SSH Essentials Cheatsheet

A comprehensive guide to SSH (Secure Shell) commands, configuration, and usage scenarios, covering basic connections, key management, port forwarding, and more. This cheat sheet provides a quick reference for both beginners and experienced users.

Basic SSH Usage

Connecting to a Remote Server

`ssh user@host`	Connects to the specified host as the given user. **Example:** `ssh john.doe@example.com`
`ssh -p port user@host`	Connects to the host on a specific port. **Example:** `ssh -p 2222 john.doe@example.com`
`ssh -i private_key user@host`	Connects using a specific private key file. **Example:** `ssh -i ~/.ssh/id_rsa john.doe@example.com`
`ssh -v user@host`	Verbose mode, useful for debugging connection issues.
`ssh -T user@host command`	Execute a single command on the remote host without opening a shell. **Example:** `ssh -T john.doe@example.com uptime`
`ssh -q user@host`	Quiet mode, suppresses most warning and diagnostic messages.

SSH Configuration File (~/.ssh/config)

The `~/.ssh/config` file allows you to define settings for SSH connections.

Example:

```
Host example
  HostName example.com
  User john.doe
  Port 2222
  IdentityFile ~/.ssh/id_rsa
```

Now you can simply use `ssh example` to connect.

Common SSH Options

`HostName`	The actual hostname or IP address of the server.
`User`	The username to use for the connection.
`Port`	The port number to connect to (default is 22).
`IdentityFile`	Specifies the private key file for authentication.
`StrictHostKeyChecking`	Controls how SSH handles unknown host keys (`yes`, `no`, `ask`).
`ProxyCommand`	Command to use to connect to the server.

Key Management

Generating SSH Keys

`ssh-keygen`	Generates a new SSH key pair (private and public key). **Example:** `ssh-keygen -t rsa -b 4096 -C "your_email@example.com"`
`ssh-keygen -t ed25519`	Generates a new Ed25519 SSH key pair (private and public key). **Example:** `ssh-keygen -t ed25519 -C "your_email@example.com"`
`ssh-keygen -t rsa -b 4096`	Generates a new RSA SSH key pair with 4096 bits.
`ssh-keygen -f keyfile`	Creates a key without prompting.

Copying Keys to Remote Servers

`ssh-copy-id user@host`	Copies your public key to the remote server's `~/.ssh/authorized_keys` file. **Example:** `ssh-copy-id john.doe@example.com`
`cat ~/.ssh/id_rsa.pub | ssh user@host 'mkdir -p ~/.ssh && cat >> ~/.ssh/authorized_keys'`	Alternative method to copy the public key manually.
`pbcopy < ~/.ssh/id_rsa.pub`	Copy the public key to clipboard.

Key Security

Always protect your private key. Ensure it has appropriate permissions (e.g., `chmod 600 ~/.ssh/id_rsa`). Never share your private key with anyone.

Use a strong passphrase when generating your SSH key. This adds an extra layer of security.

Port Forwarding

Local Port Forwarding

`ssh -L local_port:host:remote_port user@ssh_server`	Forwards traffic from `local_port` on your machine to `remote_port` on `host` as seen from `ssh_server`. **Example:** `ssh -L 8080:localhost:80 john.doe@example.com` (Access the web server on example.com via `localhost:8080` on your machine).
`ssh -L 8080:192.168.1.10:80 john.doe@example.com`	Access the web server on 192.168.1.10 on your machine.

Remote Port Forwarding

`ssh -R remote_port:host:local_port user@ssh_server`	Forwards traffic from `remote_port` on `ssh_server` to `local_port` on `host` as seen from your machine. **Example:** `ssh -R 9000:localhost:3000 john.doe@example.com` (Someone connecting to `example.com:9000` will be forwarded to your machine's port 3000).

Dynamic Port Forwarding (SOCKS Proxy)

`ssh -D local_port user@ssh_server`	Creates a SOCKS proxy on `local_port` on your machine, routing all traffic through `ssh_server`. **Example:** `ssh -D 1080 john.doe@example.com` (Configure your browser to use `localhost:1080` as a SOCKS proxy).
`ssh -N -D 1080 user@ssh_server`	Background the process and don't execute a remote command.

Common Options

- `-N`: Do not execute a remote command. Useful for port forwarding only.
- `-f`: Requests ssh to go to background after authentication.

Advanced SSH Usage

Executing Commands Remotely

`ssh user@host 'command'`	Executes a single command on the remote host. **Example:** `ssh john.doe@example.com 'df -h'` (Shows disk space usage on the remote server).

`ssh user@host << EOF command1 command2 EOF`	Executes multiple commands using a 'here document'.
	
	Example:
	`ssh john.doe@example.com << EOF`
	`mkdir test_dir`
	`cd test_dir`
	`pwd`
	`EOF`

`ssh user@host bash -s < script.sh`	Execute a shell script. **Example:** `ssh john.doe@example.com bash -s < script.sh`

SCP (Secure Copy)

`scp file user@host:destination`	Copies a file to a remote host. **Example:** `scp myfile.txt john.doe@example.com:/home/john.doe/`
`scp user@host:file destination`	Copies a file from a remote host. **Example:** `scp john.doe@example.com:/home/john.doe/myfile.txt .`
`scp -r directory user@host:destination`	Copies a directory recursively to a remote host. **Example:** `scp -r mydirectory john.doe@example.com:/home/john.doe/`
`scp -P port user@host:file destination`	Copies a file from a remote host on a specific port. **Example:** `scp -P 2222 john.doe@example.com:/home/john.doe/myfile.txt .`

SSH Agent Forwarding

`ssh -A user@host`	Enables agent forwarding, allowing you to use your local SSH keys on the remote server for further connections. Use with caution, as it can pose a security risk. **Note:** Ensure `ForwardAgent yes` is in your `~/.ssh/config` or the server's `/etc/ssh/ssh_config`.
`ssh -o ForwardAgent=yes user@host`	Enables agent forwarding, allowing you to use your local SSH keys on the remote server for further connections. Use with caution, as it can pose a security risk.

Mosh (Mobile Shell)

Mosh is a mobile shell that provides a more robust and responsive connection, especially over unreliable networks. It tolerates intermittent connectivity and IP address changes.

Basic Usage:

1. Install mosh on both your local machine and the remote server.
2. `mosh user@host`