

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

Read this story for free via [Prateek Jain's Friend Link](#). [Upgrade](#) to access the best of Medium.

Member-only story

Secure Server Access with Teleport

Deploy Teleport with Docker to secure Linux, Windows, Kubernetes, and cloud resources.



Prateek Jain

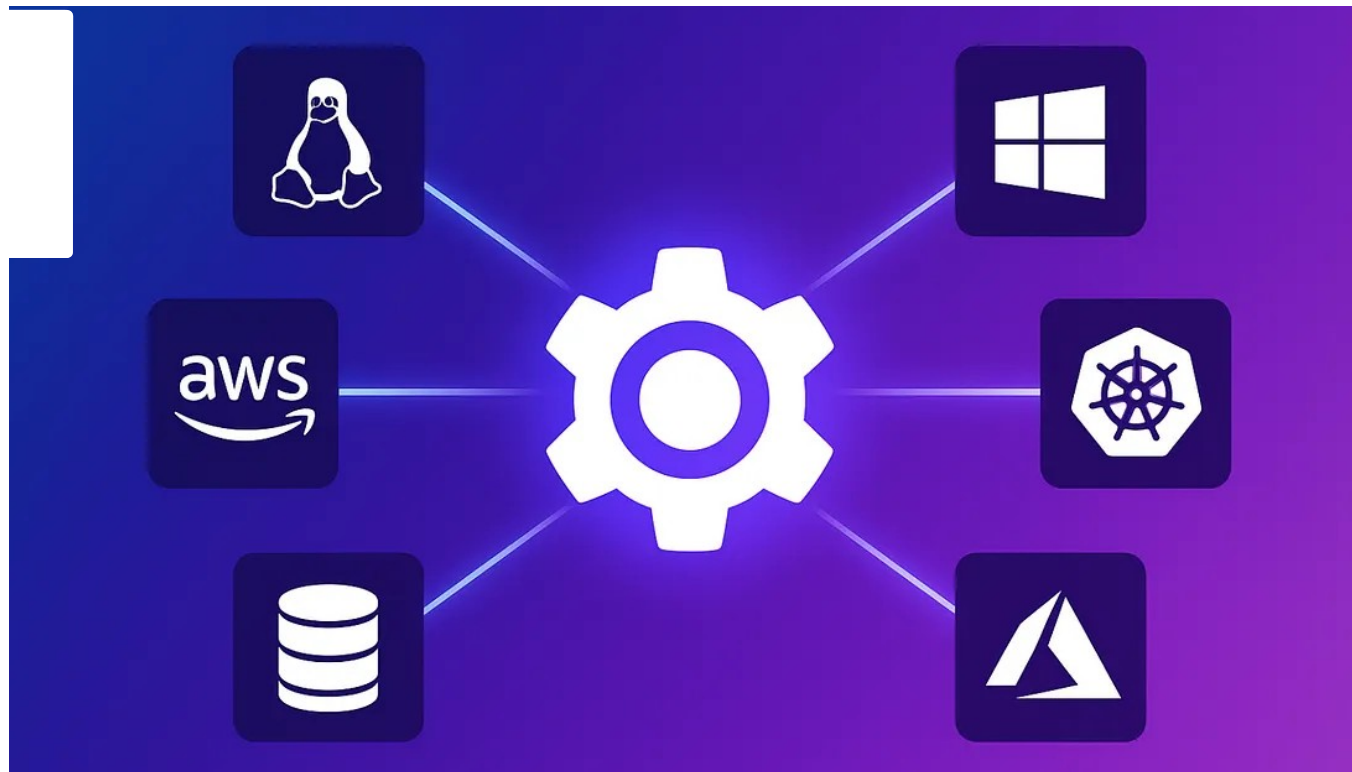
Follow

8 min read · 6 hours ago



All your favorite parts of Medium are now in one sidebar for easy access.

Okay, got it



As a DevOps, Cloud, or Sysadmin professional, one of the most common tasks we deal with is **managing access to Linux virtual machines**. In my day-to-day role, I often spin up EC2 instances for my team and then need to share access with other engineers or developers.

Friend link for non-Medium members: [Secure Server Access with Teleport](#)

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

But the traditional ways of doing this come with several problems:

Sharing SSH keys: You generate SSH keys, share them via Slack/Email, and copy them to multiple servers. Not only is this insecure, but it's also hard to rotate keys later.

- **Password management:** Some teams still create local usernames and passwords for each server. This is time-consuming, error-prone, and insecure (passwords get reused or forgotten).
- **Onboarding/Offboarding:** When someone joins a project, you have to manually add their user or key on all servers. When they leave, you must remember to remove them everywhere; miss one server, and that's a security hole.
- **IP restrictions:** Restricting SSH access to certain IPs is a partial solution, but it doesn't solve the problem of key sprawl or auditing user activity.
- **Audit gaps:** During incidents, it's hard to answer questions like *who logged into this server? what did they run?*. Traditional SSH gives you no central visibility.

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

This is exactly where **Teleport** shines. Teleport provides a **single access** way for SSH, Kubernetes, databases, internal web apps, and more. It manages shared SSH keys, enforces strong authentication (MFA, SSO), and records all access for auditing.

In this blog, I'll show you how to quickly set up Teleport using **Docker**, secure it with Let's Encrypt certificates, and connect additional EC2 instances to manage via the Teleport web console.

Architecture of This Demo

We'll use two EC2 instances:

- **EC2 #1 (t3.micro)** → runs Teleport with Docker.
- **EC2 #2** → a regular VM that we'll register to Teleport as a managed node.

Final setup: You'll log into <https://teleport.your-domain.com> (Here my domain will be <https://teleport.prateek.cloud>) and authenticate with your

Teleport account, and SSH into EC2 #2 directly from the browser.

All your favorite parts of
Medium are now in one
sidebar for easy access.

[Okay, got it](#)

Step 1: Launch an EC2 Instance for Teleport

Start with an EC2 instance running Ubuntu 24.04 with the following ports open: 22 (SSH), HTTP (80) and HTTPS (443).

Update the system and install Docker:

```
sudo apt update && sudo apt upgrade -y
curl -fsSL https://get.docker.com | sh
sudo usermod -aG docker ubuntu
```

All your favorite parts of
Medium are now in one
sidebar for easy access.

[Okay, got it](#)

```
intu@ip-172-31-35-254:~$ docker version
Client: Docker Engine - Community
 Version:           28.4.0
 API version:       1.51
 Go version:        go1.24.7
 Git commit:        d8eb465
 Built:             Wed Sep  3 20:57:32 2025
 OS/Arch:           linux/amd64
 Context:           default

Server: Docker Engine - Community
 Engine:
  Version:           28.4.0
  API version:       1.51 (minimum version 1.24)
  Go version:        go1.24.7
  Git commit:        249d679
  Built:             Wed Sep  3 20:57:32 2025
  OS/Arch:           linux/amd64
  Experimental:      false
 containerd:
  Version:           1.7.27
  GitCommit:        05044ec0a9a75232cad458027ca83437aae3f4da
 runc:
  Version:           1.2.5
  GitCommit:        v1.2.5-0-g59923ef
 docker-init:
  Version:           0.19.0
  GitCommit:        de40ad0
```

Step 2: Prepare Directories

Teleport needs config and data directories:

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

```
mkdir -p ~/teleport/config ~/teleport/data
```

Step 3: Generate Teleport Configuration

We'll generate a `teleport.yaml` that runs both **Auth** and **Proxy** roles, sets the cluster name, and enables ACME (Let's Encrypt) for automatic TLS.

```
sudo docker run --rm \
  --hostname teleport.example.com \
  --entrypoint=/usr/local/bin/teleport \
  public.ecr.aws/gravitational/teleport-distroless:18.2.0 \
  configure \
  --roles=auth,proxy \
  --cluster-name=teleport.example.com \
  --public-addr=teleport.example.com:443 \
  --acme \
  --acme-email=hello@prateek.cloud \
  > ~/teleport/config/teleport.yaml
```

Explanation:

All your favorite parts of Medium are now in one sidebar for easy access.

Okay, got it

`--hostname teleport.example.com` Sets the hostname.

`--entrypoint=/usr/local/bin/teleport` Overrides the default container entrypoint.

- `--roles=auth,proxy` single-node cluster with Auth + Proxy.
- `--cluster-name` logical cluster name.
- `--public-addr` the public domain where users will connect.
- `--acme` auto-provisions certificates from Let's Encrypt.
- `--acme-email` email for Let's Encrypt registration/renewal notices.

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

```
ubuntu@ip-172-31-35-254:~$ sudo docker run --rm \
  --hostname teleport.prateek.cloud \
  --entrypoint=/usr/local/bin/teleport \
  public.ecr.aws/gravitational/teleport-distroless:18.2.0 \
  --configure \
  --roles=auth,proxy \
  --cluster-name=teleport.prateek.cloud \
  --public-addr=teleport.prateek.cloud:443 \
  --acme \
  --acme-email=hello@prateek.cloud \
  > ~/teleport/config/teleport.yaml
Unable to find image 'public.ecr.aws/gravitational/teleport-distroless:18.2.0' locally
18.2.0: Pulling from gravitational/teleport-distroless
fd4aa3667332: Pull complete
bfb59b82a9b6: Pull complete
017886f7e176: Pull complete
62de241dac5f: Pull complete
2780920e5dbf: Pull complete
7c12895b777b: Pull complete
3214acf345c0: Pull complete
5664b15f108b: Pull complete
045fc1c20da8: Pull complete
4aa0ea1413d3: Pull complete
da7816fa955e: Pull complete
ddf74a63f7d8: Pull complete
e7fa9df358f0: Pull complete
c058825cfc6: Pull complete
7faf0cfa885c: Pull complete
5b14f6c9a813: Pull complete
33ce0b1d99fc: Pull complete
f45e0372ce60: Pull complete
37ece6e2ba46: Pull complete
4e8f33650906: Pull complete
dfdc1089d130: Pull complete
Digest: sha256:7fad7ef68a77334961f00d133044d97442fbf34e7410fe601c999aca13b5e21a
Status: Downloaded newer image for public.ecr.aws/gravitational/teleport-distroless:18.2.0
ubuntu@ip-172-31-35-254:~$ ls ~/teleport/config/teleport.yaml
/home/ubuntu/teleport/config/teleport.yaml
```

Step 4: Run Teleport

Now run the container in detached mode:

All your favorite parts of Medium are now in one sidebar for easy access.

Okay, got it

```
sudo docker run -d --restart unless-stopped \
  --hostname teleport.example.com \
  --name teleport \
  -v ~/teleport/config:/etc/teleport \
  -v ~/teleport/data:/var/lib/teleport \
  -p 443:443 \
  -p 80:80 \
  public.ecr.aws/gravitational/teleport-distroless:18.2.0
```

- **443:** required forever: Teleport Proxy (TLS routing, web UI, `tsh` client).
- **80:** required for Let's Encrypt HTTP-01 ACME challenges (automatic cert issuance & renewals). Keep open while certs are being created and during renewals.

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

```
intu@ip-172-31-35-254:~$ sudo docker run -d --restart unless-stopped \
--hostname teleport.prateek.cloud \
--name teleport \
-v ~/teleport/config:/etc/teleport \
-v ~/teleport/data:/var/lib/teleport \
-p 443:443 \
-p 80:80 \
public.ecr.aws/gravitational/teleport-distroless:18.2.0
928819924e05c2a0a6cc983351e671e7d5da3764b5a71e83f611251fb51bb48d
```

Check logs:

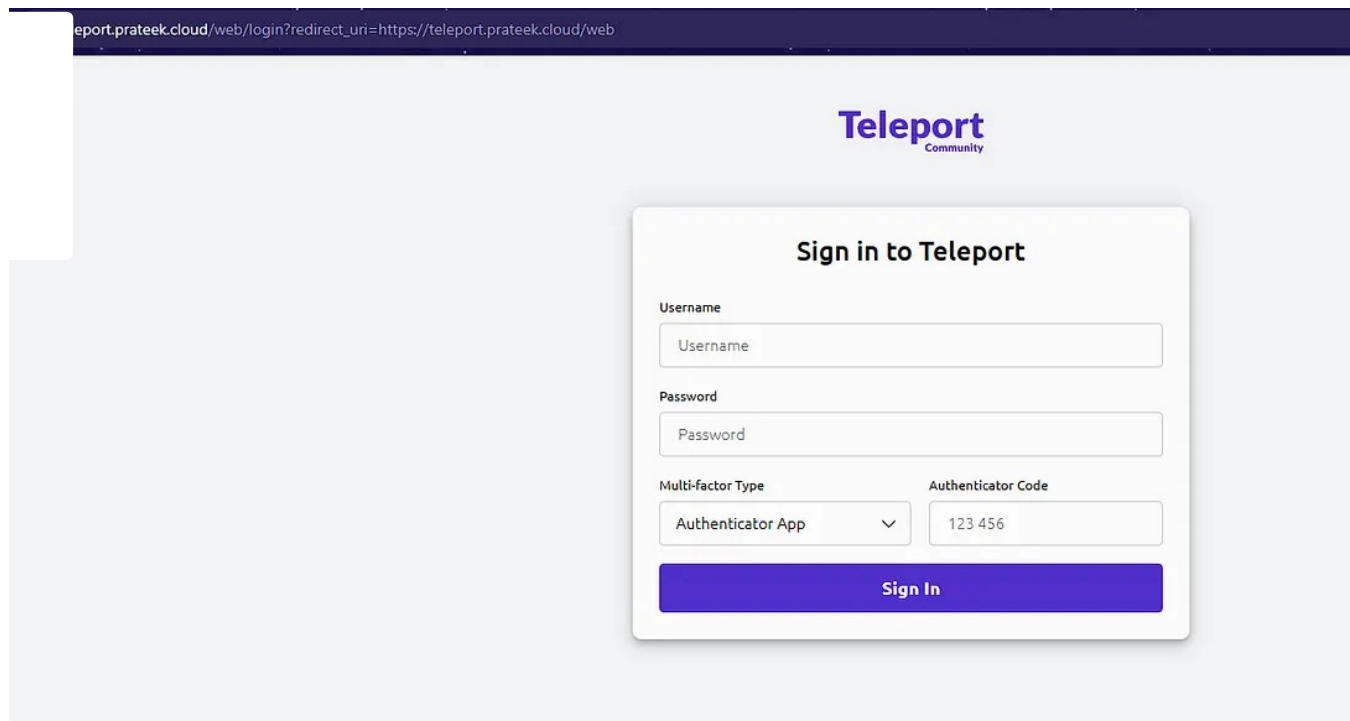
```
sudo docker logs -f teleport
```

You should see messages about **ACME certificate generation**. After a minute, open: <https://teleport.example.com>

You'll see the Teleport login page with a valid TLS certificate.

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)



teleport.prateek.cloud/web/login?redirect_uri=https://teleport.prateek.cloud/web

Teleport
Community

Sign in to Teleport

Username

Password

Multi-factor Type Authenticator Code

Authenticator App

Sign In

Step 5: Create an Admin User

Teleport ships without users. Let's add an admin account:

```
sudo docker exec -it teleport tctl users add admin --roles=editor,access --login
```

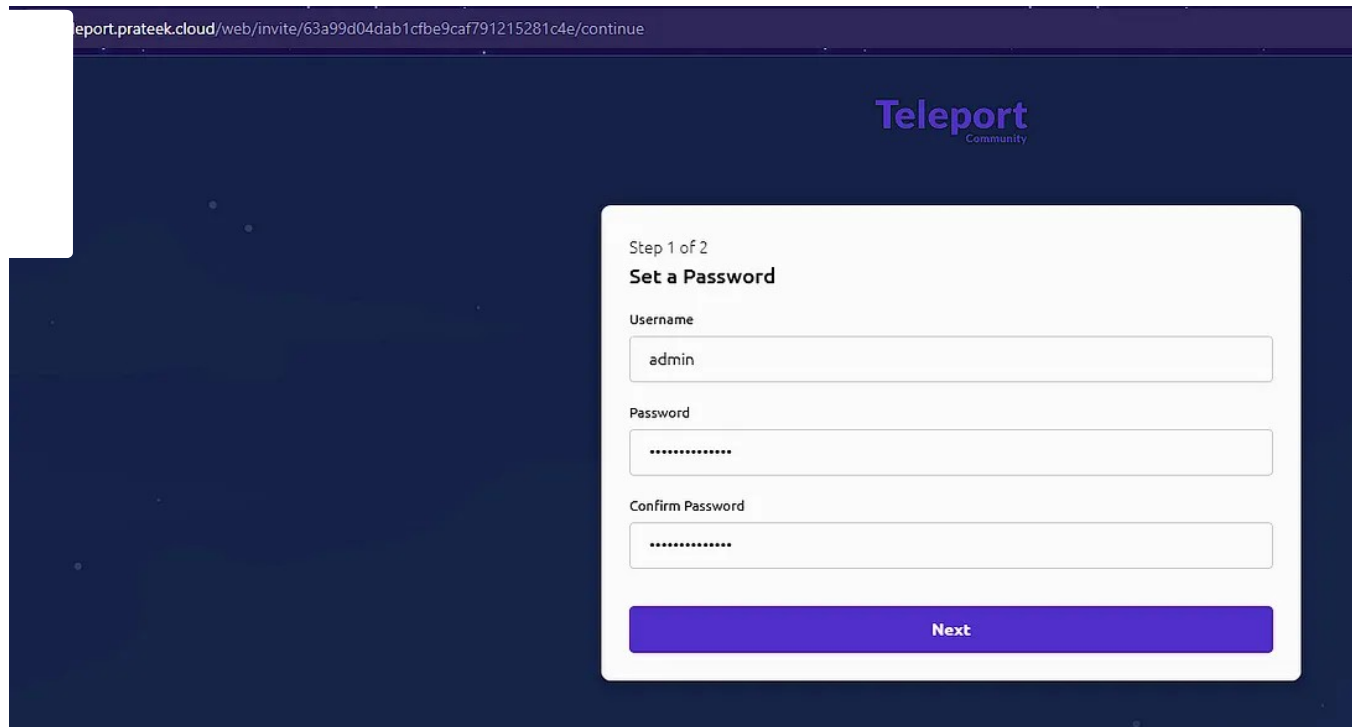
Okay, got it

This generates a **signup** URL like:

Open it in your browser → set password & MFA → you now have an admin account.

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)



The screenshot shows the Teleport Community web interface. The URL in the browser is `teleport.prateek.cloud/web/invite/63a99d04dab1cfbe9caf791215281c4e/continue`. The page has a dark blue background with the Teleport logo (a stylized 'T' in blue and purple) and the word 'Community' in small text. A white registration form is centered on the page. The form is titled 'Step 1 of 2' and 'Set a Password'. It contains three input fields: 'Username' with the value 'admin', 'Password' with masked characters '*****', and 'Confirm Password' with masked characters '*****'. A blue 'Next' button is at the bottom of the form.

Step 6: Add Another EC2 Node

Now let's register another EC2 instance with Teleport.

Before going ahead, please confirm you have an instance running with port 443 open from the teleport instance IP

All your favorite parts of Medium are now in one sidebar for easy access.

Okay, got it

¹ Log in to the Teleport Web UI → **Enroll New Resources** > **Ubuntu 18.04+**

Here you can add labels as per the requirement

The screenshot shows the 'Configure Resource' page in the Teleport Web UI. The breadcrumb navigation at the top indicates the path: 'Ubuntu 18.04+' (selected) → 'Configure Resource' → 'Set Up Access' → 'Test Connection'. The main heading is 'Configure Resource' with a subheading 'Install and configure the Teleport SSH Service'. The page is divided into two main steps. Step 1, titled 'Step 1 (Optional) Add Labels', contains a table for adding labels. The table has two columns: 'Key (required field)' and 'Value (required field)'. A single row is visible with the key 'Name' and the value 'demo-server'. Below the table is a '+ Add Another Label' button and an 'Edit Labels' button. Step 2, titled 'Step 2', instructs the user to 'Run the following command on the server you want to add'. The command is displayed in a dark box: 'sudo bash -c "\$(curl -fsSL https://teleport.prateek.cloud/scripts/0d1d/install-node.sh)"'. At the bottom, a light gray box with a clock icon contains the text: 'After running the command above, we'll automatically detect your new Teleport instance.'

Ubuntu 18.04+ — Configure Resource — 2 Set Up Access — 3 Test Connection

Configure Resource

Install and configure the Teleport SSH Service

Step 1 (Optional)
Add Labels

Key (required field)	Value (required field)
Name	demo-server

+ Add Another Label

Edit Labels

Step 2
Run the following command on the server you want to add

```
sudo bash -c "$(curl -fsSL https://teleport.prateek.cloud/scripts/0d1d/install-node.sh)"
```

⌚ After running the command above, we'll automatically detect your new Teleport instance.

All your favorite parts of Medium are now in one sidebar for easy access.

Okay, got it

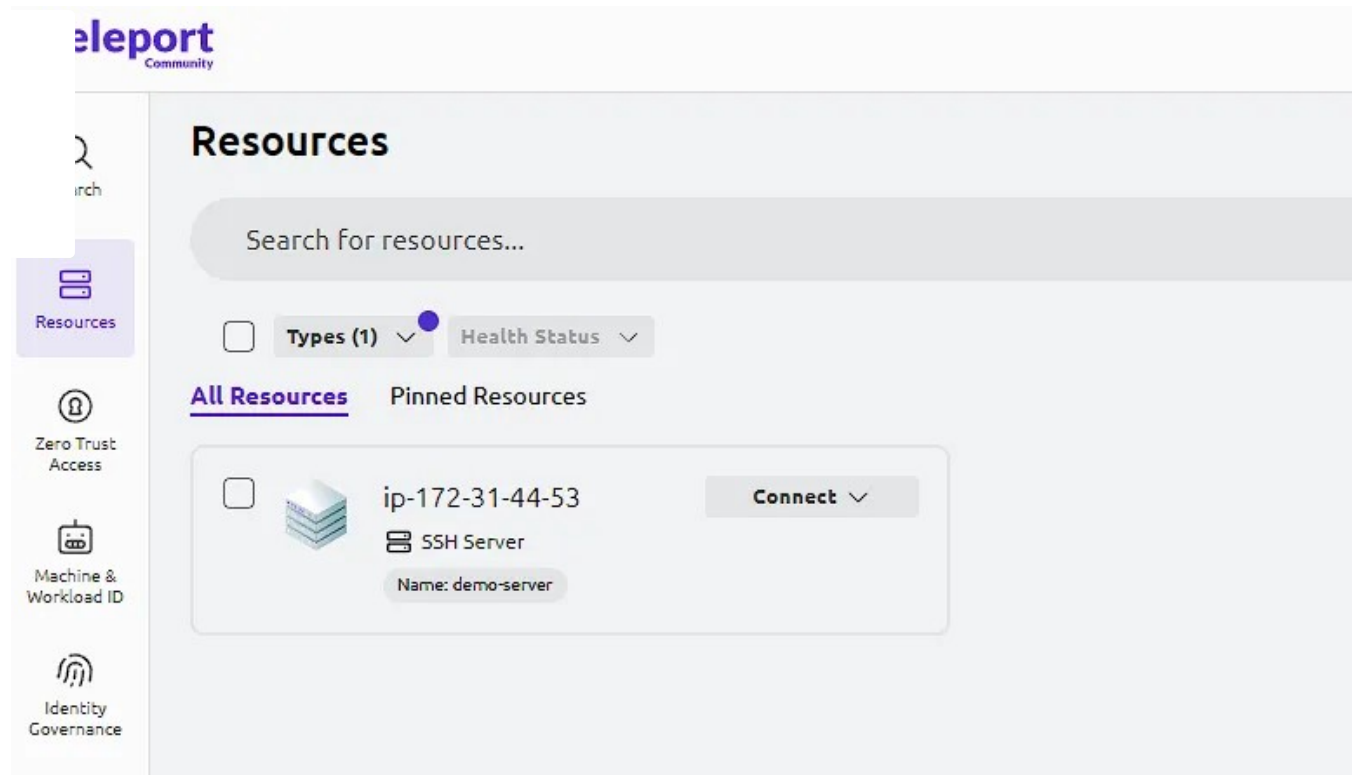
- 2 Copy the provided curl command and run it in the instance you are adding . resource. This installs the Teleport agent and registers the node.

```
Teleport has been started.  
  
View its status with 'sudo systemctl status teleport.service'  
View Teleport logs using 'sudo journalctl -u teleport.service'  
To stop Teleport, run 'sudo systemctl stop teleport.service'  
To start Teleport again if you stop it, run 'sudo systemctl start teleport.service'  
  
You can see this node connected in the Teleport web UI or 'tsh ls' with the name 'ip-172-31-44-53'  
Find more details on how to use Teleport here: https://goteleport.com/docs/
```

4. Back in the Teleport console, the new EC2 appears under **SSH Resources**.
By default, the node name is its private IP.

All your favorite parts of Medium are now in one sidebar for easy access.

Okay, got it



Step 7: Rename the Node (Optional)

To make things cleaner, edit the Teleport config on the node:

```
sudo vim /etc/teleport.yaml
```

✂+ nodename to something meaningful:

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

```
version: v3
teleport:
  nodename: demo-server
  data_dir: /var/lib/teleport
  join_params:
    token_name: 0d10133-
    method: token
  proxy_server: teleport.prateek.cloud:443
  log:
    output: stderr
    severity: INFO
```

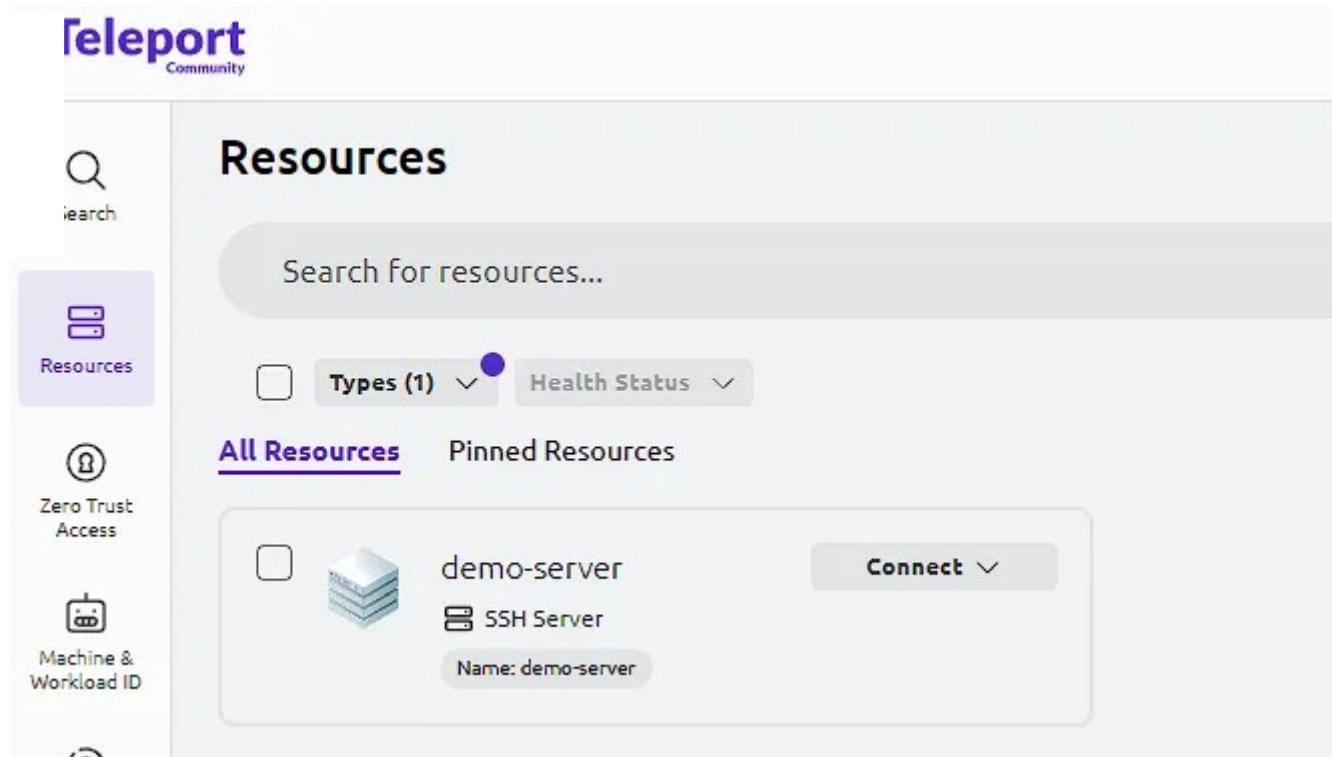
Restart the agent:

```
sudo systemctl restart teleport
```

Now the server appears in Teleport with your chosen name.

All your favorite parts of Medium are now in one sidebar for easy access.

Okay, got it



Step 8: SSH into the Node via Teleport

From the Web UI:

- Go to SSH Resources
- Click the node → Connect → select **ubuntu** as login user

- A new browser tab opens, and you're logged into the EC2 instance.

All your favorite parts of Medium are now in one sidebar for easy access.

Okay, got it



Step 9: Add More Users

You can add users via CLI or the Teleport Web UI.

From CLI:

```
sudo docker exec -it teleport tctl users add dev1 --roles=access --logins=ubuntu
```

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

This generates a signup URL you can share. Each user sets up their own sword + MFA, so you never have to share keys or passwords again.

Troubleshooting Tips

- **ACME fails:** Ensure ports 80/443 are open and your domain A record points to the EC2's public IP.
- **Teleport logs show `self-signed cert`** Double-check ACME flags and firewall.
- **Node not visible:** Verify the join script ran successfully and check `/var/lib/teleport` on the node.
- **User invite expired:** Run `tctl users add` again to generate a fresh URL.

What Resources Can You Add to Teleport?

Teleport isn't just for Linux VMs. Once you have the Teleport Proxy running, you can enroll many different resource types into your cluster:

- **Linux & Windows Servers:** SSH or RDP into servers with passwordless

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

access, audit logging, and MFA.

Kubernetes Clusters: Secure `kubectl` access with SSO and auditing, no more sharing kubeconfigs.

Databases: Access PostgreSQL, MySQL, MongoDB, Redis, and cloud-managed databases (such as Amazon RDS or Aurora) through Teleport with short-lived certificates.

- **Internal Web Apps:** Protect internal dashboards, UIs, and admin panels with Teleport's Application Access.
- **Desktops:** RDP into Windows desktops with MFA, SSO, and session recording.
- **Cloud CLIs:** Proxy AWS/GCP/Azure CLI access through Teleport to enforce authentication and auditing.

This makes Teleport a **one-stop Zero Trust access platform**, whether your infrastructure is Linux VMs, Kubernetes, RDS databases, or Windows servers.

Best Practices for Running Teleport in Production

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

When moving beyond a demo, here are some best practices to keep your teleport setup secure and reliable:

Restrict Teleport access to your company VPN

Place Teleport Proxy behind your VPN or corporate network.

- This ensures only authenticated employees can even reach the Teleport login page.

Restrict server access to Teleport Proxy IPs

- On each server (or via security groups/firewalls), only allow inbound SSH/RDP from your Teleport Proxy.
- This ensures users cannot bypass Teleport to access servers directly.

Use RBAC (Role-Based Access Control)

- Assign users the minimum roles they need (`access` , `editor` , `auditor` , etc.).
- Avoid giving `editor` Create custom roles for everyone, including developers, DB admins, etc.

Enable MFA for all users

All your favorite parts of Medium are now in one sidebar for easy access.

Okay, got it

Make MFA mandatory (TOTP or hardware keys like YubiKeys).

This blocks account takeover even if credentials are leaked.

Integrate with your SSO (Not available with OSS)

- Use Okta, Google Workspace, GitHub, or any OIDC/SAML IdP.
- This makes onboarding/offboarding automatic and centralised.

Audit & Session Recording

- Enable session recording to know exactly what commands were executed.
- Review audit logs periodically, especially after incidents.

Conclusion

Traditional SSH access is painful at scale. You have to manage SSH keys,

rotate credentials, restrict IPs, and manually track user activity, all of which is error-prone and insecure.

All your favorite parts of Medium are now in one sidebar for easy access.

Okay, got it

h Teleport, you get:

- Centralised, certificate-based access management
- Strong authentication with MFA & SSO (with paid plans)
- Easy onboarding/offboarding
- Full audit logs of who did what, when
- Browser-based SSH access (no keys to distribute)

In this guide, we deployed Teleport in Docker on an EC2 instance, secured it with Let's Encrypt TLS, created an admin user, and registered another EC2 node. Now, all SSH access is routed through Teleport with proper authentication and audit trails.

Suppose you're managing multiple servers as a DevOps, Cloud, or Sysadmin engineer. In that case, Teleport can save you hours of hassle, improve your security posture, and give you confidence in your access controls.

All your favorite parts of Medium are now in one sidebar for easy access.

[Okay, got it](#)

Start with a single node as we did, and then expand Teleport across your
it, databases, and Kubernetes clusters. Once you use passwordless,
itable access, you'll never want to go back.

You can follow me on X ([@PrateekJainDev](#)) and LinkedIn ([in/prateekjaindev](#))
for more such posts!

Thanks for reading, and keep securing! 🗝️