April 13, 2026 · 8 min read · By Yiğit Konur

using claude code's new native ssh remote on a mac mini / darwin

ai software-engineering

so you got yourself a mac mini. maybe it's sitting headless under your desk, maybe it's in a closet running 24/7 as your dev server. you want to connect to it with claude code's ssh remote feature from your macbook, your ipad, whatever.

you type in the host, hit connect, and get slapped with:

```
Unsupported remote platform: darwin. Only Linux hosts are supported for SSH connection
```

yeah. claude code ssh remote only works with linux hosts. your mac mini runs macos (darwin). dead end, right?

nah.

the workaround: run linux inside your mac

here's the move: you spin up a lightweight linux vm on your mac mini using orbstack (https://orbstack.dev). orbstack is basically docker desktop but actually good -- it also runs full linux machines with near-native performance on apple silicon.

the key insight: **orbstack automatically mounts your entire macos filesystem into every linux vm**. so your code at `/Users/yourname/dev` on macos? it's right there at `/Users/yourname/dev` inside the vm too. same files. no syncing. no copying. just works via virtiofs.

so instead of ssh-ing into macos (which claude code rejects), you ssh into a linux vm that has full access to all your mac's files. claude code sees linux, everybody's happy.

-

the setup

1. install orbstack on the mac mini

```bash
brew install orbstack
```

or grab it from orbstack.dev (https://orbstack.dev). open it once to finish setup.

2. create an ubuntu vm

```bash
orbctl create ubuntu:noble dev-sandbox
```

this gives you ubuntu 24.04 arm64. takes like 10 seconds. it auto-creates a user matching your macos username.

3. install ssh server in the vm

```bash
orb -m dev-sandbox -u root bash -c '
  apt-get update && apt-get install -y openssh-server &&
  systemctl enable ssh &&
  systemctl start ssh
'
```

4. set a password (for initial key copy)

```bash
orb -m dev-sandbox -u root bash -c 'echo "yourusername:yourpassword" | chpasswd'
```

5. get the vm's ip

```bash
orbctl info dev-sandbox
```

look for the `IPv4` line. something like `192.168.139.x`. this is on orbstack's internal network, only reachable from the mac mini itself.

6. copy your ssh key from the mac mini

```bash
ssh-copy-id yourusername@192.168.139.x
```

7. fix the home directory

orbstack maps your macos home to `/Users/yourname` inside the vm, but ssh defaults to `/home/yourname`. fix it so you land in the right place:

```bash
orb -m dev-sandbox -u root usermod -d /Users/yourname yourusername
```

make sure your ssh keys are in the right spot:

```bash
orb -m dev-sandbox mkdir -p /Users/yourname/.ssh
orb -m dev-sandbox -u root bash -c 'cp /home/yourname/.ssh/authorized_keys /Users/your
```

8. (optional) passwordless sudo

```bash
orb -m dev-sandbox -u root bash -c 'echo "yourusername ALL=(ALL) NOPASSWD:ALL" > /etc/
```

9. create a convenience symlink

```bash
orb -m dev-sandbox ln -sf /Users/yourname/dev ~/dev
```

connecting from your macbook

your macbook can't reach the vm directly -- its ip is on orbstack's internal network inside the mac mini. so you proxy through the mac mini.

add this to `~/.ssh/config` on your **macbook**:

```ini
Host dev
  HostName 192.168.139.x
  Port 22
  User yourusername
  ProxyCommand ssh -W %h:%p yourusername@<mac-mini-lan-ip>
  IdentityFile ~/.ssh/id_ed25519
  StrictHostKeyChecking no
  UserKnownHostsFile /dev/null
```

replace `192.168.139.x` with the vm ip from step 5, and `<mac-mini-lan-ip>` with your mac mini's actual lan ip (like `192.168.1.200`).

also copy your macbook's ssh key into the vm:

```bash
ssh-copy-id -o "ProxyCommand ssh -W %h:%p yourusername@<mac-mini-lan-ip>" yourusername
```

now `ssh dev` from your macbook drops you straight into the linux vm with access to all your mac mini's files.

important: use `ProxyCommand` not `ProxyJump`. some tools (including claude code) don't support `ProxyJump` yet.

-

connect claude code

now in claude code, set up an ssh remote connection to host `dev` (or whatever you named it in your ssh config). it connects through your mac mini into the linux vm, sees ubuntu, and everything works.

your code is right there at `/Users/yourname/dev`. edits from claude code write directly to the mac's filesystem. no lag, no sync issues.

what you end up with

- claude code thinks it's talking to a linux box (because it is)
- all your mac mini's files are accessible at their original paths
- full cpu and ram (orbstack shares resources dynamically, no fixed allocation)
- near-native performance on apple silicon
- the vm uses like 900mb of disk

things to know

- **your files are safe.** `/Users` is mounted from macos via virtiofs. deleting the vm doesn't touch your files. they live on the mac's disk.
- **orbstack needs to be running.** if the mac mini reboots, orbstack starts automatically, but you may need to verify the vm comes back up. run `orbctl start dev-sandbox` to be sure.
- **the vm ip can change.** if you recreate the vm, update your ssh config with the new ip. or use orbstack's dns: `dev-sandbox.orb.local` might work depending on your setup.
- **linux tools work.** need docker inside the vm? install it. need specific linux packages for your dev workflow? go for it. it's a full ubuntu system.

-

tldr

mac mini + orbstack linux vm = claude code ssh remote actually working on your apple silicon mac. the vm sees all your macos files through virtiofs, claude code sees linux, problem solved.

SOURCES

OrbStack (https://orbstack.dev)

Anthropic Claude Code docs (https://docs.anthropic.com/en/docs/claude-code)

-