Comprensión de Secure Shell Packet Exchange

Contenido

Introducción

Este documento describe el intercambio de nivel de paquete durante la negociación de Secure Shell (SSH).

Prerequisites

Requirements

Cisco recomienda que conozca los conceptos básicos de seguridad:

- Autenticación
- Confidencialidad
- Integridad
- Métodos de intercambio de claves

Componentes Utilizados

Este documento no se limita a una versión de hardware específica.

La información que contiene este documento se creó a partir de los dispositivos en un ambiente de laboratorio específico. Todos los dispositivos que se utilizan en este documento se pusieron en funcionamiento con una configuración verificada (predeterminada).

Protocolo SSH

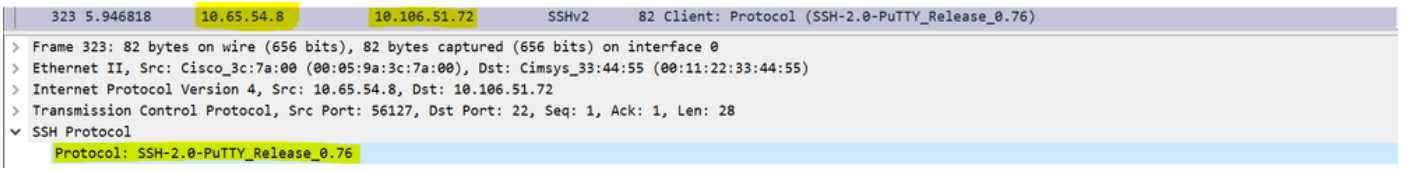
El protocolo SSH es un método para el inicio de sesión remoto seguro de un ordenador a otro. Las aplicaciones SSH se basan en una arquitectura cliente-servidor, que conecta una instancia de cliente SSH con un servidor SSH.

Intercambio SSH

1. El primer paso de SSH se llama Identification String Exchange.

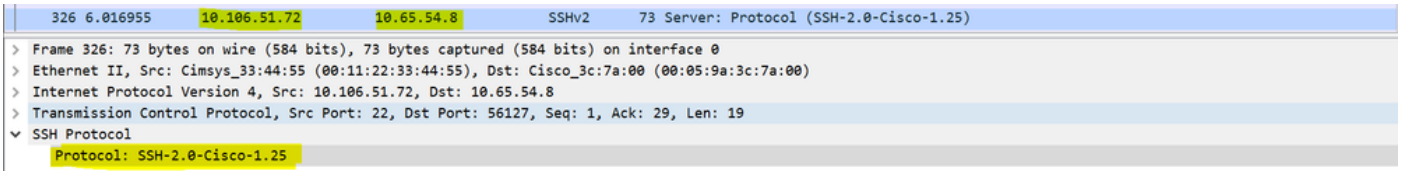
a. El cliente construye un paquete y lo envía al servidor que contiene:

- Versión del protocolo SSH
- Versión del software



La versión del protocolo de cliente es SSH2.0 y la versión de software es Putty_0.76.

b. El servidor responde con su propio Identification String Exchange, incluyendo su versión de protocolo SSH y versión de software.

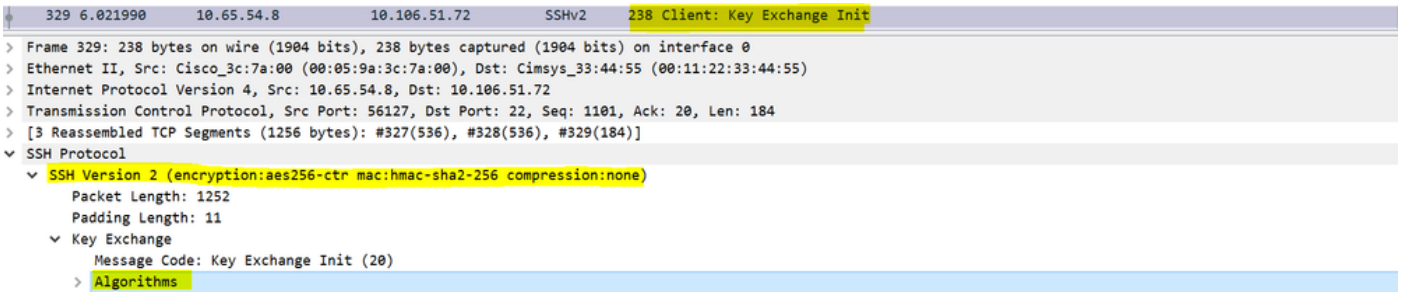


La versión del protocolo del servidor es SSH2.0 y la versión del software es Cisco1.25

2. El siguiente paso es **Algorithm Negotiation**. En este paso, tanto el cliente como el servidor negocian estos algoritmos:

- Intercambio de claves
- Cifrado
- HMAC (código de autenticación de mensajes basado en hash)
- Compresión

1. El cliente envía un mensaje Key Exchange Init al servidor, especificando los algoritmos que soporta. Los algoritmos se enumeran por orden de preferencia.



Key Exchange Init

```
Algorithms
    Cookie: 47a96215afc92003180b60342970a105
    kex_algorithms length: 315
    kex_algorithms string [truncated]: curve448-sha512,curve25519-sha256,curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,dif
    server_host_key_algorithms length: 123
    server_host_key_algorithms string: rsa-sha2-512,rsa-sha2-256,ssh-rsa,ssh-ed448,ssh-ed25519,ecdsa-sha2-nistp256,ecdsa-sha2-nistp384,ecdsa-sha2-nistp521,ssh-dss
    encryption_algorithms_client_to_server length: 189
    encryption_algorithms_client_to_server string: aes256-ctr,aes256-cbc,rijndael-cbc@lysator.liu.se,aes192-ctr,aes192-cbc,aes128-ctr,aes128-cbc,chacha20-poly1305
    encryption_algorithms_server_to_client length: 189
    encryption_algorithms_server_to_client string: aes256-ctr,aes256-cbc,rijndael-cbc@lysator.liu.se,aes192-ctr,aes192-cbc,aes128-ctr,aes128-cbc,chacha20-poly1305
    mac_algorithms_client_to_server length: 155
    mac_algorithms_client_to_server string: hmac-sha2-256,hmac-sha1,hmac-sha1-96,hmac-md5,hmac-sha2-256-etm@openssh.com,hmac-sha1-etm@openssh.com,hmac-sha1-96-etm
    mac_algorithms_server_to_client length: 155
    mac_algorithms_server_to_client string: hmac-sha2-256,hmac-sha1,hmac-sha1-96,hmac-md5,hmac-sha2-256-etm@openssh.com,hmac-sha1-etm@openssh.com,hmac-sha1-96-etm
    compression_algorithms_client_to_server length: 26
    compression_algorithms_client_to_server string: none,zlib,zlib@openssh.com
    compression_algorithms_server_to_client length: 26
    compression_algorithms_server_to_client string: none,zlib,zlib@openssh.com
```

Algoritmos admitidos por el cliente

b. El servidor responde con su propio mensaje Key Exchange Init, enumerando los algoritmos que soporta.

c. Dado que estos mensajes se intercambian simultáneamente, ambas partes comparan sus listas de algoritmos. Si hay una coincidencia en los algoritmos soportados por ambos lados, proceden al siguiente paso. Si no hay una coincidencia exacta, el servidor selecciona el primer algoritmo de la lista del cliente que también admite.

d. Si el cliente y el servidor no pueden acordar un algoritmo común, el intercambio de claves falla.

```
334 6.093250    10.106.51.72    10.65.54.8    SSHv2    366 Server: Key Exchange Init
Frame 334: 366 bytes on wire (2928 bits), 366 bytes captured (2928 bits) on interface 0
Ethernet II, Src: Cimsys_33:44:55 (00:11:22:33:44:55), Dst: Cisco_3c:7a:00 (00:05:9a:3c:7a:00)
Internet Protocol Version 4, Src: 10.106.51.72, Dst: 10.65.54.8
Transmission Control Protocol, Src Port: 22, Dst Port: 56127, Seq: 20, Ack: 1285, Len: 312
SSH Protocol
  SSH Version 2 (encryption:aes256-ctr mac:hmac-sha2-256 compression:none)
      Packet Length: 308
      Padding Length: 4
    Key Exchange
        Message Code: Key Exchange Init (20)
      Algorithms
```

Init de intercambio de claves del servidor

3. Después de esto, ambos lados entran en la **Key Exchange**fase para generar el secreto compartido mediante el intercambio de claves DH y autenticar el servidor:

a. El cliente genera un par de llavesPublic and Privatey envía la clave pública DH en el paquete de inicialización de intercambio de grupo DH. Este par de claves se utiliza para calcular claves secretas.

```
337 6.201114    10.65.54.8    10.106.51.72    SSHv2    326 Client: Diffie-Hellman Group Exchange Init
Frame 337: 326 bytes on wire (2608 bits), 326 bytes captured (2608 bits) on interface 0
Ethernet II, Src: Cisco_3c:7a:00 (00:05:9a:3c:7a:00), Dst: Cimsys_33:44:55 (00:11:22:33:44:55)
Internet Protocol Version 4, Src: 10.65.54.8, Dst: 10.106.51.72
Transmission Control Protocol, Src Port: 56127, Dst Port: 22, Seq: 1309, Ack: 612, Len: 272
SSH Protocol
  SSH Version 2 (encryption:aes256-ctr mac:hmac-sha2-256 compression:none)
      Packet Length: 268
      Padding Length: 6
    Key Exchange
        Message Code: Diffie-Hellman Group Exchange Init (32)
        Multi Precision Integer Length: 256
        DH client e: 1405ab00ff368031363467ad6653967d5a64eac4734e5dc6…
      Padding String: 5c81f2cffc95
```

Client DH Public Key & Diffie-Hellman Group Exchange Init

b. El servidor genera su propioPublic and Privatepar de claves. Utiliza la clave pública del cliente y su propio par de claves para calcular el secreto compartido.

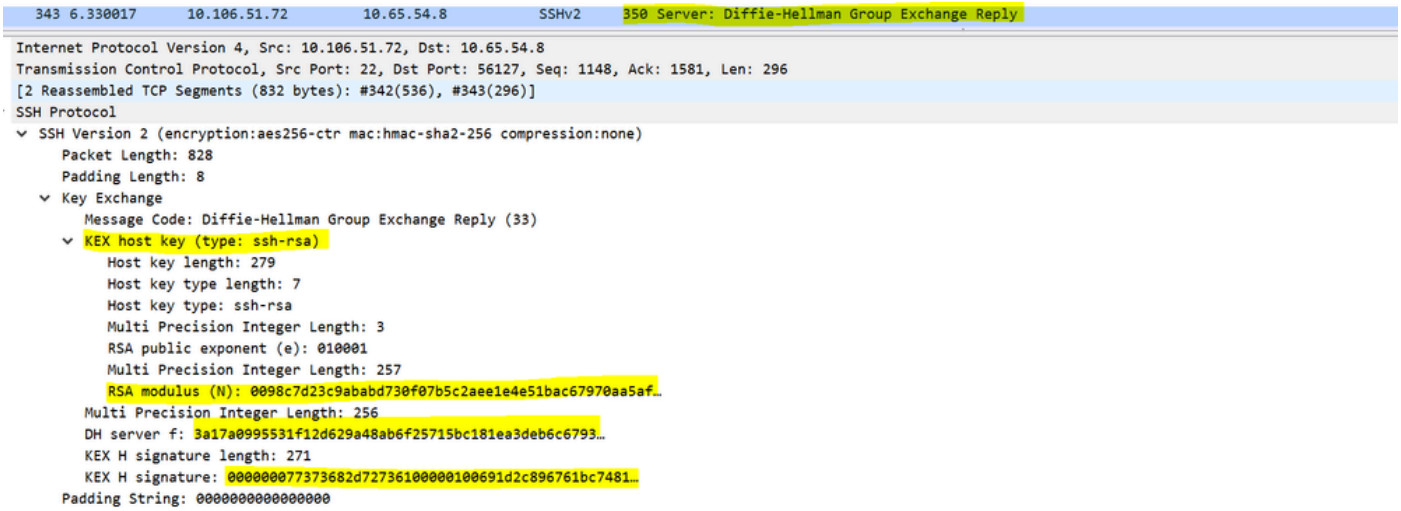
c. El servidor también calcula un hash de Exchange con estas entradas:

- Cadena de identificación de clientes
- Cadena de identificación del servidor
- Carga útil del cliente KEXINIT
- Carga útil del servidor KEXINIT
- Servidores Clave pública desde claves de host (par de claves RSA)
- Clave pública DH de clientes
- Servidores DH Clave pública
- Clave secreta compartida

d. Después de calcular el hash, el servidor lo firma con su clave privada RSA.

e. El servidor crea un mensaje DH_Exchange_Reply que incluye:

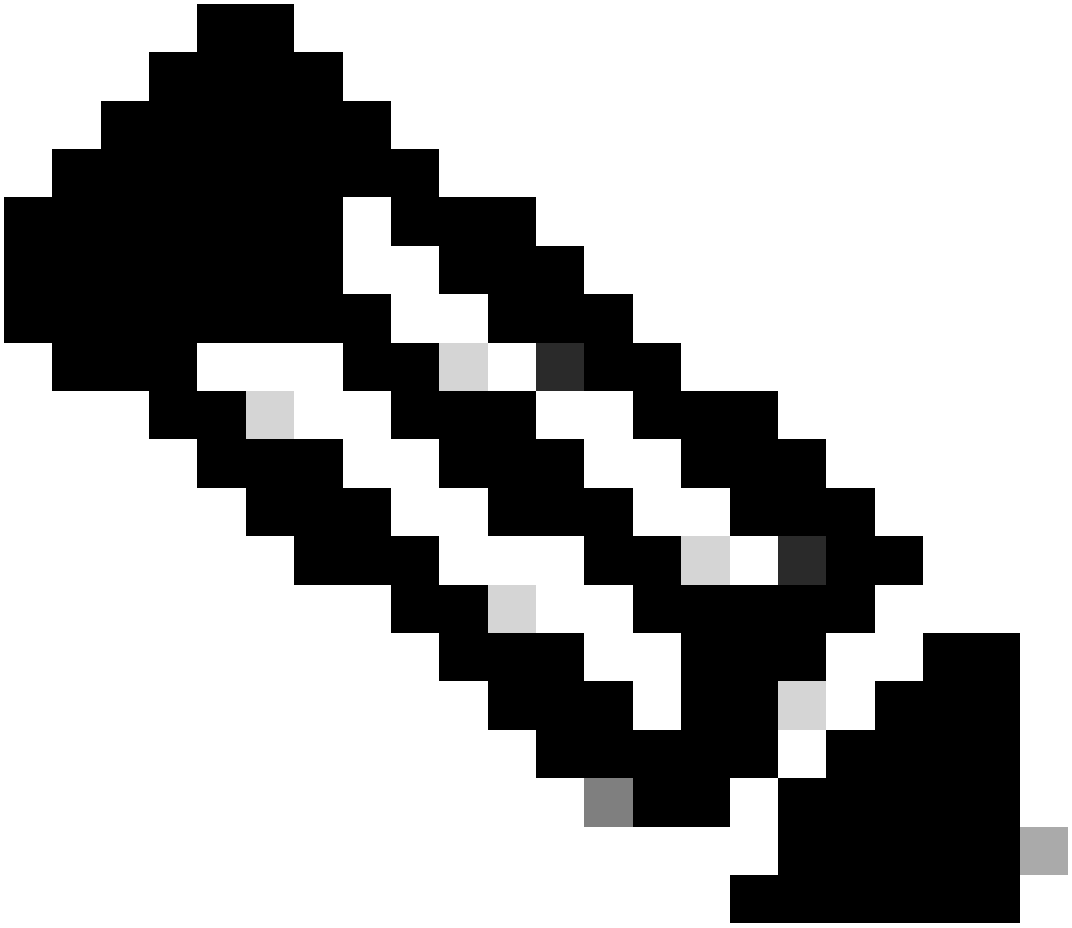
- RSA-Public Key of Server (para ayudar al cliente a autenticar el servidor)
- DH: clave pública del servidor (para calcular el secreto compartido)
- HASH (para autenticar el servidor y probar que el servidor ha generado el secreto compartido, ya que la clave secreta forma parte del cálculo del hash)



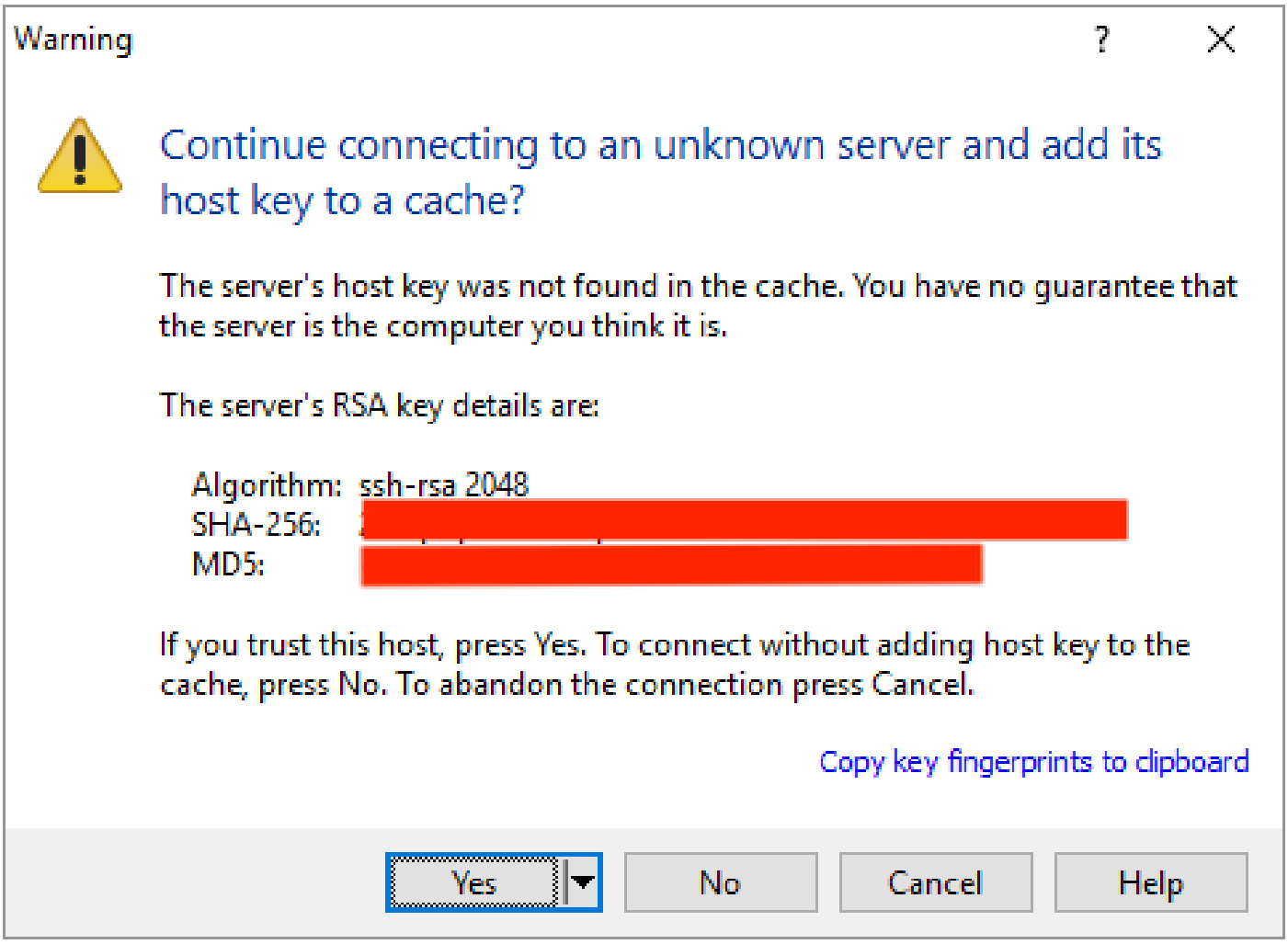
Respuesta de Server DH Public Key y Diffie-Hellman Group Exchange

f. Después de recibir DH_Exchange_Reply, el cliente calcula el hash de la misma manera y lo compara con el hash recibido, descifrándolo mediante la clave pública RSA del servidor.

g. Antes de descifrar el HASH recibido, el cliente debe verificar la clave pública del servidor. Esta verificación se realiza a través de un certificado digital firmado por una autoridad de certificación (CA). Si el certificado no existe, corresponde al cliente decidir si acepta la clave pública del servidor.



Nota: Cuando inicie SSH por primera vez en un dispositivo que no utilice un certificado digital, es posible que aparezca una ventana emergente solicitándole que acepte manualmente la clave pública del servidor. Para evitar ver esta ventana emergente cada vez que se conecte, puede agregar la clave de host del servidor a la caché.

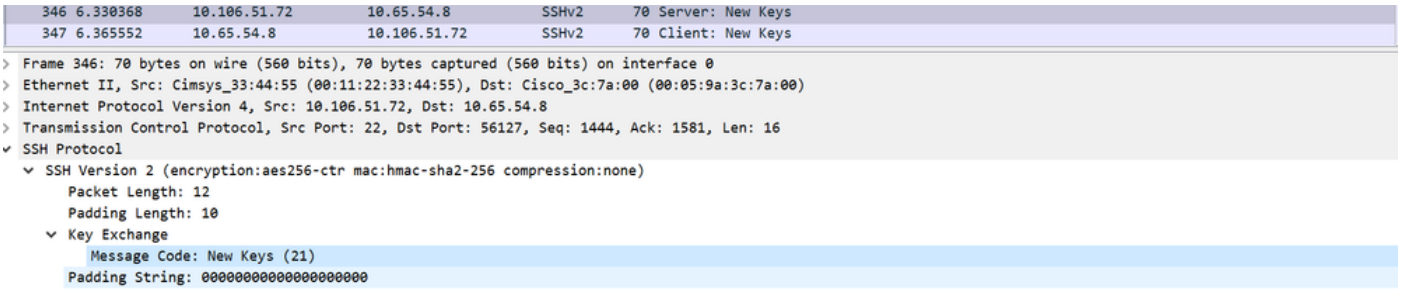


Clave RSA del servidor

4. Dado que el secreto compartido se genera ahora, ambos extremos lo utilizan para derivar estas claves :

- Claves de cifrado
- Teclas IV: son números aleatorios que se utilizan como entrada de algoritmos simétricos para mejorar la seguridad
- Claves de integridad

El final del intercambio de claves se señala mediante el intercambio del NEW KEYS' mensaje, que informa a cada parte que todos los mensajes futuros serán cifrados y protegidos usando estas nuevas claves .



Claves nuevas de cliente y servidor

5. El paso final es la solicitud de servicio. El cliente envía un paquete de petición de servicio SSH al servidor para iniciar la autenticación de usuario. El servidor responde con un mensaje de aceptación del servicio SSH, solicitando al cliente que inicie sesión. Este intercambio ocurre sobre el canal seguro establecido.

Información Relacionada

- https://www.cisco.com/c/en/us/support/docs/security-vpn/secure-shell-ssh/4145-ssh.html
- https://datatracker.ietf.org/doc/html/rfc4253
- Soporte técnico y descargas de Cisco

Acerca de esta traducción
Cisco ha traducido este documento combinando la traducción automática y los recursos humanos a fin de ofrecer a nuestros usuarios en todo el mundo contenido en su propio idioma.

Tenga en cuenta que incluso la mejor traducción automática podría no ser tan precisa como la proporcionada por un traductor profesional.

Cisco Systems, Inc. no asume ninguna responsabilidad por la precisión de estas traducciones y recomienda remitirse siempre al documento original escrito en inglés (insertar vínculo URL).