podemos estar seguros de que nos estamos comunicando con el servidor legítimo y no con un impostor? ¿Cómo garantizamos que los datos que enviamos no pueden ser interceptados y leídos por terceros en una red inherentemente insegura como Internet?.[1]

La respuesta a estos desafíos reside en una rama de las matemáticas y la informática conocida como **criptografía asimétrica**, o criptografía de clave pública. Este elegante y robusto concepto es la piedra angular sobre la que se construyen dos de las tecnologías de seguridad más importantes de la actualidad: los Certificados Digitales (la base de HTTPS) y el protocolo Secure Shell (SSH).[2]

Este documento está diseñado para ser un viaje completo, desde los fundamentos teóricos hasta la implementación práctica. Se comenzará por desmitificar los principios de la criptografía de clave pública, para luego aplicar ese conocimiento al mundo de los certificados digitales y, finalmente, dominar la herramienta esencial de cualquier administrador de sistemas: SSH. El objetivo no es solo aprender a ejecutar comandos, sino comprender profundamente *por qué* funcionan, capacitando al lector no solo para *hacer*, sino para *entender* y tomar decisiones informadas sobre la seguridad de sus sistemas.

Parte I: Los Cimientos - Entendiendo la Criptografía de Clave Pública

Para construir una fortaleza digital, primero debemos entender los materiales y principios de su construcción. La criptografía de clave pública es el cimiento sobre el que se erige toda la

seguridad de las comunicaciones modernas.

1.1. El Secreto del Par de Claves: Tu Identidad Digital Única

En el corazón de la criptografía asimétrica se encuentra un par de claves matemáticamente vinculadas: una **clave pública** y una **clave privada**.[4] Estas claves se generan juntas, pero tienen roles completamente opuestos:

- **Clave Pública:** Como su nombre indica, está diseñada para ser compartida libremente. Se puede distribuir a cualquiera sin comprometer la seguridad.
- **Clave Privada:** Es el secreto más guardado. Debe permanecer exclusivamente en posesión de su propietario y nunca, bajo ninguna circunstancia, debe ser compartida.[4]

La magia de este par de claves reside en su relación complementaria: lo que una clave cifra, solo la otra puede descifrarlo, y viceversa. Están diseñadas de tal manera que, computacionalmente, es imposible derivar la clave privada a partir de la clave pública.[4]
Para hacer estos conceptos abstractos más tangibles, se pueden utilizar dos analogías:

1. **Analogía del Buzón con Dos Ranuras (para la Confidencialidad):** Imagine un buzón especial. Tiene una ranura abierta al público (la clave pública) por donde cualquiera puede depositar un mensaje cifrado. Sin embargo, solo una persona posee la llave única (la clave privada) que puede abrir el buzón y leer los mensajes que contiene. Esto ilustra perfectamente cómo se logra la confidencialidad: muchos pueden enviar información segura, pero solo el destinatario previsto puede acceder a ella.[2]
2. **Analogía del Sello de Cera Personal (para la Autenticación):** Piense en un sello de cera con un diseño único, que solo usted posee (la clave privada). Puede usarlo para sellar un documento, creando una marca inconfundible. Cualquiera que conozca el diseño de su sello (la clave pública) puede verificar que el sello en el documento es auténtico y que, por lo tanto, el documento proviene de usted y no ha sido alterado. Sin embargo, no pueden usar ese conocimiento para replicar su sello. Esto demuestra el principio de la **firma digital**, que garantiza la autenticidad y la integridad.[7]

1.2. Dos Caras de la Misma Moneda: Confidencialidad vs. Autenticación

La misma tecnología de par de claves resuelve dos problemas fundamentalmente diferentes simplemente invirtiendo el rol de las claves. Esta dualidad funcional es la razón por la que un único principio criptográfico da lugar a tecnologías tan diversas como SSH y SSL/TLS.

- **Confidencialidad (Cifrado):** El objetivo es mantener un mensaje secreto para todos excepto para el destinatario.
    - **Flujo de trabajo:** Alicia quiere enviar un mensaje secreto a Roberto. Alicia toma la **clave pública de Roberto** y la usa para cifrar el mensaje. Ahora, el mensaje es ilegible para cualquiera que lo intercepte. Cuando Roberto recibe el mensaje, usa

su propia **clave privada** para descifrarlo y leer el contenido original. Solo él puede hacerlo, porque solo él tiene la clave privada correspondiente.[2]

- **Autenticación (Firma Digital):** El objetivo es probar quién envió un mensaje y que este no ha sido modificado.
    - **Flujo de trabajo:** Roberto quiere enviar un documento a Alicia y demostrar que es realmente él quien lo envía. Roberto usa su **clave privada** para crear una "firma digital" del documento. Envía el documento junto con la firma. Alicia, al recibirlo, usa la **clave pública de Roberto** para verificar la firma. Si la verificación es exitosa, Alicia tiene la certeza de dos cosas: que el documento fue firmado por Roberto (autenticidad) y que no ha sido alterado desde que se firmó (integridad).[7]

1.3. El Enfoque Híbrido: Lo Mejor de Dos Mundos

A pesar de su robustez, la criptografía asimétrica tiene un inconveniente: es computacionalmente intensiva y, por lo tanto, relativamente lenta en comparación con su contraparte, la criptografía simétrica (donde se utiliza la misma clave para cifrar y descifrar).[4] Cifrar grandes cantidades de datos con claves asimétricas sería ineficiente.
La solución que adoptan protocolos del mundo real como SSH y TLS/SSL es un **enfoque híbrido**. Este método combina la seguridad de la criptografía asimétrica con la velocidad de la simétrica en un proceso ingenioso:

1. Al inicio de la comunicación, el cliente y el servidor utilizan la **criptografía asimétrica** (lenta pero segura) para autenticarse mutuamente y negociar de forma segura una clave secreta temporal. Esta clave es simétrica y única para esa sesión.
2. Una vez que ambas partes tienen esta clave de sesión compartida, abandonan la criptografía asimétrica.
3. El resto de la comunicación se cifra y descifra utilizando la **criptografía simétrica** (rápida y eficiente) con la clave de sesión que acaban de intercambiar.[5]

Este enfoque no es un detalle técnico menor; es una lección fundamental en ingeniería de sistemas. Demuestra un compromiso pragmático entre la seguridad teórica máxima y el rendimiento práctico, una optimización necesaria para que las comunicaciones seguras sean viables a la escala de Internet.

Parte II: Certificados Digitales - El Pasaporte de Identidad en Internet

Si la criptografía de clave pública es el lenguaje de la confianza, los certificados digitales son los documentos oficiales que formalizan esa confianza en la vasta y anónima extensión de Internet. Son la razón por la que su navegador muestra un candado al conectarse a su banco.

2.1. Anatomía de un Certificado Digital

Un certificado digital, a menudo llamado certificado SSL/TLS, es un documento electrónico que utiliza criptografía para vincular una identidad (como el nombre de dominio de un sitio web, www.ejemplo.com) con una clave pública. Crucialmente, este documento está firmado digitalmente por una entidad de confianza, que actúa como un notario digital.[10]
Un certificado contiene información estandarizada, similar a un pasaporte [12]:

- **Sujeto (Subject):** El nombre del propietario del certificado (por ejemplo, el Nombre Común o CN es www.ejemplo.com).
- **Clave Pública del Sujeto:** La clave pública que se está certificando, perteneciente al sujeto.
- **Emisor (Issuer):** El nombre de la entidad que emitió y firmó el certificado (la Autoridad de Certificación).
- **Periodo de Validez:** Las fechas de inicio y fin durante las cuales el certificado es válido.
- **Firma Digital del Emisor:** La prueba criptográfica de que el emisor ha verificado la información y garantiza su autenticidad. Esta firma se crea con la clave privada del emisor.[14]

2.2. La Cadena de Confianza: ¿Por Qué Confías en tu Banco Online?

El problema fundamental es: ¿cómo sabe su navegador que el certificado presentado por mibanco.com es legítimo y no una falsificación creada por un atacante? No se puede confiar ciegamente en lo que dice el propio servidor.
Aquí es donde entra en juego la **Infraestructura de Clave Pública (PKI)**, un sistema basado en **Autoridades de Certificación (CA)**. Las CAs son organizaciones (como Let's Encrypt, DigiCert o GlobalSign) cuya función es verificar la identidad de las entidades que solicitan un certificado.[7]
El sistema funciona a través de una **cadena de confianza**:

1. **Certificados Raíz (Root Certificates):** Su sistema operativo y su navegador vienen con una lista preinstalada de certificados públicos pertenecientes a CAs raíz de alta confianza. Son el ancla de todo el sistema de confianza.[15]
2. **Verificación en Cadena:** Cuando su navegador se conecta a https://mibanco.com, el servidor le presenta su certificado. El navegador mira quién lo firmó (el emisor).
3. Si el emisor es una de las CAs raíz de confianza, la verificación es exitosa.
4. Si no, el navegador comprueba si el certificado del emisor fue, a su vez, firmado por otra CA (una CA intermedia). Este proceso se repite, subiendo por la cadena de certificados, hasta que se encuentra una CA raíz que está en la lista de confianza del navegador.[13]
5. Si se puede trazar una ruta válida hasta una CA raíz de confianza, el certificado del sitio web se considera válido y la conexión se establece de forma segura. Si no, el navegador

muestra una advertencia de seguridad.

Este modelo de confianza es centralizado y jerárquico: se confía en un sitio porque se confía en la entidad que lo firmó, y se confía en esa entidad porque el fabricante del sistema operativo nos dijo que lo hiciéramos. Este modelo es lo que permite que la confianza escale a miles de millones de dispositivos en la Internet pública.

2.3. Taller Práctico - HTTPS en Ubuntu Server con Nginx

Para comprender verdaderamente el mecanismo de confianza, no hay nada como verlo fallar de forma controlada. En este taller, se configurará un servidor web en Ubuntu con un certificado **autofirmado**, lo que nos permitirá observar la reacción del navegador y entender por qué ocurre.

Paso 1: Instalar Nginx

Primero, se necesita un servidor web. En un terminal de Ubuntu Server, se instalan Nginx y se verifica su estado:

Bash

```
sudo apt update
sudo apt install nginx
sudo systemctl status nginx
```

Si está activo, Nginx estará sirviendo una página por defecto en el puerto 80 (HTTP).

Paso 2: Generar Clave Privada y Certificado Autofirmado con OpenSSL

Ahora, se creará un par de clave/certificado. Como se está firmando el certificado uno mismo (en lugar de que lo haga una CA), se le llama "autofirmado".

Bash

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
-keyout /etc/ssl/private/nginx-selfsigned.key \
-out /etc/ssl/certs/nginx-selfsigned.crt
```

Desglose del comando [17]:

- openssl req -x509: Herramienta para crear un certificado X.509.
- -nodes: "No DES", significa no cifrar la clave privada con una contraseña. Esto es conveniente para que el servidor se reinicie sin intervención manual.
- -days 365: Establece la validez del certificado en un año.
- -newkey rsa:2048: Genera una nueva clave privada RSA de 2048 bits.
- -keyout: Especifica la ruta para guardar la clave privada.
- -out: Especifica la ruta para guardar el certificado.

El comando hará una serie de preguntas (País, Estado, Organización, etc.). La más importante es Common Name (e.g. server FQDN or YOUR name). Aquí se debe introducir la dirección IP del servidor o su nombre de dominio si lo tuviera.[19]

Paso 3: Configurar Nginx para Usar el Certificado (HTTPS)

Se debe indicar a Nginx que use estos nuevos archivos para las conexiones seguras. Se crea un nuevo archivo de configuración para el sitio:

Bash

```
sudo nano /etc/nginx/sites-available/mi_sitio_seguro
```

Y se pega la siguiente configuración, reemplazando ip_del_servidor con la IP real del servidor:

Nginx

```
server {
    listen 443 ssl;
    listen [::]:443 ssl;

    server_name ip_del_servidor;

    ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
    ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;

    root /var/www/html;
    index index.html index.htm;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

Se guarda el archivo, se crea un enlace simbólico para activarlo y se prueba la configuración de Nginx:

Bash

```
sudo ln -s /etc/nginx/sites-available/mi_sitio_seguro /etc/nginx/sites-enabled/
sudo nginx -t
```

Si la sintaxis es correcta, se reinicia Nginx para aplicar los cambios.

Bash

```
sudo systemctl restart nginx
```

Paso 4: Ajustar el Firewall y Probar

Se debe permitir el tráfico en el puerto 443 (HTTPS) a través del firewall ufw:

Bash

```
sudo ufw allow 'Nginx Full'
sudo ufw delete allow 'Nginx HTTP' # Opcional, si se quiere forzar HTTPS
```

Ahora, desde un navegador en la máquina local, se accede al servidor usando https://ip_del_servidor.

El Momento de la Verdad

El navegador mostrará una advertencia prominente: "Su conexión no es privada" o "Advertencia: Riesgo potencial de seguridad a continuación".[17] Este es el punto de aprendizaje clave. La advertencia no significa que la conexión no esté cifrada; de hecho, lo está. Lo que el navegador está diciendo es: "He recibido un certificado que cifra la comunicación, pero el emisor de este certificado soy yo mismo (el servidor). No puedo verificar su identidad a través de ninguna de las Autoridades de Certificación en las que confío".
Este ejercicio demuestra que el cifrado (confidencialidad) sin una autenticación robusta y verificable tiene un valor limitado. Se ha establecido un canal seguro, pero no se puede estar

seguro de con quién se está hablando. Para eliminar esta advertencia, se necesitaría un certificado firmado por una CA de confianza, como Let's Encrypt.[18]

Parte III: Secure Shell (SSH) - La Llave Maestra para tus Servidores

Si los certificados digitales son el pasaporte para el público general en la web, SSH es la llave maestra de alta seguridad para los administradores de sistemas. Es la herramienta fundamental para gestionar servidores remotos de forma segura.

3.1. Acceso Remoto Seguro: Más Allá del Telnet

SSH, o Secure Shell, es un protocolo de red que proporciona a los administradores una forma segura de acceder a un ordenador a través de una red no segura.[1] Funciona en una arquitectura cliente-servidor, donde un cliente SSH en la máquina local se conecta a un servidor SSH (demonio) que se ejecuta en la máquina remota.[21] SSH proporciona un canal de comunicación cifrado y autenticado, asegurando la confidencialidad e integridad de todos los datos transmitidos, incluyendo comandos, salida de texto y transferencias de archivos.

3.2. El Duelo: Contraseñas vs. Claves SSH

SSH admite dos métodos principales de autenticación: contraseñas y claves de clave pública. Aunque las contraseñas son familiares, son significativamente menos seguras.

- **Autenticación por Contraseña:** Se basa en "algo que sabes". Este método es vulnerable a:
  - **Ataques de fuerza bruta:** Bots que intentan miles de contraseñas por segundo.
  - **Contraseñas débiles:** Los humanos tienden a elegir contraseñas fáciles de adivinar o reutilizar.
  - **Phishing y exposición:** Las contraseñas pueden ser robadas o filtradas en brechas de seguridad de otros servicios.[21]
- **Autenticación por Clave SSH:** Se basa en "algo que tienes" (la clave privada) y utiliza la misma criptografía asimétrica que se ha discutido.[22] El proceso de autenticación es un desafío-respuesta criptográfico: el servidor usa la clave pública del cliente para emitir un desafío que solo la clave privada correspondiente puede resolver correctamente, probando así la identidad del cliente sin enviar nunca la clave privada a través de la red.[23]

Las ventajas de la autenticación por clave son abrumadoras:

- **Seguridad Superior:** Un par de claves SSH (por ejemplo, de 2048 o 4096 bits) es

exponencialmente más complejo que cualquier contraseña que un humano pueda memorizar, lo que lo hace inmune a los ataques de fuerza bruta.[6]

- **Comodidad y Automatización:** Permite inicios de sesión sin intervención del usuario, lo cual es esencial para scripts, copias de seguridad automatizadas y sistemas de integración continua.[21]
- **Gestión de Acceso Granular:** El acceso se controla añadiendo o eliminando claves públicas del servidor. Esto permite revocar el acceso de un usuario específico de forma instantánea sin afectar a otros y sin la necesidad de cambiar contraseñas compartidas, lo que mejora la auditoría y la trazabilidad.[21]

A diferencia del modelo de confianza centralizado de los certificados SSL/TLS, SSH utiliza un modelo de confianza descentralizado y explícito. La confianza se establece manualmente: "confío en esta clave pública porque yo, el administrador, la he colocado en este servidor". Este modelo es perfecto para gestionar un número conocido de sistemas y usuarios, donde un administrador puede establecer esa confianza inicial.

3.3. El Ecosistema de Ficheros de SSH: Entendiendo el Directorio ~/.ssh

Tanto en el cliente como en el servidor, el directorio ~/.ssh (ubicado en el directorio home del usuario) es el centro de operaciones de SSH. Comprender los archivos que contiene es fundamental para dominar su funcionamiento.

Nombre del Fichero	Ubicación Principal	Contenido	Propósito y Nivel de Sensibilidad	Referencias
id_rsa, id_ed25519	**Cliente**	Tu clave SSH **privada**.	**EXTREMADAMENTE SENSIBLE.** Es tu identidad digital. Protege este archivo como una contraseña maestra. Sus permisos deben ser 600 (solo lectura/escritura para el propietario). SSH ignorará una clave privada con permisos demasiado abiertos.	[6]

id_rsa.pub, id_ed25519.pub	**Cliente** (y se copia al servidor)	Tu clave SSH **pública**.	**PÚBLICO.** Diseñado para ser compartido. Es la "cerradura" que instalas en los servidores a los que quieres acceder. Contiene la clave, el tipo de clave y un comentario opcional (generalmente usuario@host).	[6]
authorized_keys	**Servidor**	Una lista de claves **públicas** autorizadas (una por línea).	**SENSIBLE.** Este archivo es la lista de control de acceso. Cualquier persona que posea la clave privada correspondiente a una de las claves públicas listadas aquí puede iniciar sesión como este usuario en el servidor.	[23]
known_hosts	**Cliente**	Una lista de las claves **públicas de los servidores** a los que te has conectado previamente.	**SENSIBLE.** Este es el mecanismo de defensa del cliente contra ataques de Man-in-the-Middle (MITM). Al conectarse a un servidor, el cliente verifica que la clave pública del servidor coincida con la que está guardada en este archivo. Si no	[27]

			coincide, SSH mostrará una advertencia severa.	

Parte IV: Taller Práctico Integral de SSH

En esta sección, se aplicará toda la teoría para configurar un flujo de trabajo de SSH seguro y práctico, cubriendo la generación de claves en Windows y WSL, la concesión de acceso a un servidor Ubuntu y el fortalecimiento de la seguridad del servidor.

4.1. Generación de Claves en Diferentes Entornos

El primer paso es crear un par de claves SSH en la máquina local.

En Windows (usando PowerShell)

Las versiones modernas de Windows 10 y 11 incluyen un cliente OpenSSH de forma nativa, accesible a través de PowerShell o el Símbolo del sistema.[29]

1. Abrir PowerShell.
2. Ejecutar el comando ssh-keygen. Se recomienda usar el algoritmo Ed25519 por su seguridad y rendimiento, aunque RSA sigue siendo una alternativa robusta.[31]
   PowerShell
   ssh-keygen.exe -t ed25519

   O para RSA:
   PowerShell
   ssh-keygen.exe -t rsa -b 4096

3. El programa hará varias preguntas:
   - Enter file in which to save the key...: Se puede presionar Enter para aceptar la ubicación predeterminada, que es C:\Users\<TuUsuario>\.ssh\id_ed25519.[32]
   - Enter passphrase (empty for no passphrase):: Se puede introducir una frase de contraseña. Esto cifra el archivo de la clave privada en el disco, añadiendo una capa de seguridad de "algo que sabes" a "algo que tienes". Si alguien roba el archivo de la clave privada, todavía necesitaría la frase de contraseña para usarla. Para scripts automatizados, se puede dejar en blanco.[22]

En WSL (Ubuntu)

El proceso dentro del Subsistema de Windows para Linux (WSL) es idéntico al de un sistema Linux nativo.

1. Abrir una terminal de WSL (por ejemplo, Ubuntu).
2. Ejecutar el mismo comando ssh-keygen:
   Bash
   ssh-keygen -t ed25519

3. Las claves se guardarán en el sistema de archivos de WSL, en la ruta ~/.ssh/ (que es /home/<tu_usuario_wsl>/.ssh/).[33] Desde Windows, esta ruta es accesible a través de \\wsl$\Ubuntu\home\<tu_usuario_wsl>\.ssh\.[30]

4.2. Estableciendo la Confianza: Concediendo Acceso al Servidor

Una vez generadas las claves, se debe copiar la clave **pública** al servidor Ubuntu para autorizar el acceso.

Método 1 (Recomendado, desde WSL): Usando ssh-copy-id

La herramienta ssh-copy-id es la forma más sencilla y segura de hacerlo, ya que automatiza el proceso de añadir la clave al archivo authorized_keys del servidor y se asegura de que los permisos del directorio y del archivo sean los correctos.[35]
Desde la terminal de WSL, se ejecuta:

Bash

ssh-copy-id -i ~/.ssh/id_ed25519.pub usuario@ip_del_servidor

Se solicitará la contraseña del usuario en el servidor remoto *por última vez*. Después de esto, la autenticación por clave estará configurada.[37]

Método 2 (Manual, desde Windows PowerShell): Replicando ssh-copy-id

Dado que ssh-copy-id no está disponible por defecto en PowerShell, se puede realizar el proceso manualmente. Esto también sirve para entender qué hace el script por debajo.
Desde PowerShell, se ejecuta el siguiente comando en una sola línea:

PowerShell

```
cat $env:USERPROFILE\.ssh\id_ed25519.pub | ssh usuario@ip_del_servidor "mkdir -p ~/.ssh; chmod 700 ~/.ssh; cat >> ~/.ssh/authorized_keys; chmod 600 ~/.ssh/authorized_keys"
```

Desglose del comando [37]:

- cat...: Lee el contenido del archivo de la clave pública.
- |: El operador "pipe" redirige la salida del comando cat como entrada para el comando ssh.
- ssh... "...": Se conecta al servidor remoto y ejecuta la serie de comandos entre comillas.
- mkdir -p ~/.ssh; chmod 700 ~/.ssh: Crea el directorio .ssh si no existe y establece sus permisos a 700 (solo el propietario puede leer, escribir y ejecutar).
- cat >> ~/.ssh/authorized_keys: Toma la clave pública que viene por la entrada estándar y la añade (>>) al final del archivo authorized_keys.
- chmod 600 ~/.ssh/authorized_keys: Establece los permisos del archivo a 600 (solo el propietario puede leer y escribir).

Estos permisos son cruciales. El servidor SSH es muy estricto y rechazará la autenticación por clave si los permisos en el directorio .ssh o en el archivo authorized_keys son demasiado abiertos.[40]

4.3. La Conexión Segura y el Fichero known_hosts

Ahora, se puede intentar la conexión sin contraseña desde Windows PowerShell o WSL:

Bash

```
ssh usuario@ip_del_servidor
```

La primera vez que se conecte a este servidor, se verá un mensaje como este:

```
The authenticity of host '123.45.67.89 (123.45.67.89)' can't be established.
ED25519 key fingerprint is SHA256:xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

Este es un paso de seguridad vital. El cliente SSH está presentando la clave pública del servidor y preguntando si se confía en ella. Al escribir yes, el cliente guarda la clave pública

del servidor en el archivo ~/.ssh/known_hosts en la máquina local. En futuras conexiones, el cliente comparará la clave del servidor con la guardada. Si no coinciden, se emitirá una advertencia grave, ya que podría indicar un ataque de suplantación de identidad (Man-in-the-Middle).[27]
Una vez aceptada la clave, si no se configuró una frase de contraseña para la clave privada, el inicio de sesión será instantáneo.

4.4. Fortaleciendo el Bastión: Asegurando el Servidor SSH

La configuración por defecto de muchos sistemas permite la autenticación tanto por clave como por contraseña para mantener la compatibilidad. Sin embargo, una vez que la autenticación por clave funciona correctamente, deshabilitar la autenticación por contraseña es el paso más importante para asegurar un servidor SSH expuesto a Internet.[42]

1. **Conectarse al servidor Ubuntu** usando la clave SSH recién configurada.
2. **Editar el archivo de configuración del demonio SSH.** Se necesita usar sudo ya que es un archivo de sistema.
   Bash
   sudo nano /etc/ssh/sshd_config

3. **Modificar las directivas clave.** Dentro del archivo, se deben buscar y modificar las siguientes líneas:
   - Cambiar PasswordAuthentication yes a PasswordAuthentication no. Si la línea está comentada (empieza con #), se debe quitar el #.[43]
   - **Mejor Práctica Adicional:** Asegurarse de que PermitRootLogin esté configurado como prohibit-password o no. Esto impide que el usuario root inicie sesión directamente a través de SSH, forzando a los administradores a iniciar sesión con un usuario normal y luego elevar privilegios con sudo, lo que mejora la auditoría.[41]
4. **Guardar el archivo y reiniciar el servicio SSH** para que los cambios surtan efecto.
   Bash
   sudo systemctl restart sshd

   O en versiones más antiguas:
   Bash
   sudo service ssh restart

Para verificar que el cambio ha funcionado, se debe cerrar la sesión actual, abrir una **nueva terminal** e intentar conectarse de nuevo. La conexión con la clave SSH debería funcionar. Si se intenta conectar con un usuario que no tiene una clave autorizada, la conexión será rechazada inmediatamente, en lugar de pedir una contraseña. Se ha eliminado con éxito el vector de ataque de fuerza bruta por contraseña.

Conclusión: Unificando el Conocimiento y Próximos Pasos

A lo largo de este recorrido, se ha desentrañado el funcionamiento interno de dos de las tecnologías de seguridad más cruciales de la infraestructura digital: los Certificados Digitales y el protocolo SSH. Aunque a primera vista pueden parecer dispares, ahora es evidente que ambos se basan en el mismo principio fundamental: la **criptografía de clave pública**. La diferencia clave radica en su modelo de confianza y su aplicación.

- Los **Certificados Digitales** (y el sistema PKI) implementan un modelo de confianza **centralizado y jerárquico**, diseñado para escalar a la inmensidad de la Internet pública. La confianza se delega en un conjunto de Autoridades de Certificación raíz, permitiendo que un navegador verifique la identidad de un sitio web que nunca ha visitado antes. Su principal objetivo es la **verificación de la identidad de un servidor para el público general**.
- **SSH**, por otro lado, utiliza un modelo de confianza **descentralizado y explícito**. La confianza no se delega, sino que se establece directamente por el administrador al colocar una clave pública en un servidor. Este modelo es ideal para la **autenticación de usuarios y sistemas conocidos en un entorno controlado**, como la administración de una flota de servidores.

La criptografía asimétrica es la tecnología habilitadora para ambos, pero se aplica de manera distinta para resolver problemas diferentes: el acceso seguro para administradores (SSH) frente a la verificación de identidad para todos (SSL/TLS).

Con los conocimientos y las habilidades prácticas adquiridas, el siguiente paso es explorar temas más avanzados para seguir mejorando la seguridad y la eficiencia:

- **Agentes SSH (ssh-agent):** Una herramienta que carga las claves privadas en memoria y gestiona su uso, evitando la necesidad de teclear la frase de contraseña de la clave en cada conexión dentro de una misma sesión de trabajo.[22]
- **El Fichero de Configuración del Cliente (~/.ssh/config):** Un archivo potente en la máquina cliente que permite crear alias, especificar claves por defecto para diferentes hosts, configurar puertos y simplificar drásticamente la conexión a múltiples servidores.[46]
- **Certificados de CA Gratuitos con Let's Encrypt:** Como siguiente paso al taller de certificados autofirmados, aprender a usar herramientas como Certbot para obtener e instalar automáticamente certificados SSL/TLS gratuitos y universalmente confiables de la autoridad de certificación Let's Encrypt, eliminando las advertencias del navegador y llevando la seguridad del servidor web a un nivel de producción.[18]

Dominar estos conceptos no es solo una habilidad técnica; es una mentalidad. Significa comprender que la seguridad no es un producto que se instala, sino un proceso de configuración deliberada, fortalecimiento y vigilancia constante.

Obras citadas

1. La guía del desarrollador para empezar a usar SSH (Conéctate a tu servidor de forma segura) - Kinsta, fecha de acceso: octubre 14, 2025, https://kinsta.com/es/blog/guia-desarrollador-usar-ssh/
2. Criptografía asimétrica | Cloud Key Management Service, fecha de acceso: octubre 14, 2025, https://cloud.google.com/kms/docs/asymmetric-encryption?hl=es-419
3. ¿Qué es SSH? | Protocolo Secure Shell (SSH) - Cloudflare, fecha de acceso: octubre 14, 2025, https://www.cloudflare.com/es-es/learning/access-management/what-is-ssh/
4. Criptografía de clave asimétrica - CERES - FNMT, fecha de acceso: octubre 14, 2025, https://www.cert.fnmt.es/curso-de-criptografia/criptografia-de-clave-asimetrica
5. ¿Qué es el cifrado asimétrico? - IBM, fecha de acceso: octubre 14, 2025, https://www.ibm.com/es-es/think/topics/asymmetric-encryption
6. Qué es una clave SSH: autenticación segura sin contraseña - Raiola Networks, fecha de acceso: octubre 14, 2025, https://raiolanetworks.com/ayuda/clave-ssh/
7. Criptografía asimétrica - Wikipedia, la enciclopedia libre, fecha de acceso: octubre 14, 2025, https://es.wikipedia.org/wiki/Criptograf%C3%ADa_asim%C3%A9trica
8. Sistemas de claves públicas y privadas I. Criptografía de clave pública - Dialnet, fecha de acceso: octubre 14, 2025, https://dialnet.unirioja.es/descarga/articulo/8832538.pdf
9. ¿Qué son los certificados digitales? | Fortinet, fecha de acceso: octubre 14, 2025, https://www.fortinet.com/lat/resources/cyberglossary/digital-certificates
10. www.lenovo.com, fecha de acceso: octubre 14, 2025, https://www.lenovo.com/es/es/glossary/digital-certificate/#:~:text=Un%20certificado%20digital%20es%20un,de%20confianza%20entre%20las%20partes.
11. Qué es el certificado digital y para qué sirve | Wolters Kluwer, fecha de acceso: octubre 14, 2025, https://www.wolterskluwer.com/es-es/expert-insights/que-es-el-certificado-digital-para-que-sirve
12. Certificados digitales - IBM, fecha de acceso: octubre 14, 2025, https://www.ibm.com/docs/es/integration-bus/10.0.0?topic=overview-digital-certificates
13. ¿Qué es un certificado digital? ¿Cómo funciona? | Lenovo España, fecha de acceso: octubre 14, 2025, https://www.lenovo.com/es/es/glossary/digital-certificate/
14. Claves privadas, claves públicas y certificados digitales - IBM, fecha de acceso: octubre 14, 2025, https://www.ibm.com/docs/es/sia?topic=osdc-private-keys-public-keys-digital-certificates-7
15. Cadena de confianza - Wikipedia, la enciclopedia libre, fecha de acceso: octubre 14, 2025, https://es.wikipedia.org/wiki/Cadena_de_confianza

16. ¿Cuál es la diferencia entre claves PGP, claves SSH y claves SSL? - Quora, fecha de acceso: octubre 14, 2025, https://es.quora.com/Cu%C3%A1l-es-la-diferencia-entre-claves-PGP-claves-SSH-y-claves-SSL
17. Cómo crear un certificado SSL autofirmado para Apache en Ubuntu 18.04 - DigitalOcean, fecha de acceso: octubre 14, 2025, https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-apache-in-ubuntu-18-04-es
18. Cómo crear un certificado SSL autofirmado para Apache en Ubuntu 20.04 - DigitalOcean, fecha de acceso: octubre 14, 2025, https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-apache-in-ubuntu-20-04-es
19. Usando OpenSSL para generar y formatear certificados - IBM, fecha de acceso: octubre 14, 2025, https://www.ibm.com/docs/es/api-connect/10.0.x_cd?topic=profile-using-openssl-generate-format-certificates
20. SSH: Guía básica para aprender a utilizar este protocolo - Raiola Networks, fecha de acceso: octubre 14, 2025, https://raiolanetworks.com/blog/ssh/
21. SSH: qué es y cómo funciona este protocolo | Blog de Arsys, fecha de acceso: octubre 14, 2025, https://www.arsys.es/blog/ssh
22. Llaves SSH vs contraseña : r/linuxquestions - Reddit, fecha de acceso: octubre 14, 2025, https://www.reddit.com/r/linuxquestions/comments/ehapud/ssh_keys_vs_password/?tl=es-419
23. Cómo utilizar la autenticación de clave pública SSH | Linode Docs, fecha de acceso: octubre 14, 2025, https://www.linode.com/docs/guides/use-public-key-authentication-with-ssh/?lang=es
24. Explicación de las claves SSH: Guía para un acceso remoto rápido y seguro - DataCamp, fecha de acceso: octubre 14, 2025, https://www.datacamp.com/es/tutorial/ssh-keys
25. What significance does the user/host at the end of an SSH public key file hold?, fecha de acceso: octubre 14, 2025, https://serverfault.com/questions/743548/what-significance-does-the-user-host-at-the-end-of-an-ssh-public-key-file-hold
26. Do I need to name my public RSA key to "authorized_keys" in ~/.ssh folder? - Reddit, fecha de acceso: octubre 14, 2025, https://www.reddit.com/r/linuxquestions/comments/uipm15/do_i_need_to_name_my_public_rsa_key_to_authorized/
27. What is the difference between authorized_keys and known_hosts file for SSH?, fecha de acceso: octubre 14, 2025, https://security.stackexchange.com/questions/20706/what-is-the-difference-between-authorized-keys-and-known-hosts-file-for-ssh
28. Add public key to known_hosts file - linux - Stack Overflow, fecha de acceso: octubre 14, 2025,

https://stackoverflow.com/questions/34906302/add-public-key-to-known-hosts-file
29. Conectar PowerShell mediante SSH - IONOS, fecha de acceso: octubre 14, 2025, https://www.ionos.com/es-us/digitalguide/servidores/configuracion/powershell-ssh/
30. Generate SSH Keys on Windows 10 - Ubuntu, fecha de acceso: octubre 14, 2025, https://ubuntu.com/tutorials/ssh-keygen-on-windows
31. How to Set Up SSH Keys on Ubuntu: A Comprehensive Guide - DigitalOcean, fecha de acceso: octubre 14, 2025, https://www.digitalocean.com/community/tutorials/how-to-set-up-ssh-keys-on-ubuntu-22-04
32. Configura clave SSH en GitHub para Windows 11 con Power-Shell - Medium, fecha de acceso: octubre 14, 2025, https://medium.com/@aitorevi/configura-clave-ssh-en-github-para-windows-11-con-power-shell-afb51f818328
33. Generate SSH keys en Ubuntu: crear y configurar una clave SSH - IONOS, fecha de acceso: octubre 14, 2025, https://www.ionos.com/es-us/digitalguide/servidores/configuracion/generar-claves-ssh-en-ubuntu/
34. Generate SSH RSA Key Pairs on Windows with WSL | Blog, fecha de acceso: octubre 14, 2025, https://ardalis.com/generate-ssh-rsa-keys-windows-wsl/
35. SSH Copy ID for Copying SSH Keys to Servers, fecha de acceso: octubre 14, 2025, https://www.ssh.com/academy/ssh/copy-id
36. What, exactly, does ssh-copy-id do? - Stack Overflow, fecha de acceso: octubre 14, 2025, https://stackoverflow.com/questions/22700818/what-exactly-does-ssh-copy-id-do
37. Cómo configurar las llaves SSH en Ubuntu 18.04 - DigitalOcean, fecha de acceso: octubre 14, 2025, https://www.digitalocean.com/community/tutorials/como-configurar-las-llaves-ssh-en-ubuntu-18-04-es
38. Claves SSH entre Windows 10 y Linux : r/linuxadmin - Reddit, fecha de acceso: octubre 14, 2025, https://www.reddit.com/r/linuxadmin/comments/1j0ziba/ssh_keys_between_windows_10_and_linux/?tl=es-419
39. Is there an equivalent to ssh-copy-id for Windows? - Server Fault, fecha de acceso: octubre 14, 2025, https://serverfault.com/questions/224810/is-there-an-equivalent-to-ssh-copy-id-for-windows
40. can I share my SSH keys between WSL and Windows? - Super User, fecha de acceso: octubre 14, 2025, https://superuser.com/questions/1183176/can-i-share-my-ssh-keys-between-wsl-and-windows
41. Guía para configurar claves SSH en un servidor Linux [Tutorial] - Stackscale, fecha de acceso: octubre 14, 2025,

https://www.stackscale.com/es/blog/configurar-llaves-ssh-servidor-linux/
42. 12.3. Uso de pares de claves en lugar de contraseñas para la autenticación SSH | Configuración de los ajustes básicos del sistema | Red Hat Enterprise Linux, fecha de acceso: octubre 14, 2025, https://docs.redhat.com/es/documentation/red_hat_enterprise_linux/8/html/configuring_basic_system_settings/using-key-pairs-instead-of-passwords-for-ssh-authentication_using-secure-communications-between-two-systems-with-openssh
43. Proteger su servidor SSH - IBM, fecha de acceso: octubre 14, 2025, https://www.ibm.com/docs/es/aspera-fasp-proxy/1.4.x?topic=appendices-securing-your-ssh-server
44. Cómo configurar SSH sin contraseña en Linux - Hostinger, fecha de acceso: octubre 14, 2025, https://www.hostinger.com/es/tutoriales/configurar-ssh-sin-contrasena-linux
45. Cómo configurar claves SSH: gestión eficaz de claves para conexiones remotas seguras, fecha de acceso: octubre 14, 2025, https://www.hostinger.com/es/tutoriales/como-configurar-claves-ssh
46. Does ssh key need to be named id_rsa? - Ask Ubuntu, fecha de acceso: octubre 14, 2025, https://askubuntu.com/questions/30788/does-ssh-key-need-to-be-named-id-rsa
47. Cómo crear y utilizar claves de autenticación para las conexiones SSH a los servidores de OVHcloud, fecha de acceso: octubre 14, 2025, https://help.ovhcloud.com/csm/es-dedicated-servers-creating-ssh-keys?id=kb_article_view&sysparm_article=KB0043381
48. Ubuntu 22.04 + Nginx + SSL (acme.sh) + Cloudflare DNS Setup + Flask + tumx - GitHub Gist, fecha de acceso: octubre 14, 2025, https://gist.github.com/dsdanielpark/07f672b570203e9833a825d07163a69e