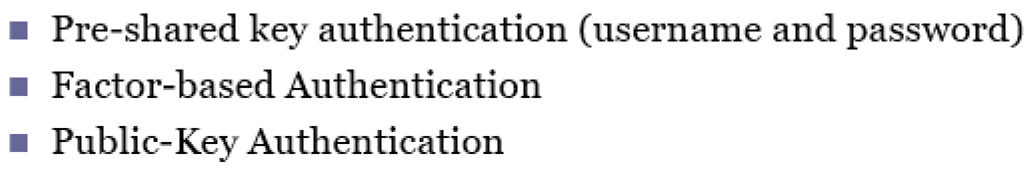
Public-key authentication

- In conventional password authentication, you prove you are who you claim to be by proving that you know the correct password. This means that if the server has been hacked, or spoofed, an attacker can learn your password.

- Public key authentication solves this problem. Public key authentication is an alternative means of identifying yourself to a login server, instead of typing a password. It is more secure and more flexible, but more difficult to set up.

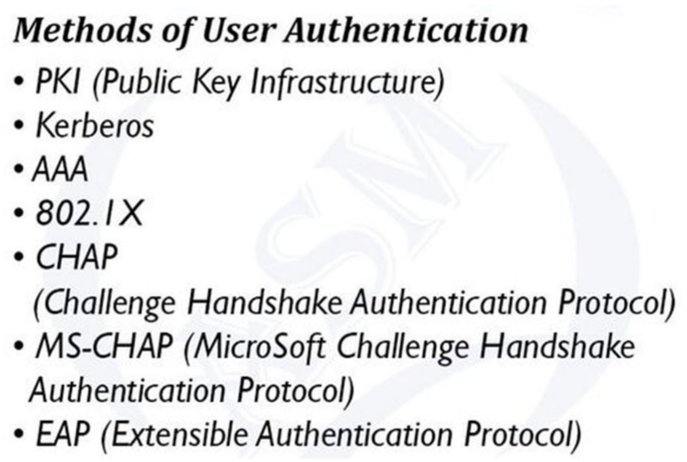
- The motivation for using public key authentication over simple passwords is security.



- Pre-shared key authentication (username and password)
- Factor-based Authentication
- Public-Key Authentication

Public-key authentication methods

- The public-key authentication methods include:
  - Digital certificates
  - Authentication using SSH
  - Authentication using SSL
  - IPsec
  - Authentication using Third-Party Services
    - Authentication with RADIUS
    - Hashing algorithms
    - Kerberos Authentication
    - Directory-Based Services

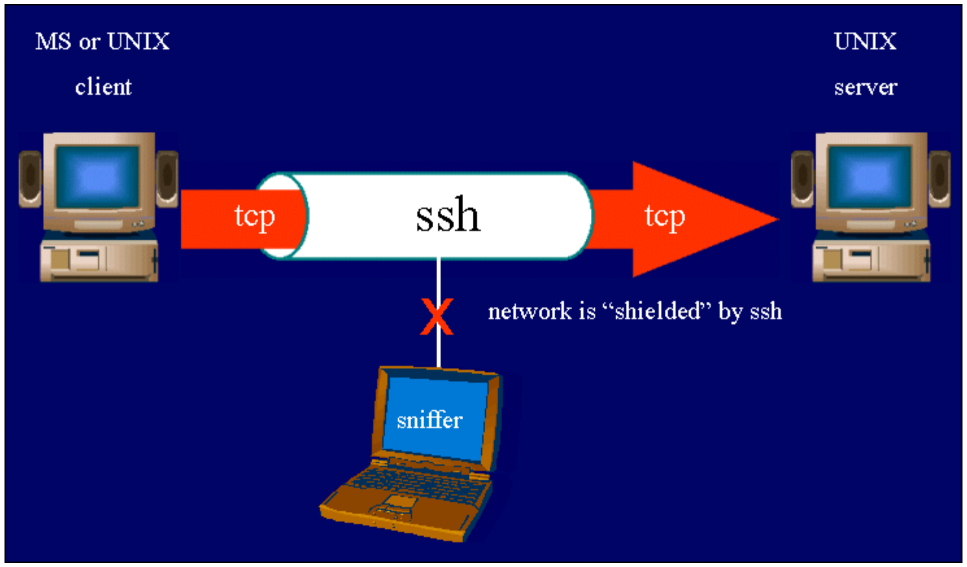


Methods of User Authentication

- *PKI (Public Key Infrastructure)*
- *Kerberos*
- *AAA*
- *802.1X*
- *CHAP (Challenge Handshake Authentication Protocol)*
- *MS-CHAP (MicroSoft Challenge Handshake Authentication Protocol)*
- *EAP (Extensible Authentication Protocol)*

SSH protocol

- The password authentication is the worst type of authentication. It is not safe as it is hacked all the time. Instead we set up SSH keys.
- Secure Shell (SSH) provides a secure channel over an insecure network in a client-server architecture, connecting an SSH client application with an SSH server.
- Using SSH authentication to connect to a remote system is a robust, more secure alternative to logging in with an account password or passphrase.
- SSH uses public-key cryptography to authenticate the remote computer and allow it to authenticate the user, if necessary.



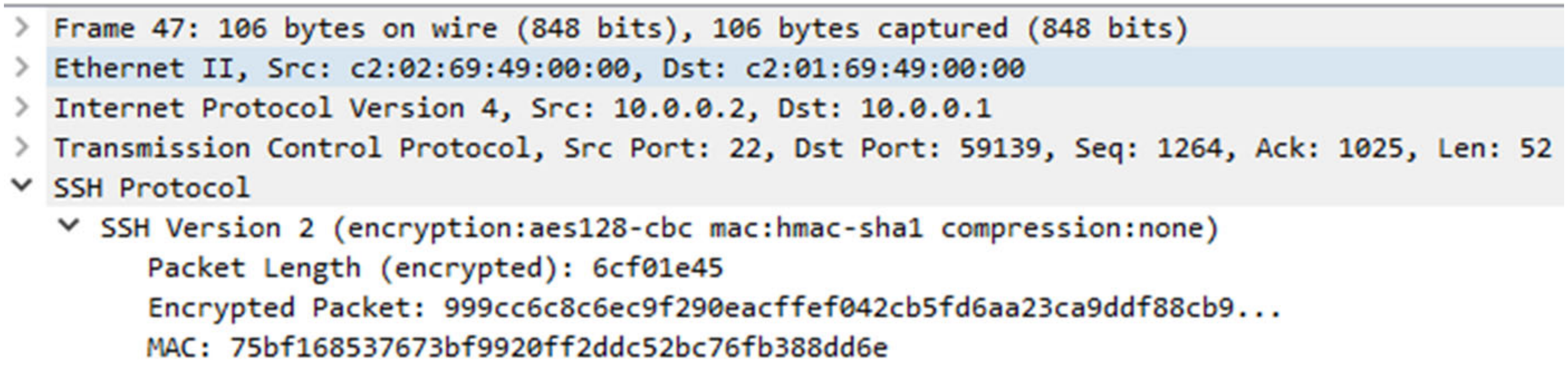
SSH protocol cont…

SSH protocol cont…

- The SSH is the replacement for various shell based protocols such as telnet, and other remote logins as well as file transfer protocols such as FTP and remote file copy protocols such as RCP.
- The deficiency with these protocols is that when you authenticate yourself to a server, it is always sent in the clear. An eavesdropper can easily intercept these requests and masquerade as a potential user. Also, all subsequent data that is transmitted is in plaintext.
- Corporations must disable telnet, ftp, and remote login services and replace them with the secured versions SSH, SFTP and SCP.

SSH protocol cont…

- SSH works on top of TCP/IP.



```
> Frame 47: 106 bytes on wire (848 bits), 106 bytes captured (848 bits)
> Ethernet II, Src: c2:02:69:49:00:00, Dst: c2:01:69:49:00:00
> Internet Protocol Version 4, Src: 10.0.0.2, Dst: 10.0.0.1
> Transmission Control Protocol, Src Port: 22, Dst Port: 59139, Seq: 1264, Ack: 1025, Len: 52
v SSH Protocol
  v SSH Version 2 (encryption:aes128-cbc mac:hmac-sha1 compression:none)
      Packet Length (encrypted): 6cf01e45
      Encrypted Packet: 999cc6c8c6ec9f290eacffef042cb5fd6aa23ca9ddf88cb9...
      MAC: 75bf168537673bf9920ff2ddc52bc76fb388dd6e
```

SSH protocol cont…

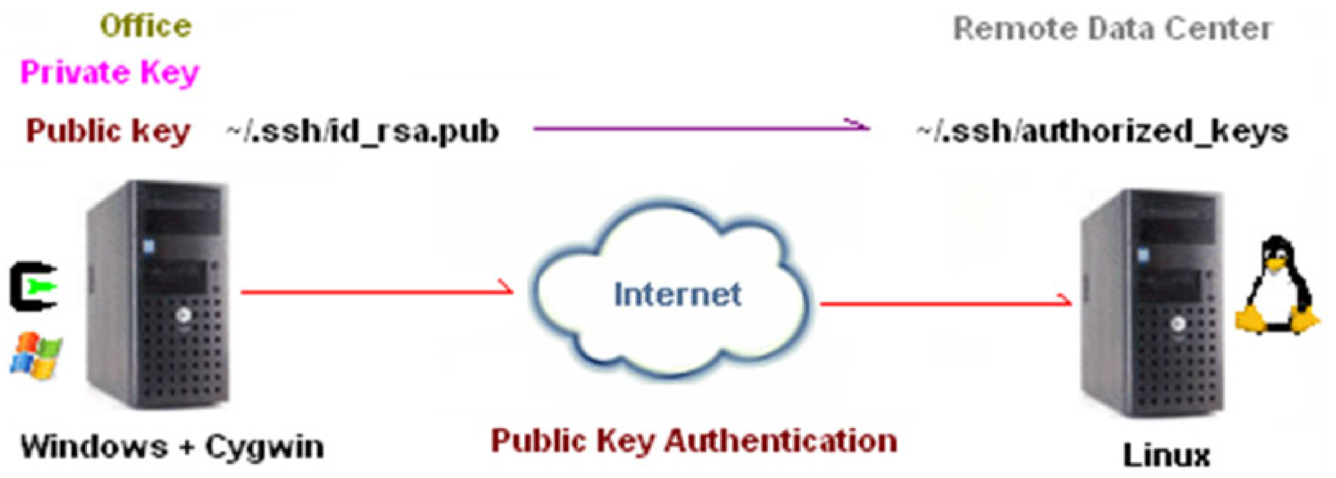
- The SSH connection between the client and the server/host happens in seven stages:
  1. Installing openssh
  2. Generating SSH keys by both client and server
  3. Exchanging the public keys
  4. Verification of the server by the client.
  5. Generation of a session key to encrypt all the communication.
  6. Authentication of the client
  7. Transmission of data

1: Installing openssh

- To use SSH algorithm:
  - The remote server must have a version of SSH installed such as OpenSSH or Tectia SSH (openssh-server).
  - Additionally, the computer you use to connect to the remote server also must have a version of SSH installed (openssh-client) and a tool such as command-line SSH on a Linux or macOS computer, or PuTTY on a Windows computer.

2: Generating SSH keys

- In order to have a secure connection SSH makes two keys:
  - a private key: which remains (only) on our own computer to identify this computer. The possession of this key is proof of the identity. The private keys need to be stored and handled carefully, and no copies of the private key should be distributed. The private keys used for authentication are called identity keys.
  - a public key: which we upload it to the server that we try to connect to it. A public key that is shared with others. Such keys are called authorized keys.



2: Generating SSH keys cont…

- SSH supports several public key algorithms, including RSA, DSA, ECDSA, ED25519, which can be used to generate SSH key pair.

- You can generate the SSH key pair on cisco or Linux machines using any of the following commands:
-
- ssh-keygen -t rsa -b 4096
- ssh-keygen -t dsa
- ssh-keygen -t ecdsa -b 521
- ssh-keygen -t ed25519

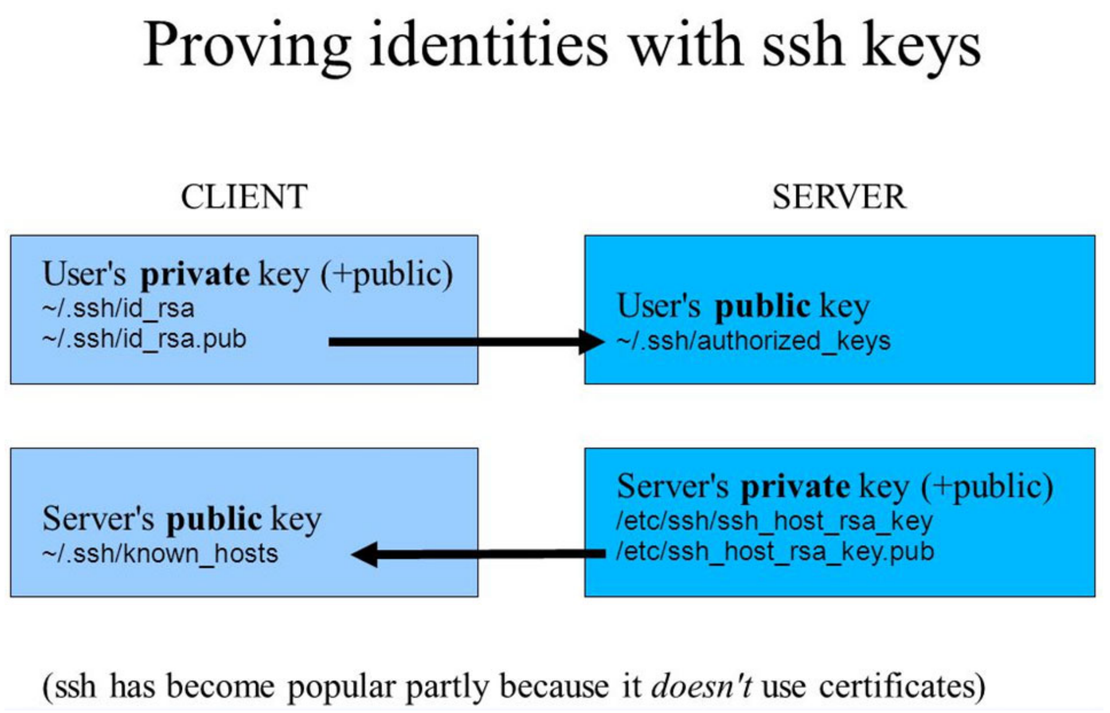
- The type of algorithm is selected using the -t option and key size bit using the -b option. Note that RSA is the default type and default key size is 2048 bits.

3: Exchanging the public keys

- Now we need to transfer client public key to the server and server public key to the client system.
- Because public key can be shared with anyone so there is no problem sharing it.
- This can be done with different methods such as using wither:
  - ssh-copy-id command (ssh-copy-id username@remote_host)
  - Use SFTP or SCP to copy the public key file (scp id_rsa.pub 192.168.2.105:/home/)
  - log into the remote system with an telnet established account username and password
  - have an administrator on the remote system add the public key.

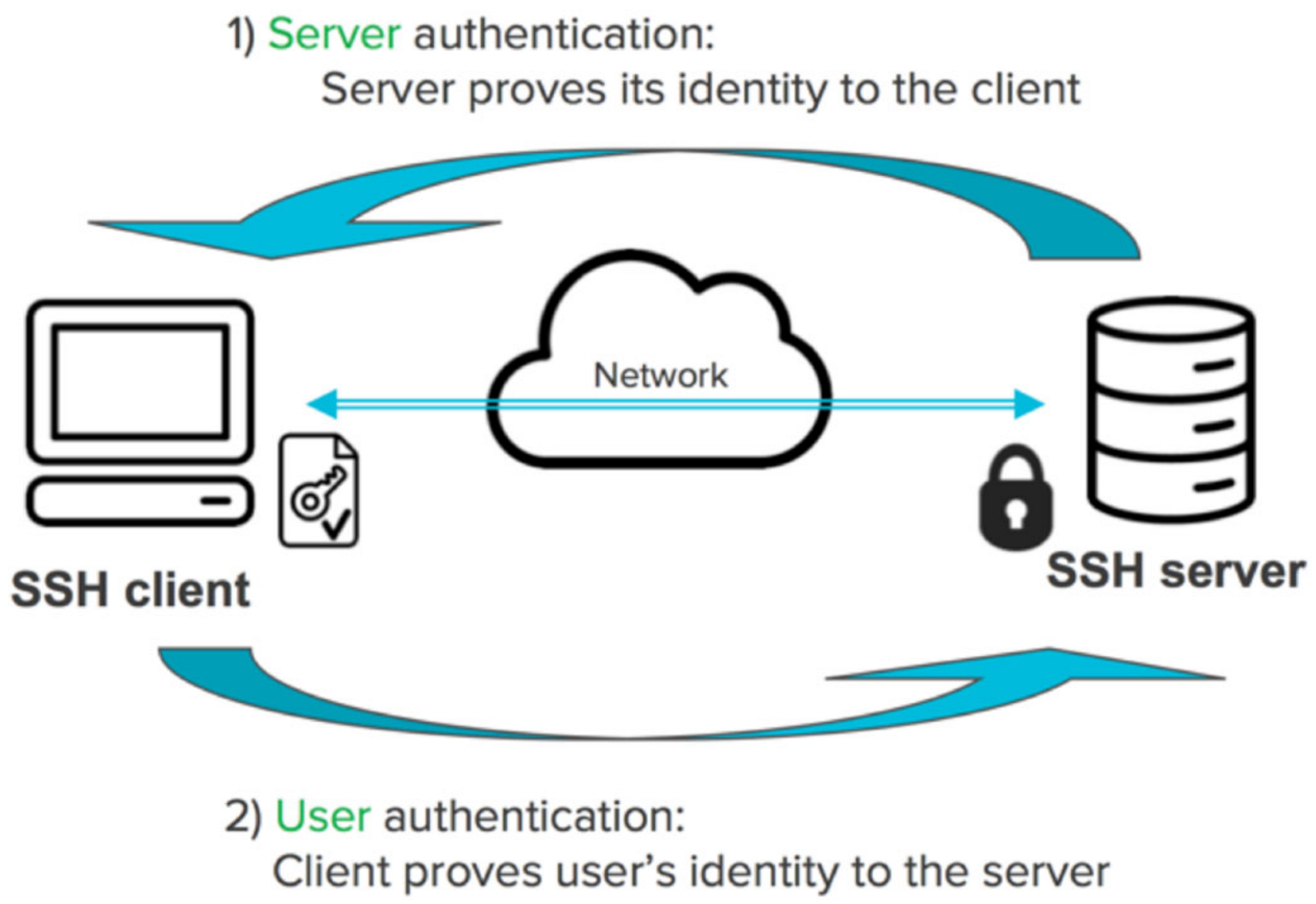
3: Exchanging the public keys cont…

- Whichever method is used:
  - The client public key is stored on the server in the $HOME/.ssh/authorized_keys file.
  - The server public key is stored on the client in the $HOME/.ssh/known_hosts file.


Proving identities with ssh keys
CLIENT
SERVER
User's **private** key (+public)
~/.ssh/id_rsa
~/.ssh/id_rsa.pub
User's **public** key
~/.ssh/authorized_keys
Server's **public** key
~/.ssh/known_hosts
Server's **private** key (+public)
/etc/ssh/ssh_host_rsa_key
/etc/ssh_host_rsa_key.pub
(ssh has become popular partly because it *doesn't* use certificates)

4: Verification of the server

- First server has to prove its identity to the client.


1) Server authentication:
Server proves its identity to the client
Network
SSH client
SSH server
2) User authentication:
Client proves user's identity to the server

4: Verification of the server cont…

- The client initiates a SSH connection with the server. Server listens to default port 22 (this port can be changed) for SSH connections. At this point, the server identity is verified.

- There are two cases:
  - If the client is accessing the server for first time
  - If the client is not accessing the server for the first time

4: Verification of the server cont…

- If the client is not accessing the server for the first time, the server sends its public key to the client.
- The client compares this identity with previously recorded information in known_hosts file for verification. If they are the same, the server has been verified successfully.

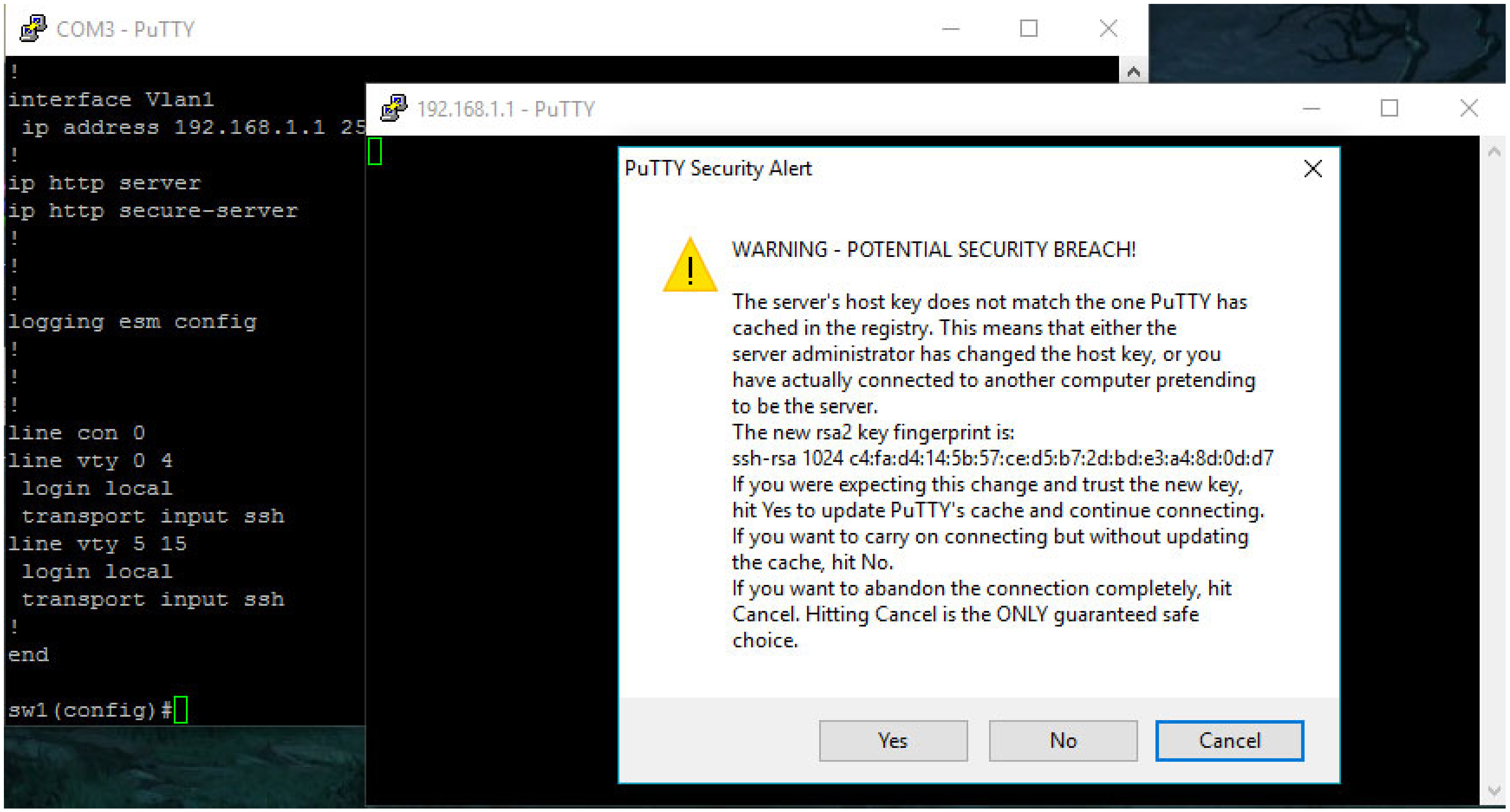
4: Verification of the server cont…

- If the client is accessing the server for first time, it will present a host key fingerprint and the client is asked to authenticate server manually.

- The idea is to prevent a Man-in-the-Middle (MITM) attack by verifying that the server's public key is what we expect to get. To prevent this attack, each server has a unique identifying code, called a fingerprint host key. These keys prevent a server from forging another server's key. A public key fingerprint is a short sequence of bytes used to identify a longer public key. Fingerprints are created by applying a hash function to a public key.

- Once the key is verified, the server is added in known_hosts file in ~/.ssh directory on client machine. The known_hosts file contains the information about all the verified servers by the client.

4: Verification of the server cont…



```
!
interface Vlan1
 ip address 192.168.1.1 25
!
ip http server
ip http secure-server
!
!
!
logging esm config
!
!
!
line con 0
line vty 0 4
 login local
 transport input ssh
line vty 5 15
 login local
 transport input ssh
!
end

sw1(config)#
```

WARNING - POTENTIAL SECURITY BREACH!

The server's host key does not match the one PuTTY has cached in the registry. This means that either the server administrator has changed the host key, or you have actually connected to another computer pretending to be the server.
The new rsa2 key fingerprint is:
ssh-rsa 1024 c4:fa:d4:14:5b:57:ce:d5:b7:2d:bd:e3:a4:8d:0d:d7
If you were expecting this change and trust the new key, hit Yes to update PuTTY's cache and continue connecting.
If you want to carry on connecting but without updating the cache, hit No.
If you want to abandon the connection completely, hit Cancel. Hitting Cancel is the ONLY guaranteed safe choice.

5: Generation of a session key

- After the server is verified, both the parties negotiate a session key using Diffie-Hellman algorithm. This algorithm (and its variants) make it possible for each party to combine their own private data with public data from the other system to arrive at an identical secret session key:
  - The client takes its private key and the server's public key and passes it through a mathematical equation to produce the shared secret (session key).
  - The server takes its private key and the client's public key and passes it through a mathematical equation to produce the shared secret (session key). Both these shared secrets are identical.
- **As we can see, this algorithm ensures that neither side can fully determine the session key.**

6: Authentication of the client

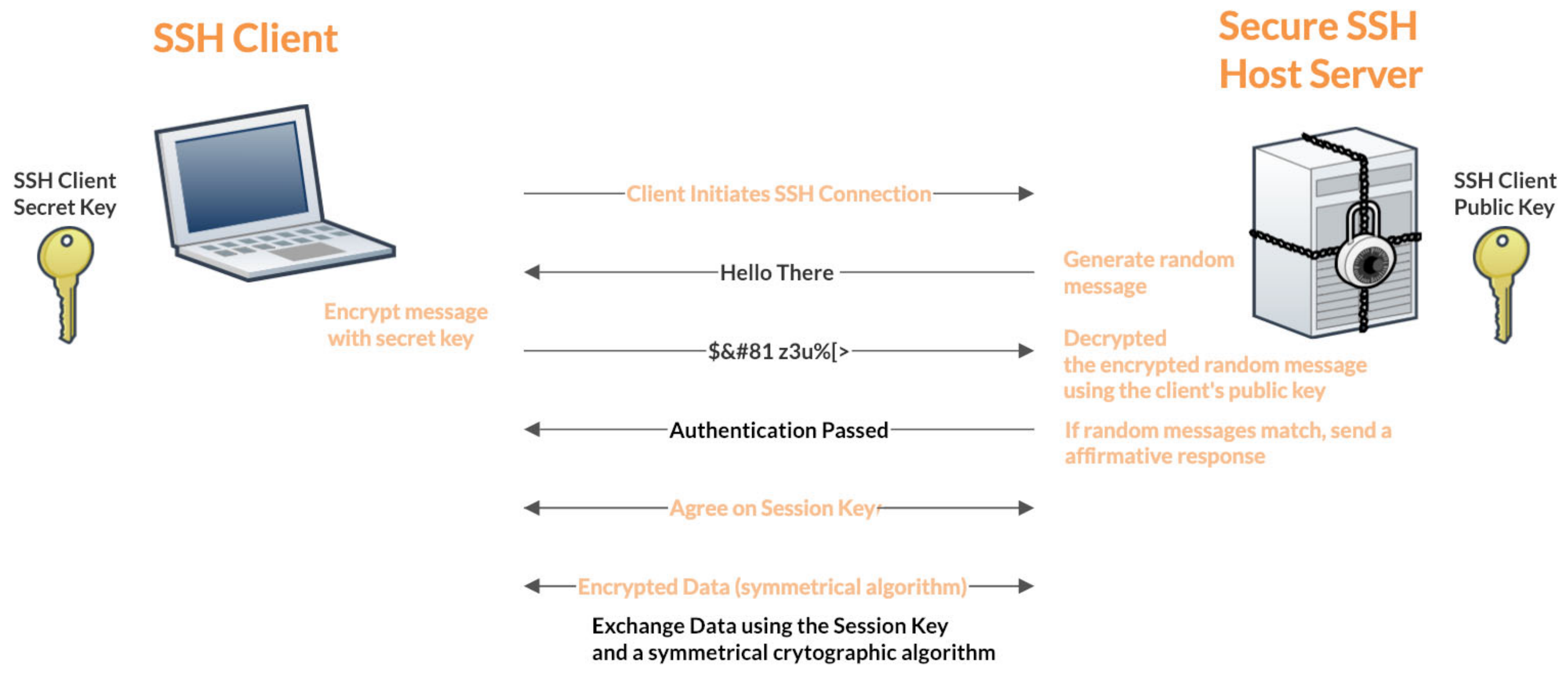
- Mutual/Two-Way Authentication refers to the combination of both Server and Client Authentication.
- The authentication is mutual, or two-way, because the server is authenticating itself to the client, and the client is authenticating itself to the server.

6: Authentication of the client cont…

- There are a few different methods that can be used for authentication, based on what the server accepts:
  - The simplest is probably password authentication, in which the server simply prompts the client for the password of the account they are attempting to login with. The password is sent through the negotiated encryption, so it is secure from outside parties. Even though the password will be encrypted, this method is not generally recommended due to the limitations on the complexity of the password. Automated scripts can break passwords of normal lengths very easily compared to other authentication methods.
  - The most popular and recommended alternative is the use of public key authentication.

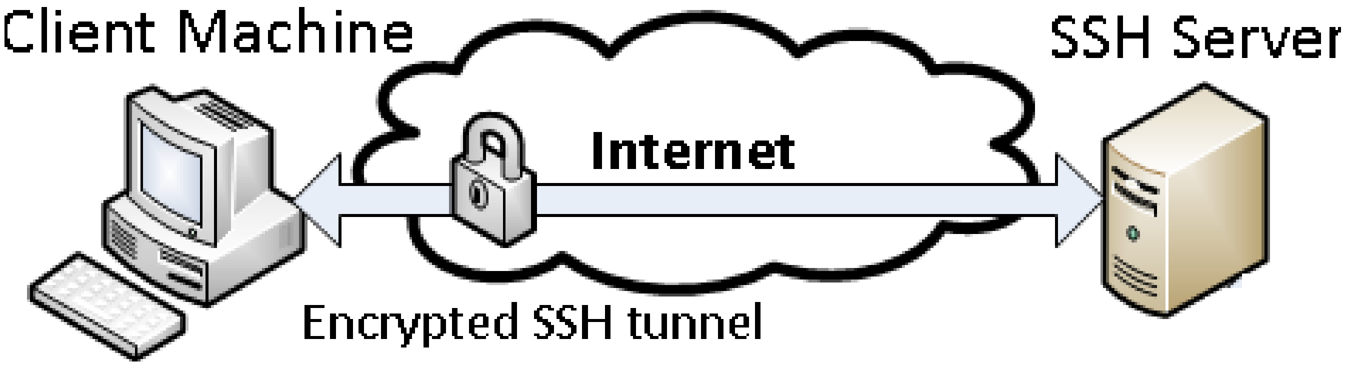
6: Authentication of client cont…

```
ssh  -l   user_name   server_IP
```


SSH Client
Secure SSH Host Server
SSH Client Secret Key
SSH Client Public Key
Client Initiates SSH Connection
Hello There
Generate random message
Encrypt message with secret key
$&#81 z3u%[>
Decrypted the encrypted random message using the client's public key
Authentication Passed
If random messages match, send a affirmative response
Agree on Session Key
Encrypted Data (symmetrical algorithm)
Exchange Data using the Session Key and a symmetrical crytographic algorithm

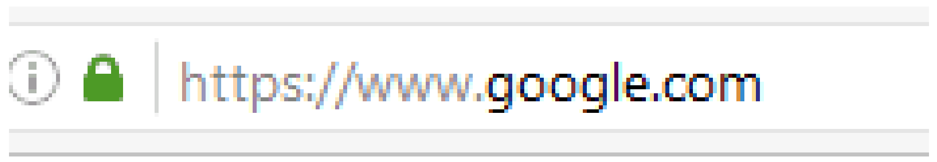
7: Transmission of data

- After the server successfully authenticate the client, a tunneled connection is established then SSH provides encrypted file transfer between the client and the server.



SSL protocol

- Secure socket layer (SSL) is public key authentication.
- One common use of SSL is to secure Web HTTP communication between a browser and a webserver, by creating an encrypted link between the two.
- When you request a URL, the server sends your browser a copy of its SSL certificate. The browser verifies that it's authentic, and the server then sends back a signed acknowledgment. Upon arrival, both start an SSL encrypted session and can share data safely.


https://www.google.com