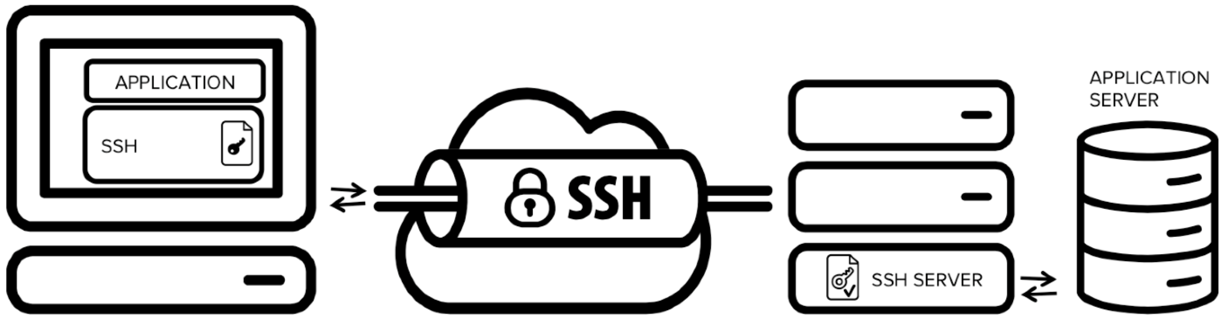
SSH

SECURE SHELL

SSH is a remote administration protocol that allows users to control and modify their remote servers over the Internet.

Uses cryptographic techniques to ensure that all communication **to** and **from** the remote server happens in an encrypted manner.



MECHANISM PROVIDED

Authenticating a remote user,

Transferring inputs from the client to the host and

Relaying the output back to the client.

OPERATING SYSTEM

MacOS or Linux - terminal

Windows - SSH clients like **PUTTY**

SSH commands consists of 3 distinct parts:

```
ssh {user}@{host}
```

ssh – open an encrypted Secure Shell Connection

{user} – refers account you want to access {host} – refers computer you want to access

Eg. 234.34.222.4 or www.mb123.com

After entering this command,it will directed to enter the password.

If password is correct you will be greeted with remote terminal window.

Host-refers to remote server

Client-Computer you are using

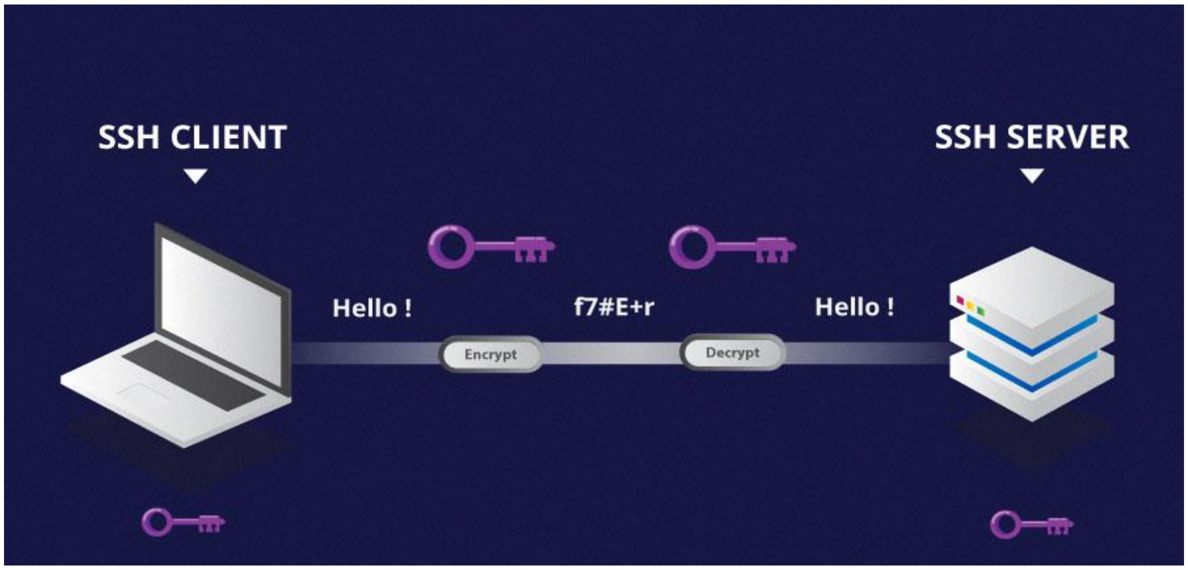
Technologies used by SSH

1. Symmetrical encryption
2. Asymmetrical encryption
3. Hashing.

<u>Symmetric Encryption</u>

Secret key is used for both encryption and decryption.

Symmetrical encryption is often called as **Shared key** or **Shared secret encryption**



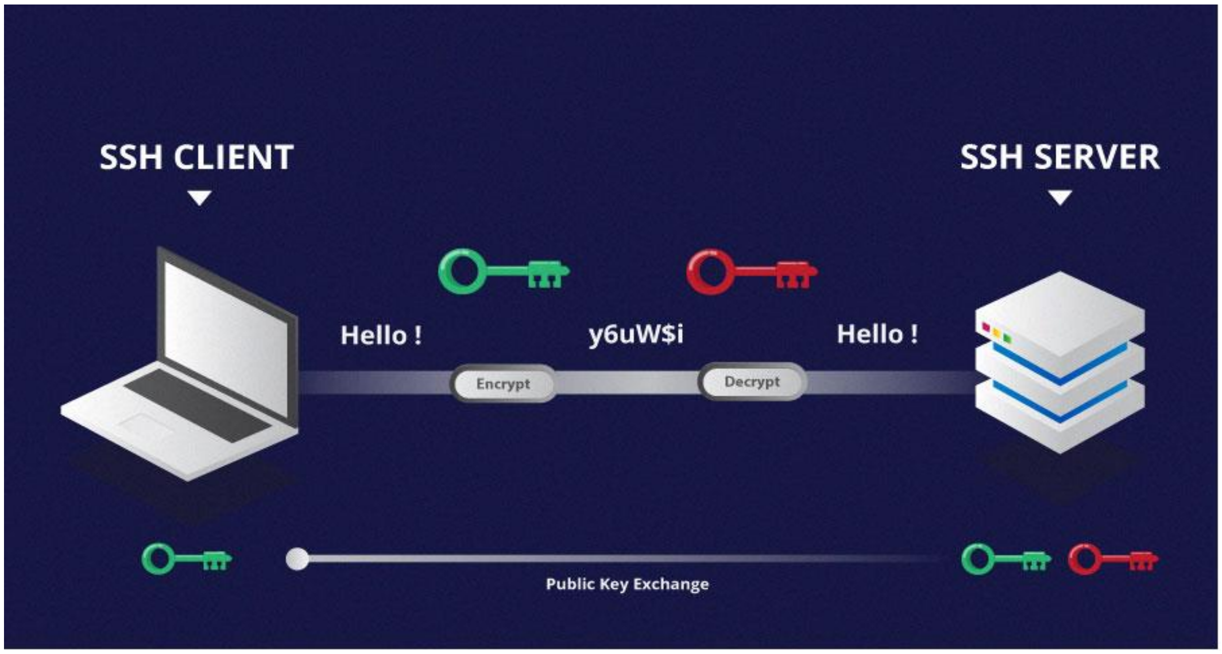
Used to encrypt the entire SSH Session. The process of creating a symmetric key is carried out by key exchange algorithm.

<u>**Asymmetric Encryption**</u>

Use two keys for encryption and decryption.

Public key and private key. Together form a public-private key pair.

The public key, as the name suggest is openly distributed and shared with all parties.



RELATION BETWEEN PRIVATE AND PUBLIC KEY

A message that is encrypted by a machine's public key, can only be decrypted by the same machine's private key. This **one-way relation** means that the public key cannot decrypt its own messages, nor can it decrypt anything encrypted by the private key.

The private key must remain private i.e. for the connection to be secured, no third party must ever know it.

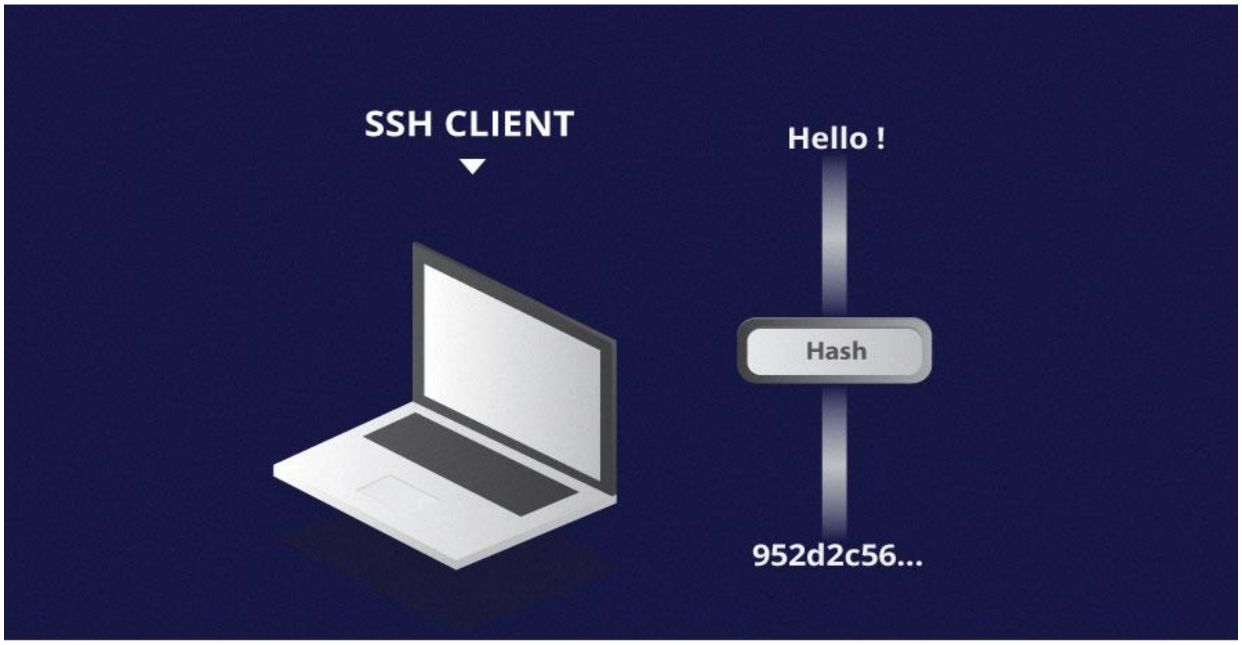
ACTIVATING SSH

Once a secured symmetric communication has been established, the server uses the clients public key to generate and transmit it to the client for authentication. If the client can successfully decrypt the message, it means that it holds the private key required for the connection. The SSH session then begins.

<u>Hashing</u>

One-way hashing is another form of cryptography used in Secure Shell Connections. One-way-hash functions differ from the above two forms of encryption in the sense that they are never meant to be decrypted.

It is easy to generate a cryptographic hash from a given input,but impossible to generate the input from the hash.



SSH uses hashes to verify the authenticity of messages. This is done using HMACs, or Hash-based Message Authentication Codes.

This means that if a client holds the correct input, they can generate the crypto-graphic hash and compare its value to verify whether they possess the correct input.

Working

SSH operates **on TCP port 22** by default. The host listens on port 22 for incoming connections. It organizes the secure connection by authenticating the client and opening the correct shell environment if the verification is successful.

SSH TUNNELING

SSH tunneling (also referred to as **SSH port forwarding**) is simply routing local network traffic through SSH to remote hosts. This implies that all your connections are secured using encryption.

It provides an easy way of setting up a basic VPN (Virtual Private Network), useful for connecting to private networks over unsecure public networks like the Internet.

For example, entire country-wide **ATM** networks run using tunneling for security.

Three types of SSH port forwarding:

Local

Remote and

Dynamic port forwarding.

Let assume the following setup

Local Host: ***192.168.43.31***

Remote Host: hostname ***server1.example.com***

Usually, you can securely connect to a remote server using SSH as follows

$ ssh admin@server1.example.com

Local port forwarding

This type of port forwarding lets you connect from your local computer to a remote server.

You can forward a local port (e.g 8080) which you can then use to access the application locally as follows. The -L flag defines the port forwarded to the remote host and remote port.

$ ssh admin@server1.example.com -L 8080: server1.example.com:3000

Now, on your local machine, open a browser, instead of accessing the remote application using the address server1.example.com:3000, you can simply use localhost:8080 or 192.168.43.31:8080.

Remote port forwarding

Remote port forwarding allows you to connect from your remote machine to the local computer.

By default, SSH does not permit remote port forwarding. You can enable this using the GatewayPorts directive in you SSHD main configuration file /etc/ssh/sshd_config on the remote host.

```
$ ssh -f -N admin@server1.example.com -R 5000:localhost:3000
```

The -f switch instructs ssh to run in the background.

Dynamic port forwarding

This is the third type of port forwarding. Unlike local and remote port forwarding which allow communication with a single port, it makes possible, a full range of TCP communications across a range of ports. Dynamic port forwarding sets up your machine as a SOCKS proxy server which listens on port 1080, by default.

```
$ ssh -f -N -D 1080 admin@server1.example.com
```

the -N flag means do not execute a remote command, you will not get a shell in this case.