Entender o Secure Shell Packet Exchange

Contents

Introdução

Este documento descreve a troca de nível de pacote durante a negociação Secure Shell (SSH).

Pré-requisitos

Requisitos

A Cisco recomenda que você tenha conhecimento de conceitos básicos de segurança:

- Autenticação
- Confidencialidade
- Integridade
- Métodos de Troca de Chaves

Componentes Utilizados

Este documento não está restrito a uma versão de hardware específica.

As informações neste documento foram criadas a partir de dispositivos em um ambiente de laboratório específico. Todos os dispositivos utilizados neste documento foram iniciados com uma configuração (padrão) inicial.

Protocolo SSH

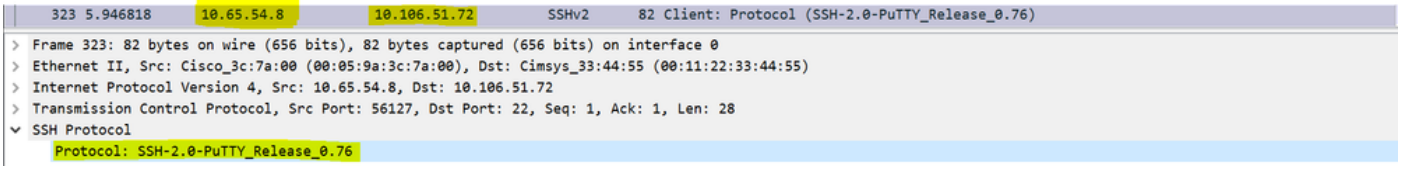
O protocolo SSH é um método para um login remoto seguro de um computador para outro. Os aplicativos SSH são baseados em uma arquitetura cliente-servidor, conectando uma instância de cliente SSH com um servidor SSH.

Troca SSH

1. A primeira etapa do SSH é chamada Identification String Exchange.

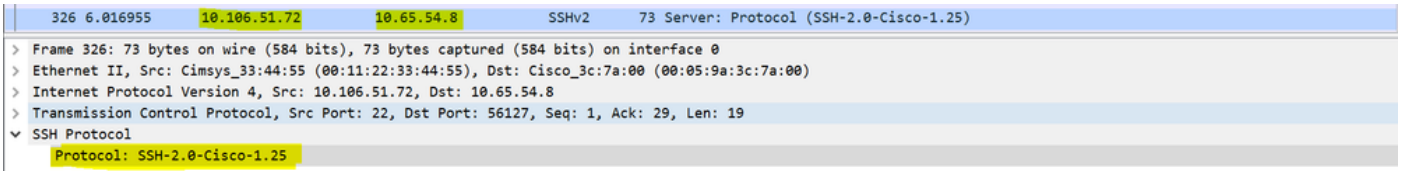
a. O cliente constrói um pacote e o envia ao servidor contendo:

- Versão do protocolo SSH
- Versão de software



A versão do protocolo do cliente é SSH2.0 e a versão do software é Putty_0.76.

b. O servidor responde com seu próprio Identification String Exchange, incluindo a versão do protocolo SSH e a versão do software.

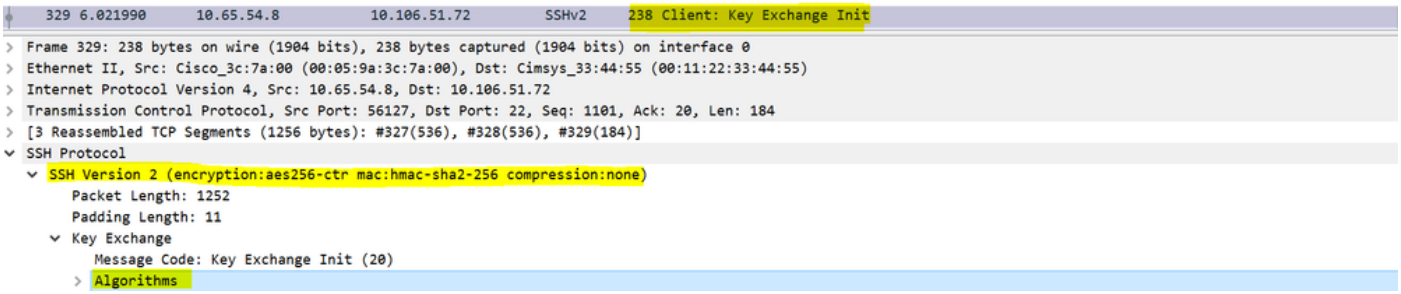


A versão do protocolo do servidor é SSH2.0 e a versão do software é Cisco1.25

2. Próxima Etapa é **Algorithm Negotiation**. Nesta etapa, o Cliente e o Servidor negociam estes algoritmos:

- Troca de chaves
- Criptografia
- HMAC (Hash-based Message Authentication Code, Código de Autenticação de Mensagem Baseado em Hash)
- Compressão

1. O cliente envia uma mensagem Key Exchange Init ao servidor, especificando os algoritmos aos quais dá suporte. Os algoritmos são listados em ordem de preferência.



Inicialização de Troca de Chaves

```
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 suportados pelo cliente

b. O servidor responde com sua própria mensagem Key Exchange Init, listando os algoritmos aos quais dá suporte.

c. Como essas mensagens são trocadas simultaneamente, ambas as partes comparam suas listas de algoritmos. Se houver uma correspondência nos algoritmos suportados por ambos os lados, eles passarão para a próxima etapa. Se não houver correspondência exata, o servidor seleciona o primeiro algoritmo na lista do cliente que ele também suporta.

d. Se o cliente e o servidor não conseguirem chegar a um acordo sobre um algoritmo comum, a troca de chaves falhará.

```
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
```

Inicialização de Troca de Chave de Servidor

3. Depois disso, ambos os lados entram na fase **Key Exchange**para gerar segredo compartilhado usando a troca de chave DH e autenticar o servidor:

a. O cliente gera um par de chavesPublic and Privatee envia a chave pública DH no pacote Init de troca de grupo DH. Esse par de chaves é usado para o cálculo de chave secreta.

```
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
```

Chave pública DH do cliente e inicialização de intercâmbio de grupo Diffie-Hellman

b. O servidor gera seu próprioPublic and Privatepar de chaves. Ele usa a chave pública do cliente e seu próprio par de chaves para computar o segredo compartilhado.

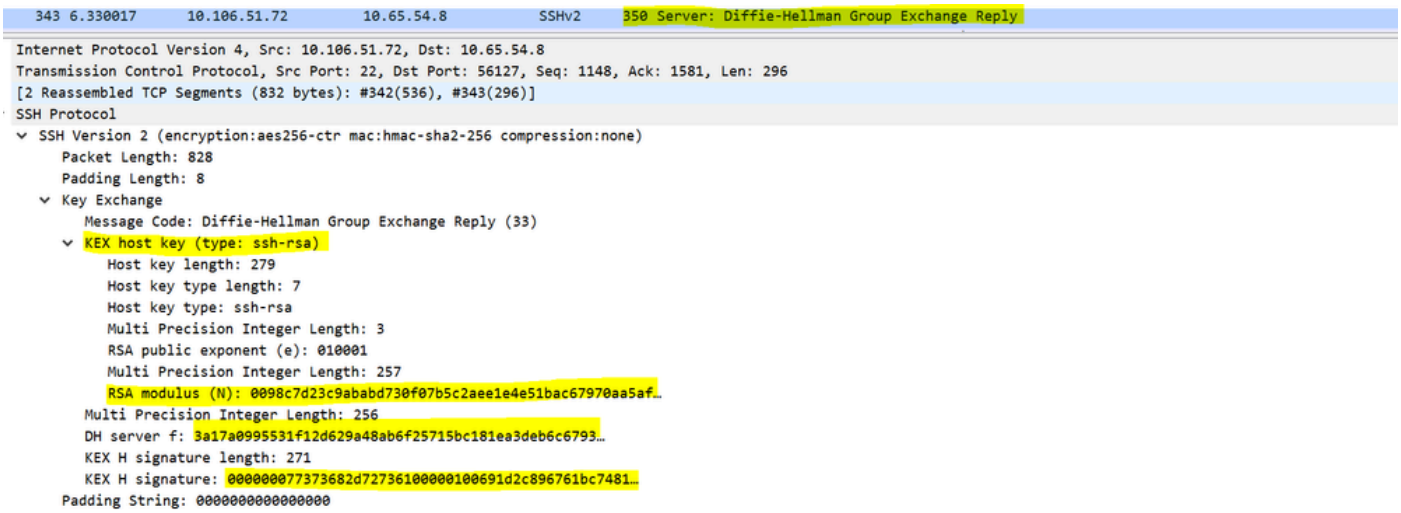
c. O servidor também calcula um hash do Exchange com estas entradas:

- Cadeia de Caracteres de Identificação de Clientes
- Cadeia de Caracteres de Identificação do Servidor
- Payload do cliente KEXINIT
- Carga do servidor KEXINIT
- Servers Public-key from Host keys (Par de chaves RSA)
- Chave pública DH dos clientes
- Chave pública DH dos servidores
- Chave Secreta Compartilhada

d. Depois de calcular o hash, o servidor o assina com sua chave privada RSA.

e. O servidor constrói uma mensagem DH_Exchange_Reply que inclui:

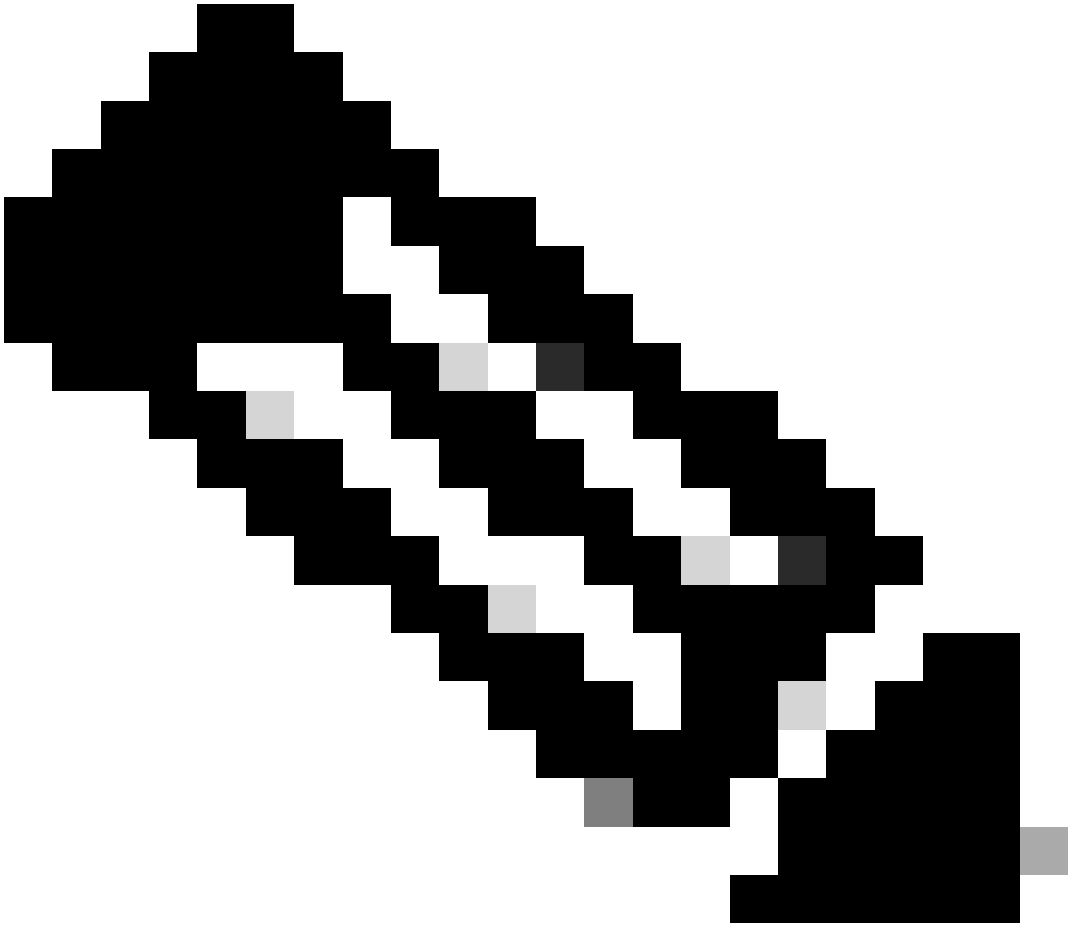
- RSA-Chave pública do servidor (para ajudar o cliente a autenticar o servidor)
- Chave DH-Pública do Servidor (para calcular o segredo compartilhado)
- HASH (para autenticar o servidor e provar que o servidor gerou o segredo compartilhado, pois a chave secreta faz parte do cálculo do hash)



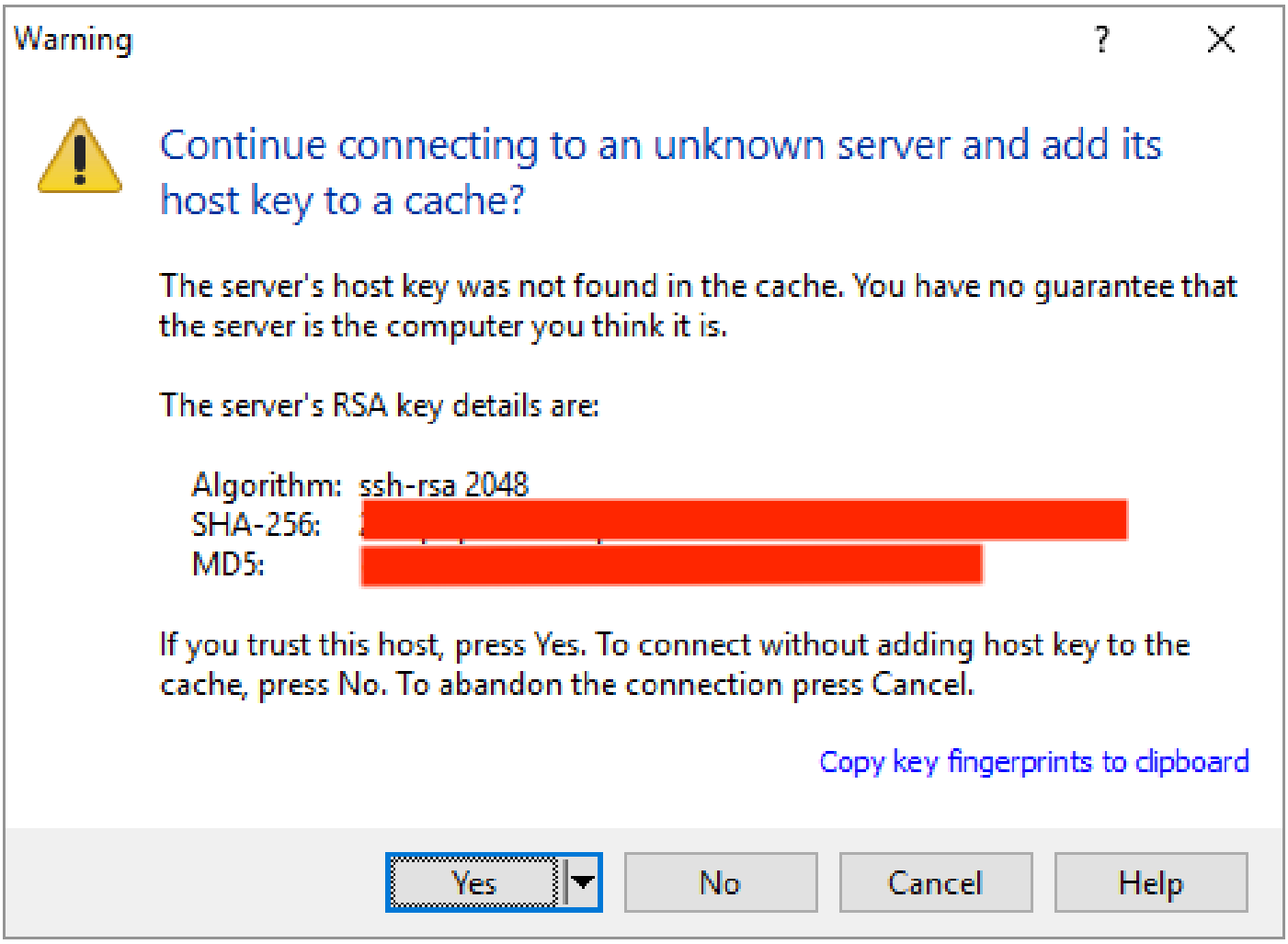
Resposta de Troca de Grupo Diffie-Hellman e Chave Pública DH do Servidor

f. Após receber a DH_Exchange_Reply, o cliente calcula o hash da mesma forma e o compara com o hash recebido, descriptografando-o usando a chave pública RSA do servidor.

g. Antes de descriptografar o HASH recebido, o cliente deve verificar a chave pública do servidor. Essa verificação é feita por meio de um certificado digital assinado por uma CA (Autoridade de Certificação). Se o certificado não existir, cabe ao cliente decidir se aceita a chave pública do servidor.



Observação: quando você usa pela primeira vez o SSH em um dispositivo que não usa um certificado digital, você pode encontrar um pop-up solicitando que você aceite manualmente a chave pública do servidor. Para evitar que esse pop-up seja exibido toda vez que você se conectar, você pode optar por adicionar a chave de host do servidor ao cache.

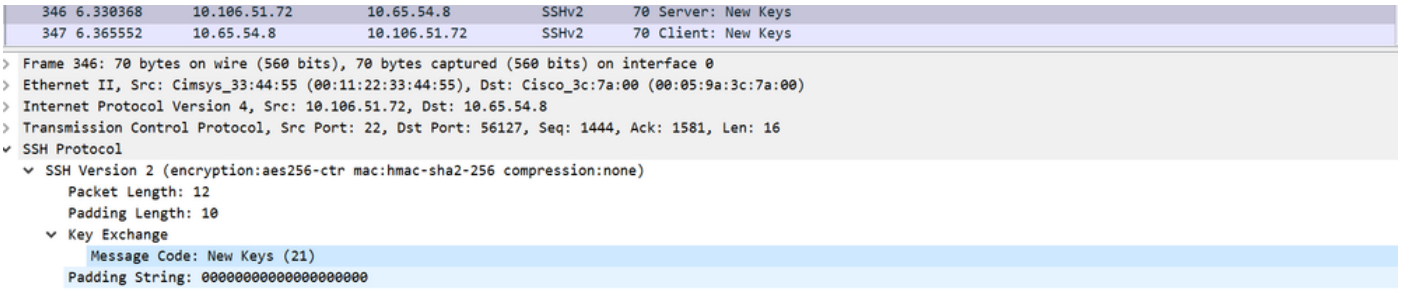


Chave RSA do servidor

4. Como o segredo compartilhado agora é gerado, ambos os endsl o usam para derivar essas chaves :

- Chaves de criptografia
- Chaves IV - São números aleatórios usados como entrada para algoritmos simétricos para aumentar a segurança
- Chaves de integridade

O fim da troca de chaves é sinalizado pela troca da NEW KEYS' mensagem, que informa a cada parte que todas as mensagens futuras serão criptografadas e protegidas usando essas novas chaves .



Chaves novas do cliente e do servidor

5. A etapa final é a Solicitação de Serviço. O cliente envia um pacote de solicitação de serviço SSH ao servidor para iniciar a autenticação do usuário. O servidor responde com uma mensagem SSH Service Accept, solicitando que o cliente faça login. Essa troca ocorre através do canal seguro estabelecido.

Informações Relacionadas

- https://www.cisco.com/c/en/us/support/docs/security-vpn/secure-shell-ssh/4145-ssh.html
- https://datatracker.ietf.org/doc/html/rfc4253
- Suporte técnico e downloads da Cisco

Sobre esta tradução
A Cisco traduziu este documento com a ajuda de tecnologias de tradução automática e humana para oferecer conteúdo de suporte aos seus usuários no seu próprio idioma, independentemente da localização.

Observe que mesmo a melhor tradução automática não será tão precisa quanto as realizadas por um tradutor profissional.

A Cisco Systems, Inc. não se responsabiliza pela precisão destas traduções e recomenda que o documento original em inglês (link fornecido) seja sempre consultado.