Proxmox

Proxmox Virtual Environment is a complete open-source platform for enterprise virtualization. With the built-in web interface you can easily manage VMs and containers, software-defined storage and networking, high-availability clustering, and multiple out-of-the-box tools using a single solution.

- Create debian-cloudinit VM template
- Tailscale Subnet Router

Create debian-cloudinit VM template

A template is a fully pre-configured operating system image that can used to deploy KVM virtual machines. Creating a special template is usually preferred over cloning an existing VM. Deploying virtual machines from templates is blazing fast, very comfortable and if you use linked clones you can optimize your storage by using base images and thin-provisioning.

Steps to create Ubuntu cloud init template in Proxmox

Choose your Ubuntu Cloud Image (replace with the url of the one you chose from above)

```
wget https://cloud-images.ubuntu.com/jammy/current/jammy-server-cloudimg-amd64.img
```

Create a new virtual machine. Here 9000 is just a VM id for the template, you can set it to anything you want but make sure change the id in other commands. The memory and cpu you set will be the initial value of the clone and can be increased later.

```
qm create 9000 --memory 2048 --core 2 --name ubuntu-cloud --net0 virtio,bridge=vmbr0
```

Import the downloaded Ubuntu disk to local-lvm storage

```
qm importdisk 9000 jammy-server-cloudimg-amd64.img local-lvm
```

Attach the new disk to the VM as a scsi drive on the SCSI controller

```
qm set 9000 --scsihw virtio-scsi-pci --scsi0 local-lvm:vm-9000-disk-0
```

Add cloud init drive

```
qm set 9000 --ide2 local-lvm:cloudinit
```

Make the cloud init drive bootable and restrict BIOS to boot from disk only

```
qm set 9000 --boot c --bootdisk scsi0
```

Add serial console

```
qm set 9000 --serial0 socket --vga serial0
```

DO NOT START YOUR VM

Now, configure hardware and cloud init, then create a template and clone. If you want to expand your hard drive you can on this base image before creating a template or after you clone a new machine.

Create template.

```
qm template 9000
```

Clone template.

```
qm clone 9000 150 --name myclone --full
```

Tailscale Subnet Router

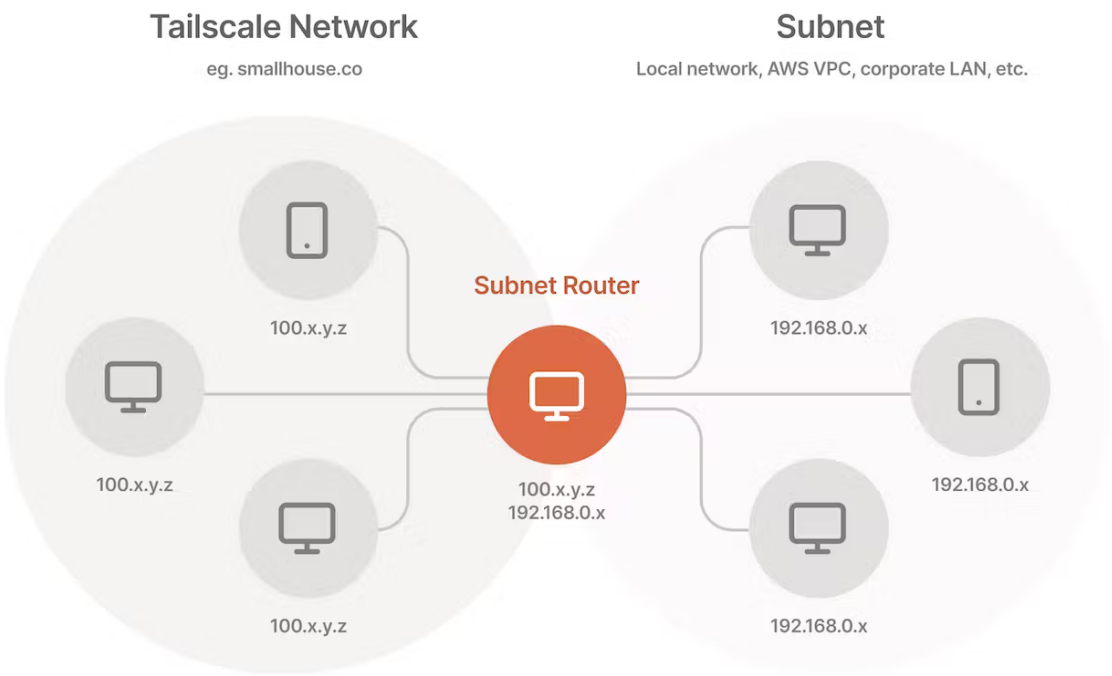
Subnet routers and traffic relay nodes

Tailscale works best when the client app is installed directly on every client, server, and VM in your organization. That way, traffic is end-to-end encrypted, and no configuration is needed to move machines between physical locations.

However, in some situations, you can't or don't want to install Tailscale on each device:

- With embedded devices, like printers, which don't run external software
- When connecting large quantities of devices, like an entire AWS VPC
- When incrementally deploying Tailscale (eg. on legacy networks)

In these cases, you can set up a "subnet router" (previously called a relay node or relaynode) to access these devices from Tailscale. Subnet routers act as a gateway, relaying traffic from your Tailscale network onto your physical subnet. Subnet routers respect features like access control policies, which make it easy to migrate a large network to Tailscale without installing the app on every device.



Step 1: Install the Tailscale client https://tailscale.com/download/linux

Step 2: Connect to Tailscale as a subnet router Enable IP forwarding

If your Linux system has a /etc/sysctl.d directory, use:

```
echo 'net.ipv4.ip_forward = 1' | sudo tee -a /etc/sysctl.d/99-tailscale.conf
echo 'net.ipv6.conf.all.forwarding = 1' | sudo tee -a /etc/sysctl.d/99-tailscale.conf
sudo sysctl -p /etc/sysctl.d/99-tailscale.conf
```

Otherwise, use:

```
echo 'net.ipv4.ip_forward = 1' | sudo tee -a /etc/sysctl.conf
echo 'net.ipv6.conf.all.forwarding = 1' | sudo tee -a /etc/sysctl.conf
sudo sysctl -p /etc/sysctl.conf
```

If your Linux node uses firewalld, you may need to also allow masquerading due to a known issue. As a workaround, you can allow masquerading with this command:

`firewall-cmd --permanent --add-masquerade`

Other distros may require different steps.

When enabling IP forwarding, ensure your firewall is set up to deny traffic forwarding by default. This is a default setting for common firewalls like ufw and firewalld, and ensures your device doesn't route traffic you don't intend. Advertise subnet routes

`sudo tailscale up --advertise-routes=192.168.0.0/24,192.168.1.0/24`

Replace the subnets in the example above with the right ones for your network. Both IPv4 and IPv6 subnets are supported.

If the device is authenticated by a user who can advertise the specified route in autoApprovers, then the subnet router's routes will automatically be approved. You can also advertise any subset of the routes allowed by autoApprovers in the tailnet policy file. If you'd like to expose default routes (0.0.0.0/0 and ::/0), consider using exit nodes instead.