# Secure Shell 패킷 교환 이해

## 목차



## 소개

이 문서에서는 SSH(Secure Shell) 협상 중 패킷 레벨 교환에 대해 설명합니다.

## 사전 요구 사항

### 요구 사항

기본 보안 개념에 대한 지식을 보유하고 있는 것이 좋습니다.

- 인증
- 기밀 보장
- <Z2>신뢰</Z1><Z4>성</Z3>
- 키 교환 방법

### 사용되는 구성 요소

이 문서는 특정 하드웨어 버전으로 제한되지 않습니다.

이 문서의 정보는 특정 랩 환경의 디바이스를 토대로 작성되었습니다. 이 문서에 사용된 모든 디바이스는 초기화된(기본) 컨피그레이션으로 시작되었습니다.

## SSH 프로토콜

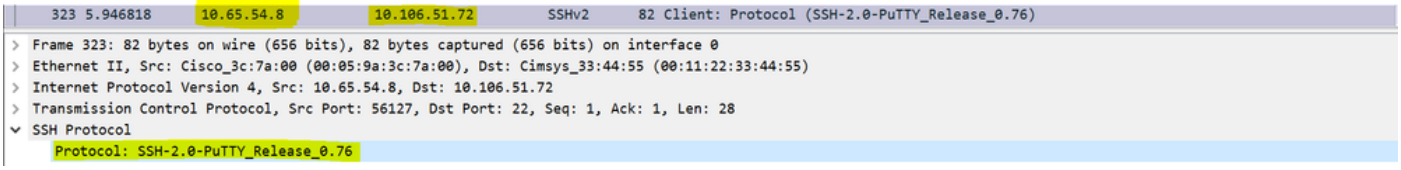
SSH 프로토콜은 한 컴퓨터에서 다른 컴퓨터로 안전하게 원격 로그인하는 방법입니다. SSH 애플리케이션은 클라이언트-서버 아키텍처를 기반으로 하며 SSH 클라이언트 인스턴스를 SSH 서버에 연결합니다.

# SSH 교환

1. SSH의 첫 번째 단계를 호출합니다 Identification String Exchange.

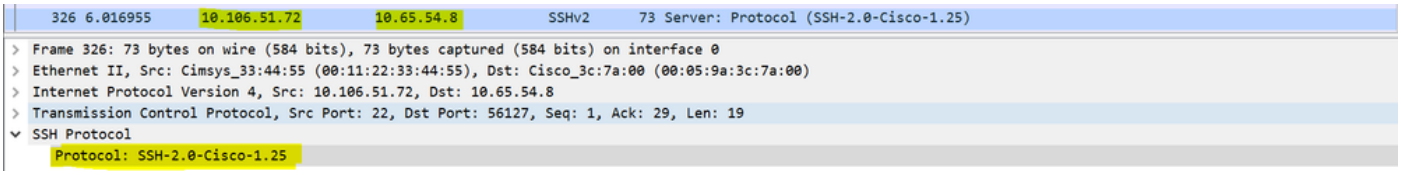
a. 클라이언트가 패킷을 구성하고 다음 항목을 포함하는 서버로 전송합니다.

- SSH 프로토콜 버전
- 소프트웨어 버전




클라이언트 프로토콜 버전은 SSH2.0이고 소프트웨어 버전은 Putty_0.76입니다.

b. 서버는 SSH 프로토콜 버전 및 소프트웨어 버전을 포함하여 자체 ID 문자열 교환으로 응답합니다
.

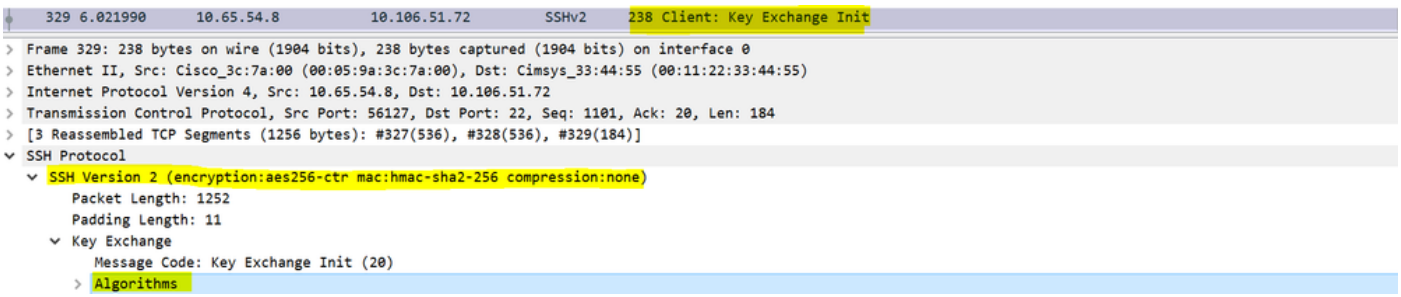



서버의 프로토콜 버전은 SSH2.0이고 소프트웨어 버전은 Cisco1.25입니다.

2. 다음 단계는 **Algorithm Negotiation**. 이 단계에서 클라이언트와 서버 모두 다음 알고리즘을 협상합니다
.

- 키 교환
- 암호화
- HMAC(해시 기반 메시지 인증 코드)
- 압축

1. 클라이언트는 지원하는 알고리즘을 지정하여 Key Exchange Init 메시지를 서버에 보냅니다. 알고리즘은 기본 설정 순서대로 나열되어 있습니다.




키 교환 초기화

```
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
```

클라이언트 지원 알고리즘

b. 서버는 자체 Key Exchange Init 메시지로 응답하며, 지원하는 알고리즘을 나열합니다.

c. 이러한 메시지는 동시에 교환되므로 양 당사자가 알고리즘 목록을 비교합니다. 양쪽이 모두 지원하는 알고리즘에서 일치가 있으면 다음 단계로 진행한다. 정확히 일치하는 항목이 없는 경우 서버는 클라이언트 목록에서 역시 지원하는 첫 번째 알고리즘을 선택합니다.

d. 클라이언트와 서버가 공통 알고리즘에 동의할 수 없는 경우 키 교환이 실패합니다.

```
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
```

서버 키 교환 초기화

3. 이후 양측이 DH 키 교환을 **Key Exchange**사용하여 공유 암호를 생성하고 서버를 인증하는 단계로 들어갑니다.

a. 클라이언트가 키 쌍을 생성하고Public and PrivateDH 그룹 교환 초기화 패킷에서 DH 공개 키를 전송합니다. 이 키 쌍은 비밀 키 계산에 사용됩니다.

```
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
```

클라이언트 DH 공개 키 및 Diffie-Hellman 그룹 교환 초기화

b. 서버가 자체Public and Private키 쌍을 생성합니다. 클라이언트의 공개 키 및 자체 키 쌍을 사용하여 공유 암호를 계산합니다.

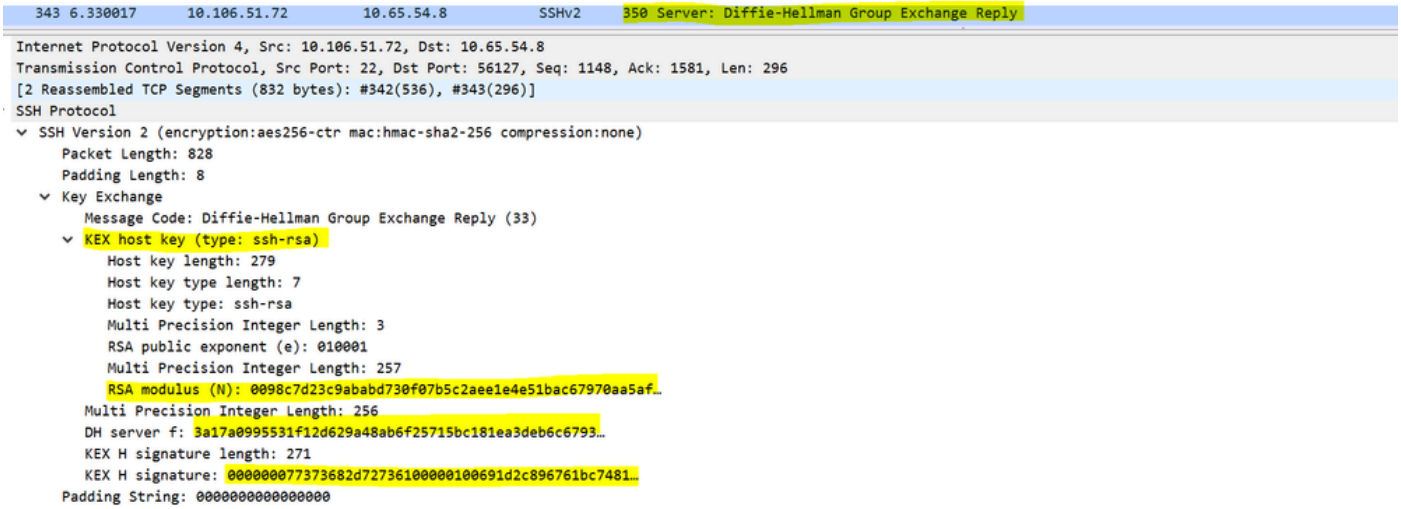
c. 또한 서버는 다음 입력으로 Exchange 해시를 계산합니다.

- 클라이언트 식별 문자열
- 서버 식별 문자열
- 클라이언트 KEXINIT의 페이로드
- 서버 KEXINIT의 페이로드
- 호스트 키의 서버 공개 키( RSA 키 쌍 )
- 클라이언트 DH 공개 키
- 서버 DH 공개 키
- 공유 암호 키

d. 해시를 계산한 후 서버는 RSA 개인 키로 서명합니다.

e. 서버는 다음을 포함하는 메시지 DH_Exchange_Reply를 구성합니다.

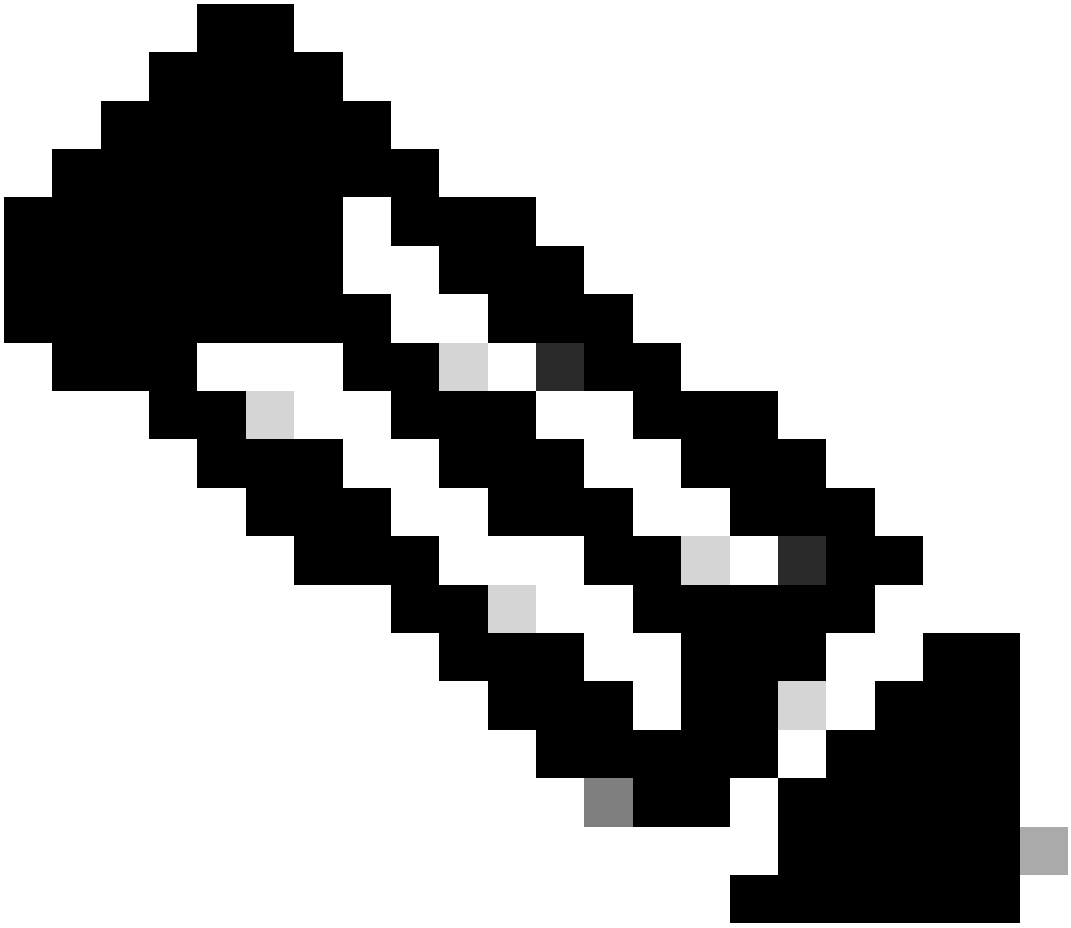
- 서버의 RSA 공개 키(클라이언트가 서버를 인증할 수 있도록 지원)
- 서버의 DH 공개 키(공유 암호 계산용)
- HASH(비밀 키가 해시 계산의 일부이므로 서버를 인증하고 서버에서 공유 비밀을 생성했음을 증명)



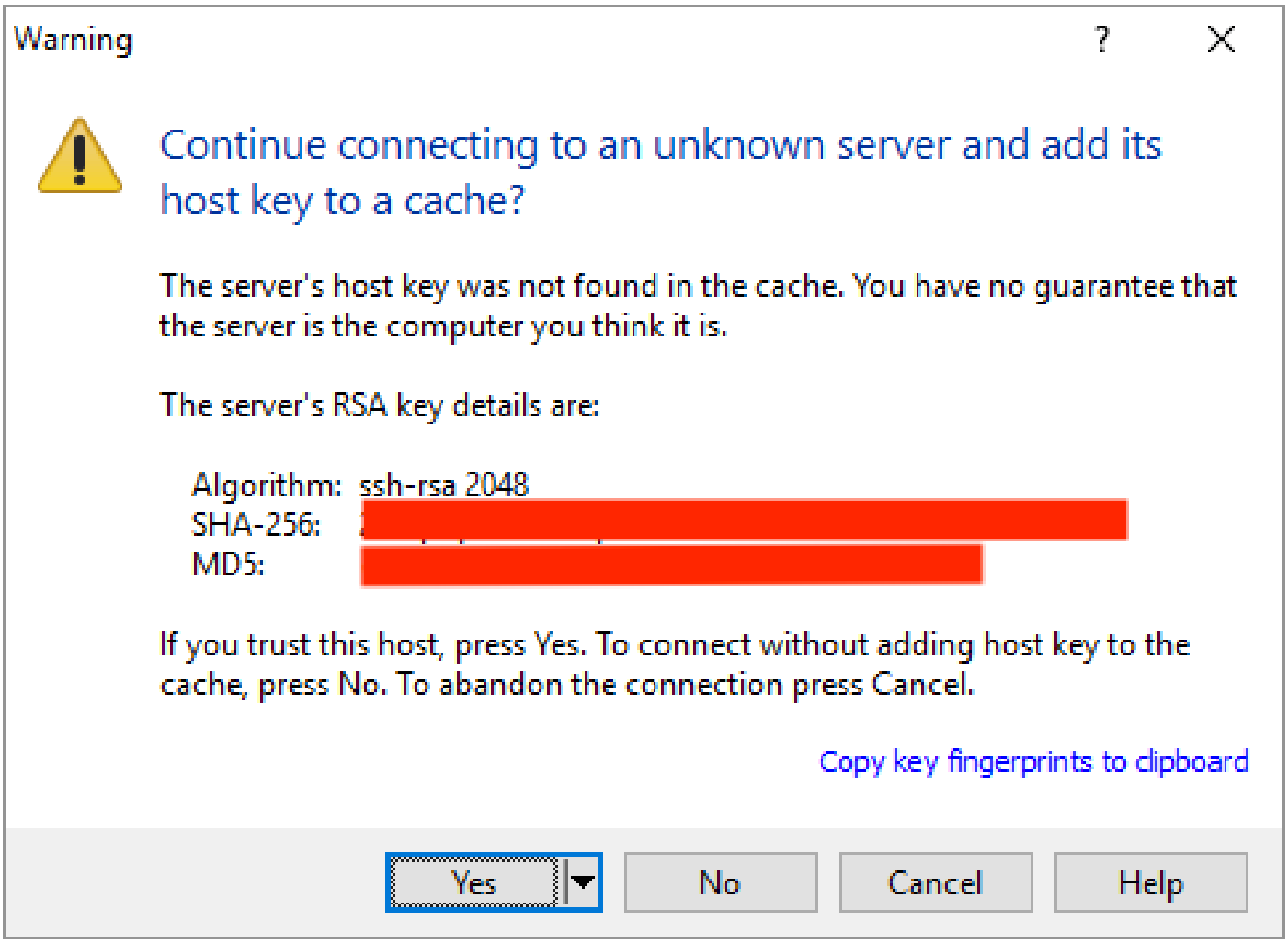

서버 DH 공개 키 및 Diffie-Hellman 그룹 교환 회신

f. 클라이언트가 DH_Exchange_Reply를 수신한 후 동일한 방법으로 해시를 계산하고 이를 수신한 해시와 비교하여 서버의 RSA 공개 키를 사용하여 해독 작업을 수행합니다.

g. 수신된 HASH를 해독하기 전에 클라이언트는 서버의 공개 키를 확인해야 합니다. 이 확인은 CA(Certificate Authority)에서 서명한 디지털 인증서를 통해 수행됩니다. 인증서가 존재하지 않는 경우, 서버의 공개 키를 수락할지 여부는 클라이언트에 달려 있습니다.



참고: 디지털 인증서를 사용하지 않는 디바이스에 SSH를 처음 적용하면 서버의 공개 키를 수동으로 수락하라는 팝업이 표시될 수 있습니다. 연결할 때마다 이 팝업이 표시되지 않도록 하려면 서버의 호스트 키를 캐시에 추가하도록 선택할 수 있습니다.

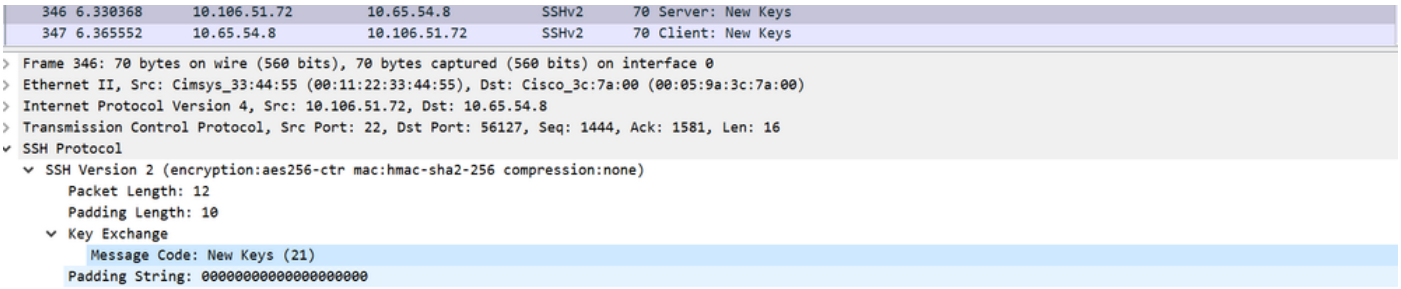



서버의 RSA 키

4. 공유 비밀이 생성되었으므로 두 엔드에서는 공유 비밀을 사용하여 다음 키를 파생합니다.

- 암호화 키
- IV 키 - 이는 보안을 강화하기 위해 대칭 알고리즘에 대한 입력으로 사용되는 난수입니다.
- 무결성 키

키 교환의 종료는 메시지 교환에 의해 신호되며, 이는 NEW KEYS' 각 당사자에게 모든 향후 메시지가 이러한 새 키를 사용하여 암호화되고 보호됨을 알립니다.




클라이언트 및 서버 새 키

5. 마지막 단계는 서비스 요청입니다. 클라이언트는 SSH 서비스 요청 패킷을 서버에 전송하여 사용자 인증을 시작합니다. 서버는 SSH Service Accept(SSH 서비스 수락) 메시지로 응답하면서 클라이언트에 로그인하라는 메시지를 표시합니다. 이 교환은 설정된 보안 채널을 통해 이루어집니다.

## 관련 정보

- https://www.cisco.com/c/en/us/support/docs/security-vpn/secure-shell-ssh/4145-ssh.html
- https://datatracker.ietf.org/doc/html/rfc4253
- Cisco 기술 지원 및 다운로드

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.