SSH Cheatsheet

By Dejan Panovski • Updated on Jan 26, 2026 • Download PDF

Quick reference for SSH commands and configuration

SSH (Secure Shell) is a protocol for securely connecting to remote systems. This cheatsheet covers common SSH commands for connecting, file transfer, tunneling, and key management.

Basic Connection

Connect to remote servers.

`ssh user@host`	Connect to host
`ssh host`	Connect with current username
`ssh -p 2222 user@host`	Connect on custom port
`ssh user@host command`	Run command on remote host
`ssh -v user@host`	Verbose mode (debug)
`ssh -q user@host`	Quiet mode

SSH Keys

Generate and manage SSH keys.

`ssh-keygen`	Generate SSH key pair
`ssh-keygen -t ed25519`	Generate Ed25519 key
`ssh-keygen -t rsa -b 4096`	Generate 4096-bit RSA key
`ssh-keygen -p -f ~/.ssh/id_ed25519`	Change key passphrase
`ssh-keygen -y -f ~/.ssh/id_ed25519`	Show public key
`ssh-keygen -R hostname`	Remove host from known_hosts

Copy SSH Key

Set up passwordless authentication.

`ssh-copy-id user@host`	Copy key to remote host
`ssh-copy-id -i ~/.ssh/key.pub user@host`	Copy specific key
`ssh-copy-id -p 2222 user@host`	Copy key on custom port

SSH Agent

Manage SSH keys in memory.

`eval "$(ssh-agent -s)"`	Start SSH agent
`ssh-add`	Add default key to agent
`ssh-add ~/.ssh/id_ed25519`	Add specific key
`ssh-add -l`	List keys in agent
`ssh-add -d ~/.ssh/id_ed25519`	Remove key from agent
`ssh-add -D`	Remove all keys

SCP (Secure Copy)

Copy files over SSH.

`scp file user@host:/path`	Copy file to remote
`scp user@host:/path/file .`	Copy file from remote
`scp -r dir user@host:/path`	Copy directory recursively
`scp -P 2222 file user@host:/path`	Copy on custom port
`scp -C file user@host:/path`	Copy with compression
`scp -p file user@host:/path`	Preserve timestamps

SFTP

Interactive file transfer.

`sftp user@host`	Connect to host
`sftp -P 2222 user@host`	Connect on custom port
`get file`	Download file (in sftp)
`put file`	Upload file (in sftp)
`ls, cd, pwd`	Navigate remote (in sftp)
`lls, lcd, lpwd`	Navigate local (in sftp)

SSH Tunneling

Create secure tunnels.

`ssh -L 8080:localhost:80 user@host`	Local port forwarding
`ssh -R 8080:localhost:80 user@host`	Remote port forwarding
`ssh -D 1080 user@host`	SOCKS proxy (dynamic)
`ssh -N -L 8080:localhost:80 user@host`	Tunnel only (no shell)
`ssh -f -N -L 8080:localhost:80 user@host`	Background tunnel

SSH Config File

Simplify connections with config.

`~/.ssh/config`	User config file
`Host myserver`	Define host alias
`HostName 192.168.1.100`	Server address
`User admin`	Username
`Port 2222`	Custom port
`IdentityFile ~/.ssh/mykey`	Private key path

Connection Options

Common SSH options.

`-p port`	Custom port
`-i keyfile`	Identity file (private key)
`-o option=value`	Set config option
`-F configfile`	Custom config file
`-J jumphost`	Jump through host (ProxyJump)
`-X`	Enable X11 forwarding
`-A`	Enable agent forwarding

Security Options

Harden SSH connections.

`-o StrictHostKeyChecking=yes`	Strict host key check
`-o UserKnownHostsFile=/dev/null`	Ignore known hosts
`-o PasswordAuthentication=no`	Disable password auth
`-o PubkeyAuthentication=yes`	Enable key auth
`-o ConnectTimeout=10`	Connection timeout

Multiplexing

Reuse SSH connections.

`ControlMaster auto`	Enable multiplexing
`ControlPath ~/.ssh/sockets/%r@%h-%p`	Socket path
`ControlPersist 600`	Keep connection for 10 min
`ssh -O check user@host`	Check connection status
`ssh -O exit user@host`	Close master connection

Common Patterns

Frequently used combinations.

`ssh -t user@host 'sudo command'`	Run sudo command
`ssh user@host 'cat file' > local`	Copy output to local
`tar czf - dir | ssh user@host 'tar xzf -'`	Copy dir via tar
`ssh -J jump user@dest`	Connect via jump host
`ssh user@host -L 3306:localhost:3306`	MySQL tunnel