**Multiple ways to create Shell session with remote Device**

**1. Basic SSH Connection**

***ssh user@hostname\_or\_ip***

Connects to a remote machine using SSH.

ssh john@192.168.1.100

This command will connect you to the machine with the IP address 192.168.1.100 as the user john.

**2. Specify a Different Port**

***ssh -p port\_number user@hostname\_or\_ip***

Connects to a remote machine using a specific port (other than the default port 22).

Example:

ssh -p 2222 john@192.168.1.100

This connects to the machine using port 2222.

**3. Execute a Command on the Remote Server**

***ssh user@hostname\_or\_ip "command"***

Runs a single command on the remote server and returns the output.

Example:

ssh john@192.168.1.100 "ls -l /var/www"

This lists the contents of the /var/www directory on the remote server.

**4. Use a Specific Private Key File**

***ssh -i /path/to/private\_key user@hostname\_or\_ip***

Specifies a private key file for SSH authentication.

Example:

ssh -i ~/.ssh/id\_rsa john@192.168.1.100

This connects to the server using the id\_rsa private key.

**5. Enable X11 Forwarding**

***ssh -X user@hostname\_or\_ip***

Enables X11 forwarding, allowing you to run GUI applications on the remote server and display them locally.

Example:

ssh -X john@192.168.1.100

After connecting, you can run a GUI application like eog (image viewer), and it will display on your local machine.

**6. Use a Specific SSH Config File**

***ssh -F /path/to/config user@hostname\_or\_ip***

Uses a specific SSH configuration file instead of the default ~/.ssh/config.

Example:

ssh -F /home/john/.ssh/my\_custom\_config john@192.168.1.100

This uses my\_custom\_config for the SSH connection.

**7. Force SSH to Use IPv4 or IPv6**

Force IPv4:

***ssh -4 user@hostname\_or\_ip***

Force IPv6:

***ssh -6 user@hostname\_or\_ip***

Example:

ssh -4 john@192.168.1.100

This forces the connection to use IPv4.

**8. SSH with Agent Forwarding**

***ssh -A user@hostname\_or\_ip***

Enables SSH agent forwarding, allowing you to use your local SSH keys on the remote server without copying them.

Example:

ssh -A john@192.168.1.100

After connecting, any subsequent SSH connections from that server will use your local keys.

**9. SSH in Quiet Mode**

***ssh -q user@hostname\_or\_ip***

Suppresses warning messages and only shows the essential output.

Example:

ssh -q john@192.168.1.100

Useful for scripts where you don’t want unnecessary output.

**10. SSH in Verbose Mode**

***ssh -v user@hostname\_or\_ip***

Enables verbose mode, showing detailed information about the SSH connection process.

Example:

ssh -v john@192.168.1.100

You can also use -vv or -vvv for even more verbosity.

**11. SSH with Compression**

***ssh -C user@hostname\_or\_ip***

Enables compression, which can speed up the transfer of data over slow networks.

Example:

ssh -C john@192.168.1.100

Useful when working with slow or high-latency connections.

**12. SSH with a Different Username**

***ssh hostname\_or\_ip -l different\_user***

Connects to the server with a different username than your current local user.

Example:

ssh 192.168.1.100 -l admin

This connects as admin instead of your default local username.

**13. SSH with a Jump Host (ProxyCommand)**

***ssh -J jump\_user@jump\_host destination\_user@destination\_host***

Connects to a destination server through a jump host.

Example:

ssh -J john@jump.server.com mary@final.server.com

This first connects to jump.server.com as john, then to final.server.com as many.

**14. SSH Tunneling (Local Port Forwarding)**

***ssh -L local\_port:destination\_host:destination\_port user@hostname\_or\_ip***

Forwards a local port to a port on the remote server.

Example:

ssh -L 8080:localhost:80 john@192.168.1.100

This forwards port 8080 on your local machine to port 80 on the remote machine.

**15. SSH Reverse Tunneling (Remote Port Forwarding)**

***ssh -R remote\_port:localhost:local\_port user@hostname\_or\_ip***

Forwards a port on the remote server to a port on your local machine.

Example:

ssh -R 9090:localhost:3000 john@192.168.1.100

This forwards port 9090 on the remote machine to port 3000 on your local machine.

**16. SSH with a Timeout**

***ssh -o ConnectTimeout=seconds user@hostname\_or\_ip***

Sets a timeout for the SSH connection attempt.

Example:

ssh -o ConnectTimeout=10 john@192.168.1.100

This will timeout after 10 seconds if the connection cannot be established.

**17. Copying Files Using SSH with SCP**

***scp /local/path/to/file user@hostname\_or\_ip:/remote/path/to/target/location***

Copies a file from the local machine to the remote server.

Example:

scp /home/john/file.txt john@192.168.1.100:/home/john/

**18. Copying Files Using SSH with SFTP**

***sftp user@hostname\_or\_ip***

Opens an SFTP session to transfer files securely.

Example:

sftp john@192.168.1.100

Once connected, use commands like get, put, ls, and cd to transfer files.

**19. Create an SSH Key Pair**

***ssh-keygen -t rsa -b 4096***

Generates a new SSH key pair using RSA encryption.

Example:

ssh-keygen -t rsa -b 4096 -C "your\_email@example.com"

This creates a key pair with a comment (typically your email).

**20. Copy SSH Public Key to Remote Server**

***ssh-copy-id user@hostname\_or\_ip***

Copies your SSH public key to the remote server's authorized\_keys file.

Example:

ssh-copy-id john@192.168.1.100

After this, you can log in without a password using your SSH key.

These commands should cover most common use cases when working with SSH, making it easier to manage remote servers and transfer files securely.