IMPORTANT PARTNER UPDATE

Cambium Networks

Following recent developments at Cambium Networks, Bluechip IT is closely monitoring the situation.

Fixed Wireless:
Support remains in place and is expected to transition to the new company over the coming weeks.

Enterprise Wi-Fi & Switching:
There is currently uncertainty around Wi-Fi APs, cnMatrix switches and NSE.
Partners using **cnMaestro Cloud should begin planning a migration to cnMaestro** On-Premises, as cloud support for Enterprise devices may not continue beyond 1 October 2026.

NSE community resource: A community-developed GitHub tool is also available for locally managing Cambium NSE3000/NSE4000 devices via SSH without a cnMaestro cloud dependency. The developer explicitly notes that this is a **personal tool and not an official** Cambium product.
Link to GitHub tool: https://github.com/SiCambium/NSELocalSSH

Please note: third-party and community resources are provided for information only and are not developed, maintained or supported by Bluechip Infotech. Use of these resources is at the user's own risk, and Bluechip accepts no liability for any issues arising from their use.

Please read on to find the official advice from Cambium Networks as of September 18th, 2026.

Bluechip will continue to monitor developments and keep partners updated as further information becomes available.

Contact your Bluechip representative if you need assistance.

COMMUNITY SURVIVAL GUIDE

Self-Hosting cnMaestro On-Premises

Keeping your Cambium network under management on spare hardware, a home lab, or the cloud — after the vendor's exit.

Version 1.0 · September 2026

Covers cnMaestro On-Premises (written against 5.2.x; steps kept version-agnostic)

Tracks: Proxmox VE · Ubuntu Server + KVM · Cloud

Contents

Read this first — do these four things today

cnMaestro On-Premises runs entirely on your own hardware and manages your devices over your own LAN. It does not need Cambium's cloud to keep working. That is the good news, and it is why self-hosting is a viable exit rather than a eulogy. The risk isn't the software you already have — it's everything you haven't downloaded yet, which has an unfortunate habit of disappearing along with the vendor. So, before anything else:

ACT WHILE THE PORTAL IS UP

1. Download the On-Premises OVA now. Get it from the Cambium support portal (`support.cambiumnetworks.com`) while you still can. Download the newest build your hardware supports (see the AVX gate, Part 0).

2. Download the firmware you rely on now. On-premises normally pulls device firmware from Cambium's cloud repository. If that goes away you can still upload images manually — but only if you have them. Grab firmware for every device model you run.

3. Record checksums and keep offline copies. Run `sha256sum` on every file, write the hashes down, and keep at least two copies on separate media. See Appendix B.

4. Confirm your CPU has AVX before you commit to a current build — many old NUCs and laptops don't have it. One command, Part 0.

FREE TIER IS YOUR FRIEND

cnMaestro **Essentials** is the licence-free tier and delivers the core management experience with no subscription. Build your escape on Essentials. Anything that depends on the paid **cnMaestro X** entitlement may become impossible to activate or renew if the licensing service is wound down. *(The free tier being free is not in question. Whether on-prem truly never phones home is the bit worth proving to yourself — pull the WAN cable and confirm your devices stay managed before you bet the farm on it.)*

Part 0 — Before you begin

What still works once the vendor is gone

Best to be clear-eyed about which parts of this system genuinely depend on Cambium's infrastructure and which were quietly running fine on your LAN the whole time. Treat the "at risk" rows as planning assumptions, not guarantees.

Capability	Cloud-dependent?	Notes
Managing devices on your LAN	No	All local between appliance and devices.
cnMaestro Essentials features	No (licence-free)	The tier to standardise on.
Downloading the OVA / upgrades	**Yes** — support portal	At risk. Archive now.
Device firmware downloads	**Yes** — firmware CDN	At risk. Mirror now; manual upload works offline.
cnMaestro X entitlement / activation	**Yes** — licensing	At risk. Avoid depending on X.
Cloud-anchor / remote-access features	Yes	Expect these to stop. Not needed locally.
TAC / official support	Yes	Assume unavailable. This guide is your support.

Hardware eligibility — the AVX gate

This is the single most important check. Current cnMaestro builds bundle MongoDB 5.0 or later, and **MongoDB 5.0+ hard-requires a CPU with the AVX instruction set.** No AVX, no boot — the database service simply won't start, and it tells you so with all the warmth of a blank console or an "illegal instruction" error.

> **WHY THIS BITES THE "SPARE BOX" PLAN**
>
> AVX arrived with Intel Sandy Bridge and AMD Bulldozer in **2011**. Crucially, the cheap mini-PCs and NUCs most likely to be "gathering dust" often use Atom-derived Celeron/Pentium chips (Apollo Lake, Gemini Lake, Jasper Lake) that **have no AVX at all**, right up to around 2021. So "an old NUC" is a coin toss, not a safe bet. Check before you invest any time.

Run these two checks on any candidate machine (from a Linux live USB if it has no OS yet):

```
grep -qw avx /proc/cpuinfo && echo "AVX: yes" || echo "AVX: NO -- current cnMaestro will not run"
grep -qwE 'vmx|svm' /proc/cpuinfo && echo "HW virt: yes" || echo "HW virt: NO or disabled in BIOS"
```

Hardware	AVX?	Verdict
2011+ Intel Core i3/i5/i7, or modern AMD/Ryzen	Yes	Good to go.
Modern Xeon / EPYC	Yes	Good (some older Xeons lack it — check).
Celeron/Pentium J&N (Apollo/ Gemini/Jasper Lake)	Usually No	Likely won’t run current builds.
Pre-2011 Core 2 Duo / old Atom	No	Won’t run current builds.

ENABLE VIRTUALISATION IN FIRMWARE

If the second check says virtualisation is off, reboot into BIOS/UEFI and enable Intel VT-x / VT-d or AMD-V / SVM. Consumer motherboard makers ship it disabled by default for reasons they have never adequately explained to anyone.

NO AVX? YOU STILL HAVE TWO OPTIONS

Option 1 — an older cnMaestro build. Builds predating the MongoDB 5.0 bump run on non-AVX hardware. The cnMaestro 3.x line is the safe non-AVX choice; pick the newest 3.x you can find for the best device support. *(Nobody published the exact build where AVX became mandatory, so treat 3.x as a good bet rather than a certainty. The real test is empirical anyway: if it boots to a login prompt instead of an “illegal instruction” or a dead Mongo, you’re in business.)* Trade-offs: older builds may not support your newest device models, get no security patches, and are still behind the same portal — so archive one now.

Option 2 — run it in the cloud (Part 4). Every modern cloud instance has AVX, so this sidesteps the problem entirely — at the cost of a monthly bill and a different security posture.

Sizing: CPU, RAM, disk

The appliance ships pre-configured, but you choose how much host resource to give it. The tiers below are the long-standing Cambium guidance by managed-device count. *(These figures are from older On-Premises documentation; current 5.x is hungrier, so treat this table as a floor, not a target, and round up.)*

Managed devices	vCPU	RAM	Disk
Up to 100	2	4 GB	≥ 80 GB
100 – 1,000	4	8 GB	≥ 120 GB
1,000 – 4,000	4–8	8–16 GB	≥ 250 GB
4,000 – 10,000	8+	16–32 GB	≥ 500 GB

SENSIBLE DEFAULT FOR THE TYPICAL READER

A hotel, school, or small WISP is almost always under a few hundred devices. Give the VM **4 vCPU, 8 GB RAM, and at least 120 GB of disk** for comfortable headroom. Leave at least 2 GB of RAM and a CPU core for the host — which is exactly why we do not install a desktop environment on the host (Part 2).

Choosing your track

Follow the branch that matches your situation. The same options are written out in the table below.

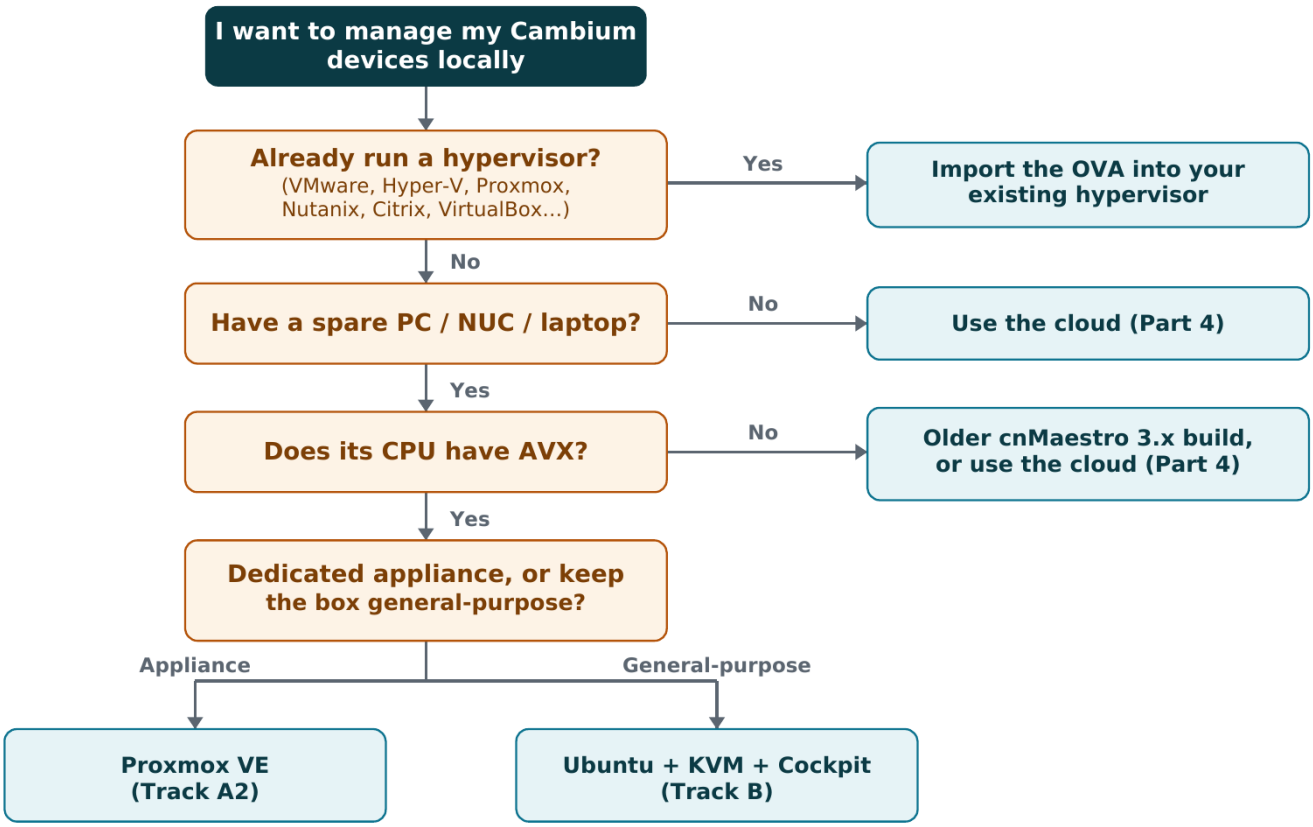

Figure 1 — Routing you to the right track. AVX and CPU-passthrough caveats apply on every branch.

Your situation	Use
You already run a hypervisor (VMware, Hyper-V, Proxmox, Nutanix, Citrix, VirtualBox…)	**Use it** — import the OVA there (see below); or **Track A1** for Proxmox
A spare box you want as a dedicated appliance, easiest web UI, built-in backups	**Track A2** — fresh Proxmox
A box you want to keep general-purpose (other services too), or you prefer Ubuntu	**Track B** — Ubuntu + KVM + Cockpit
No AVX-capable hardware, or you want it off-site with no hardware to buy	**Part 4** — cloud

Tracks A and B produce an identical running appliance; they differ only in host platform and management UI. Whichever you pick, you finish at Part 3.

OFFICIALLY UNSUPPORTED, PRACTICALLY FINE

Cambium's officially validated hypervisor was VMware ESXi. KVM (which both Proxmox and the Ubuntu track use underneath) is not officially supported, but cnMaestro runs on it reliably — the community has done it for years. Pick the platform you'll be happiest maintaining.

Already have a hypervisor? Use it

You may not need a new host at all. Because cnMaestro ships as an OVA — a portable virtual-machine export — the quickest route is often to import it into whatever you already run. In practice it has been run successfully across essentially every mainstream hypervisor: **VMware** ESXi/Workstation, **Microsoft Hyper-V**, **Oracle VirtualBox**, **Proxmox VE**, **Nutanix AHV**, **Citrix Hypervisor / XCP-ng**, and plain **libvirt/virt-manager**.

RULE OF THUMB

If your hypervisor can run a 64-bit Linux VM and expose AVX to the guest, cnMaestro will almost certainly run on it. You don't have to move to Proxmox or Ubuntu if you're already happy elsewhere — those two are simply the best free, from-scratch options for someone starting with a bare box.

Two things hold true on every platform, plus a per-platform import note:

- **AVX must reach the guest.** On KVM-based platforms (Proxmox, Nutanix AHV, libvirt) set the virtual CPU to host / host-passthrough so AVX isn't masked. VMware and VirtualBox pass host CPU features through by default.
- **Put it on your LAN with a bridge, and give it both disks** (OS + data). The `/mnt/data` UUID fix in section 3.1 applies no matter which hypervisor you use.

Platform	How to import
VMware (ESXi / Workstation)	Import the OVA directly — Deploy OVF/OVA or File → Open.
VirtualBox	Import the OVA directly — File → Import Appliance.
Proxmox VE	`qm importovf` (Track A1), or extract + `qemu-img convert`.
libvirt / virt-manager	Extract + convert to qcow2 + `virt-install --import` (Track B).
Nutanix AHV	Extract VMDKs, upload via Image Configuration, create a VM and attach both disks; set CPU passthrough.
Citrix Hypervisor / XCP-ng	Import the OVA/OVF via XenCenter/Xen Orchestra; attach both disks.
Microsoft Hyper-V	Convert disks to VHDX (`qemu-img convert -O vhdx disk.vmdk disk.vhdx`), create a Generation 1 VM, attach both disks, use a standard network adapter.

Part 1 — Track A: Proxmox VE (recommended)

Proxmox VE is a purpose-built, free virtualisation platform with a browser UI, built-in snapshots and backups, and bridged networking out of the box. For most readers it's the least fiddly way to run one appliance reliably on a spare machine, with the fiddliness budget instead spent on the appliance itself. It takes over the whole disk, so use a box you can dedicate to it.

A1. You already run Proxmox — just import

The fast path, and the procedure many in the community already use. All commands run in the Proxmox host shell (web UI Shell or SSH).

Step 1 — Confirm the host CPU has AVX

```
grep -qw avx /proc/cpuinfo && echo "AVX support detected" || echo "no AVX support"
```

Step 2 — Get the OVA onto the host and extract it

```
# copy the OVA to the host first (scp/USB), then:
tar xvf cnmaestro*.ova
```

That unpacks the `.ovf` descriptor, one or more `.vmdk` disks, an `.mf` manifest, and a validation archive.

> **IF EXTRACTION "FINDS NOTHING"**
>
> Use the glob form `tar xvf cnmaestro*.ova` and make sure you're in the directory holding the OVA. A common mistake is trying to name individual members — just extract the whole archive.

Step 3 — Import the OVF (creates the VM and its disks)

```
qm list                          # find a free VM ID, e.g. 105
qm importovf 105 ./cnmaestro_on-premises_5.x.x-rx_amd64.ovf local-lvm
```

Replace `local-lvm` with your target storage (Datacenter → Storage; could be `local-zfs`, `local`, etc.). Match the `.ovf` filename to what you extracted.

Step 4 — Set hardware for KVM and your LAN

```
qm set 105 --name cnMaestro \
  --memory 8192 --cores 4 --sockets 1 \
  --cpu host \
  --scsihw virtio-scsi-pci \
  --net0 virtio,bridge=vmbr0
qm set 105 --onboot 1            # start automatically after a power cut
```

-CPU HOST IS NOT OPTIONAL

With Proxmox's default CPU type (`kvm64`), **AVX is masked even on a CPU that has it**, and MongoDB won't start. Setting `--cpu host` exposes the real CPU features, including AVX. The only trade-off: live-migrating this VM to a host with a different CPU may not work — irrelevant for a single box.

Step 5 — Check the disks are attached, then start

Open VM 105 → Hardware. Depending on your Proxmox version, imported disks may appear as `Unused Disk 0/1`. If so, double-click each and attach it (prefer the VirtIO SCSI controller), then set the boot order under Options → Boot Order so the OS disk is first and enabled. Then:

```
qm start 105
```

Open the Console. First boot can sit on a blank screen for 10–15 minutes. If no login prompt appears after that, reboot the VM once. Now go to Part 3.

A2. Fresh Proxmox install on a spare box

THIS WIPES THE TARGET DISK

The Proxmox installer erases the disk you install it to. Back up anything on that machine first, and be certain you've selected the right disk during install.

Step 1 — Make the installer USB

Download the Proxmox VE ISO from `proxmox.com/downloads`. Write it to a USB stick (≥2 GB). On Linux, identify the stick carefully with `lsblk`, then:

```
# DANGER: /dev/sdX must be the USB stick, not your disk. Double-check with lsblk.
sudo dd if=proxmox-ve_*.iso of=/dev/sdX bs=4M status=progress oflag=sync
```

On Windows/macOS, use balenaEtcher. Boot the target machine from the stick (may need Secure Boot disabled).

Step 2 — Install

Follow the graphical installer: accept the licence, pick the target disk, set country/timezone/keyboard, set a strong `root` password and an admin email, and give the box a **static management IP** on your LAN (e.g. `192.0.2.10/24`, gateway `192.0.2.1`). Reboot and remove the USB.

Step 3 — First login & repository fix

Browse to `https://192.0.2.10:8006` (accept the self-signed certificate warning) and log in as `root`. Proxmox will nag you about a subscription on every login; without one, you must switch to the free "no-subscription" repository or updates will fail. In the host Shell:

Proxmox VE 9 (Debian 13) — modern deb822 format:

```
# disable the enterprise repos
sed -i 's/^Enabled: yes/Enabled: no/' /etc/apt/sources.list.d/pve-enterprise.sources 2>/dev/
null
sed -i 's/^Enabled: yes/Enabled: no/' /etc/apt/sources.list.d/ceph.sources 2>/dev/null

# add the no-subscription repo
cat > /etc/apt/sources.list.d/pve-no-subscription.sources <<'EOF'
Types: deb
URIs: http://download.proxmox.com/debian/pve
Suites: trixie
Components: pve-no-subscription
Signed-By: /usr/share/keyrings/proxmox-archive-keyring.gpg
EOF

apt update && apt -y dist-upgrade
```

Proxmox VE 8 (Debian 12) — legacy `.list` format:

```
sed -i 's/^deb/#deb/' /etc/apt/sources.list.d/pve-enterprise.list
echo 'deb http://download.proxmox.com/debian/pve bookworm pve-no-subscription' \
  > /etc/apt/sources.list.d/pve-no-subscription.list
apt update && apt -y dist-upgrade
```

CHECK YOUR VERSION FIRST

Run `pveversion` and `cat /etc/debian_version`. Trixie = PVE 9, Bookworm = PVE 8. Use the matching block; mixing them will break `apt`.

Step 4 — Import cnMaestro

Now follow A1 from Step 2 onward. Then go to Part 3.

Part 2 — Track B: Ubuntu Server + KVM + Cockpit

Choose this if you want the box to stay a general-purpose Linux server, or you're more comfortable in Ubuntu. We use **Ubuntu Server** — not a desktop spin — because every gigabyte a desktop consumes is resource stolen from the appliance. The management UI is **Cockpit**, a lightweight web console, so you never need a graphical desktop on the server.

> **WHY NOT KUBUNTU / A DESKTOP?**
>
> A KDE or GNOME desktop idles at 1–2 GB of RAM and adds background services and attack surface, purely to run one GTK app (virt-manager). Cockpit gives you a browser UI from your laptop for a fraction of that. Keep the server headless.

B1. Install Ubuntu Server (minimal)

Download **Ubuntu Server 24.04 LTS** and write the ISO to USB as in A2 Step 1. Boot it and follow the installer:

- Choose the minimised/normal server install — do not select any desktop.
- Set a **static IP** for the server (e.g. `192.0.2.11/24`) — we'll turn this into a bridge shortly.
- Enable **OpenSSH server** for remote management.
- Skip the snap extras. Create your admin user.

After first boot, update the system:

```
sudo apt update && sudo apt -y full-upgrade && sudo reboot
```

B2. Install the virtualisation stack

```
sudo apt update
sudo apt -y install qemu-system-x86 qemu-utils \
  libvirt-daemon-system libvirt-clients virtinst

sudo systemctl enable --now libvirtd
```

Add yourself to the groups that allow managing VMs without `sudo`, then **log out and back in** (group membership only applies to new sessions):

```
sudo usermod -aG libvirt,kvm "$USER"
# log out and back in, then verify:
id | tr ',' '\n' | grep -E 'libvirt|kvm'
```

Confirm the host is ready for hardware virtualisation:

```
virt-host-validate qemu
```

Everything relevant should say `PASS`. An IOMMU warning is fine for this use case.

B3. Add the Cockpit web UI

```
sudo apt -y install cockpit cockpit-machines
```

Cockpit is socket-activated, so it uses almost nothing until you open it. Browse to `https://192.0.2.11:9090` and log in with your server account. The Virtual machines section (from `cockpit-machines`) is where you create and control the appliance if you prefer clicking to the CLI.

> **KEEP THE CONSOLES OFF THE INTERNET**
>
> Cockpit (9090) and libvirt are management surfaces. Keep them on your LAN/management VLAN only — never port-forward them from the internet. See Part 5.

B4. Bridged networking

By default a KVM guest sits behind NAT and isn't directly reachable by your devices. cnMaestro needs to be a first-class citizen on your LAN, so we put it on a **bridge**. On Ubuntu Server this is done with netplan.

> **YOU CAN LOCK YOURSELF OUT**
>
> Reworking the network over SSH can drop your connection. Have console/KVM access before you start. Use `sudo netplan try`, which auto-reverts after 120 seconds if you don't confirm.

Find your interface name with `ip -br link` (e.g. `enp1s0`). Create `/etc/netplan/01-br0.yaml` (replace the interface name and addresses):

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp1s0:
      dhcp4: no
      dhcp6: no
  bridges:
    br0:
      interfaces: [enp1s0]
      addresses: [192.0.2.11/24]
      routes:
        - to: default
          via: 192.0.2.1
      nameservers:
        addresses: [192.0.2.1, 9.9.9.9]
      parameters:
        stp: false
        forward-delay: 0
```

Lock the file's permissions (netplan warns if it's world-readable), then test and apply:

```
sudo chmod 600 /etc/netplan/01-br0.yaml
sudo netplan try        # confirm within 120s if your connection survives
sudo netplan apply
ip -br addr show br0
```

B5. Import the OVA

Step 1 — Extract and inspect

```
mkdir -p ~/cnmaestro && cd ~/cnmaestro
tar xvf /path/to/cnmaestro_on-premises_5.x.x-rx_amd64.ova
ls -lh *.vmdk
grep -i 'ovf:href\|Disk ' *.ovf   # see disk order/roles in the descriptor
```

cnMaestro's OVA usually contains **two** disks: a smaller OS disk and a larger data disk (the one that becomes `/mnt/data`). You must convert and attach both.

Step 2 — Convert each disk to qcow2

```
# adjust the source names to match your .vmdk files; the smaller is usually the OS disk
qemu-img convert -p -O qcow2 cnmaestro-disk1.vmdk /var/lib/libvirt/images/cnmaestro-os.qcow2
qemu-img convert -p -O qcow2 cnmaestro-disk2.vmdk /var/lib/libvirt/images/cnmaestro-data.qcow2
sudo chown libvirt-qemu:kvm /var/lib/libvirt/images/cnmaestro-*.qcow2
```

QCOW2 VS RAW

qcow2 is the right choice: thin-provisioned and, most importantly, it supports snapshots — your rollback safety net before every upgrade (Part 5). The performance difference versus raw is negligible for this workload; you are not running a hyperscaler.

Step 3 — Define and start the VM

```
sudo virt-install \
  --name cnmaestro \
  --memory 8192 \
  --vcpus 4 \
  --cpu host-passthrough \
  --import \
  --disk path=/var/lib/libvirt/images/cnmaestro-os.qcow2,format=qcow2,bus=virtio \
  --disk path=/var/lib/libvirt/images/cnmaestro-data.qcow2,format=qcow2,bus=virtio \
  --network bridge=br0,model=virtio \
  --os-variant ubuntu22.04 \
  --graphics vnc,listen=127.0.0.1 \
  --video virtio \
  --noautoconsole
```

-CPU HOST-PASSTHROUGH IS MANDATORY

Exactly as on Proxmox: without it, KVM presents a generic CPU with AVX masked and MongoDB fails to start. `host-passthrough` exposes the real CPU. If `--os-variant ubuntu22.04` is rejected, run `osinfo-query os | grep -i ubuntu` and pick a valid one, or use `--os-variant generic`.

Step 4 — Autostart and console

```
sudo virsh autostart cnmaestro
sudo virsh list --all
```

View the console from Cockpit (Virtual machines → cnmaestro → Console) or with `virsh console cnmaestro`. Allow 10–15 minutes for first boot. Then go to Part 3.

> **PREFER CLICKING? USE COCKPIT INSTEAD OF VIRT-INSTALL**
>
> In Cockpit: Virtual machines → Create VM → Import an existing disk image, point it at `cnmaestro-os.qcow2`, set memory/vCPUs, and before finishing set the CPU to host-passthrough and the network to `br0`. After creation, use Add disk to attach `cnmaestro-data.qcow2` on the VirtIO bus. virt-manager over SSH X11-forwarding is a third option if you have a Linux desktop.

Part 3 — Post-import & first run (both tracks)

From here the steps are identical whether you used Proxmox, KVM/Cockpit, or the cloud — everything below happens inside the appliance.

LOGGING IN TO THE APPLIANCE CONSOLE

The appliance boots to a console login. Default credentials are documented in Cambium's On-Premises User Guide — older builds commonly used `cambium` / `cnmaestro`. *(Cambium has changed these between releases before, so treat it as a starting guess rather than a promise, and check your build's manual if the appliance is unimpressed.)* **Change the password immediately** after first login.

3.1 The /mnt/data fstab fix

This fix turns a demoralising "Service Temporarily Unavailable" page into a working appliance, and it applies on every hypervisor. The appliance's data partition is mounted by device name (e.g. `/dev/sdb1`) in `/etc/fstab`. Under KVM/Proxmox the disk can enumerate in a different order, so that device name may not exist at boot, `/mnt/data` doesn't mount, and cnMaestro can't start. The fix is to mount by the partition's stable **UUID** instead.

```
# on the appliance console:
sudo blkid                      # find the UUID of the ext4 data partition (was /dev/sdb1)
sudo nano /etc/fstab
```

Comment out the old device-based line and add a UUID-based one:

```
# /dev/sdb1  /mnt/data  ext4  noatime,errors=panic  0  2      <-- comment this out
UUID=<uuid-from-blkid>  /mnt/data  ext4  noatime,errors=panic,nofail  0  2
```

Save, then reboot and confirm the mount and service:

```
sudo reboot
# after reboot:
mount | grep /mnt/data && echo "data mounted OK"
```

WHY NOFAIL

`nofail` lets the appliance still boot (and let you in to fix things) if that disk ever fails to appear, instead of dropping to an emergency shell. Credit for this fix to the community — contributed to the Proxmox thread by micampo.

3.2 A static IP that survives reboots

Set a static management IP for the appliance so your devices and bookmarks don't chase a moving DHCP lease. Prefer the appliance's own configuration tool / setup wizard where offered.

GOTCHA: CNMAESTRO REWRITES ITS OWN NETPLAN ON BOOT

The appliance regenerates its netplan config at boot, which can wipe a hand-edited static IP. The community workaround is to write the static config and then make the file immutable — but with an important catch: **remove any redundant `dhcp4: false` line first**, because if the file is immutable and the appliance tries to add a line it can't, boot hangs. This was fixed in cnMaestro 3.0.3 and later. Sequence:

```
sudo nano /etc/netplan/<file>.yaml     # set your static address; delete any 'dhcp4: false'
line
sudo netplan apply
sudo chattr +i /etc/netplan/<file>.yaml # make immutable so it can't be overwritten
# to edit later: sudo chattr -i <file>, change, netplan apply, chattr +i again
```

3.3 First web login & the free tier

Browse to `https://<appliance-ip>` (not :8006 or :9090 — that's the host). Complete the setup wizard: set the admin account and a strong password, set the system name, and confirm the time zone. When asked about the tier, select/confirm **Essentials** — it needs no paid entitlement.

3.4 Onboarding your devices

Devices need to be told where your on-prem server is. Any one of these works; pick what suits your network:

Method	How	Best for
Manual claim	Home → Onboard Devices → Claim, enter the device MAC(s)	A handful of devices
Set URL on the device	In the AP's own UI/CLI, set the cnMaestro URL to `https://<appliance-ip>`	Per-device control
DNS (Option 15 + record)	Create an A record `cnmaestro.<yourdomain>` → appliance IP; hand devices the domain via DHCP Option 15	Reliable fleet-wide default
DHCP Option 43	Return the cnMaestro URL in the vendor-specific option (syntax varies by DHCP server)	Existing managed DHCP
Auto-provisioning	Shared Settings → Auto-Provisioning: auto-approve by subnet and assign to sites	Zero-touch at scale

THE DNS METHOD IS THE MOST PORTABLE

An A record for `cnmaestro.<yourdomain>` pointing at the appliance, combined with your DHCP handing out that domain (Option 15), is the least fragile fleet-wide approach and doesn't depend on vendor-specific Option 43 encodings. Option 43's exact format differs between ISC dhcpd, dnsmasq, MikroTik, and Windows DHCP — consult your DHCP server's syntax.

3.5 Time sync

Correct time matters for TLS, certificate validity, and coherent statistics. Point the appliance at your internal NTP server or a public one, and confirm it's synced before onboarding devices in earnest.

Part 4 — Running it in the cloud

The cloud is the natural answer when you have **no AVX-capable hardware**, no spare box at all, or you want the appliance off-site for resilience. Every current cloud instance family supports AVX, so the Part 0 gate simply doesn't apply — you can stop grepping `/proc/cpuinfo` and start worrying about the bill instead.

> **TWO THINGS CHANGE IN THE CLOUD**
>
> **Cost:** an always-on VM is a recurring monthly bill, unlike a free spare box. **Exposure:** your devices now reach the appliance across the internet, so the management plane must be locked down. This is a materially different security posture — read the hardening note below before you deploy.

AWS

Import your saved OVA as an AMI with VM Import/Export (upload to S3, then `aws ec2 import-image`), or launch the official AMI if still available. Size it around `t3.large` – `t3.xlarge` (2–4 vCPU, 8–16 GB) with a ≥120 GB `gp3` volume. In the security group, allow the admin UI only from your office/VPN IP, and device-management ports only from the networks your devices live on — never `0.0.0.0/0`.

Google Cloud

Use `gcloud compute images import` to bring in the OVA/qcow2, then launch an `e2-standard-2` / `e2-standard-4`. Restrict firewall source ranges exactly as above.

Microsoft Azure

Convert the qcow2 to a fixed-size VHD (`qemu-img convert -f qcow2 -O vpc -o subformat=fixed,force_size cnmaestro-os.qcow2 cnmaestro-os.vhd`), upload it, create a managed image, and launch a VM (2–4 vCPU, 8–16 GB). Restrict the Network Security Group as above.

> **CLOUD SECURITY POSTURE — NON-NEGOTIABLE**
>
> Never expose the cnMaestro admin UI to the whole internet. Put management behind a VPN or bastion, allow-list source IPs, enforce TLS with a real certificate, keep the host OS patched, and remember that device telemetry now leaves your premises — check your data-protection obligations for where that data lives.

Part 5 — Keeping it alive

Getting it running is the easy half. Resilience and backups are now entirely on you, and this part is what turns a stopgap into something you can rely on for years.

Snapshots & rollback

Take a snapshot before every change that could go wrong — upgrades especially. It’s your instant undo.

```
# Proxmox
qm snapshot 105 pre-upgrade-$(date +%F)
# roll back:  qm rollback 105 pre-upgrade-YYYY-MM-DD

# KVM / libvirt
sudo virsh snapshot-create-as cnmaestro pre-upgrade-$(date +%F) --disk-only --atomic
# list:  sudo virsh snapshot-list cnmaestro
```

Backups (3-2-1)

Snapshots live on the same box — they are not backups. Keep **three** copies, on **two** media, with **one** off-site. Two layers:

- **Application backup:** use cnMaestro’s own scheduled backup/export (appliance UI under Operations / Server) and copy the file off the box automatically.
- **VM backup:** Proxmox has built-in Backup jobs (vzdump). On the Ubuntu track, back up the qcow2 files with `restic` to off-site storage — snapshot first, or shut the VM down, so the image is consistent:

```
# Ubuntu/KVM example: consistent restic backup of the appliance disks
sudo virsh shutdown cnmaestro
restic -r /srv/backups/cnmaestro backup /var/lib/libvirt/images/cnmaestro-*.qcow2
sudo virsh start cnmaestro
```

Upgrades without the portal

Upgrades normally come from the support portal, which may not last. So **archive every upgrade package and OVA you can get now**, and note the required upgrade path — cnMaestro expects you to step through intermediate versions rather than jumping (for example, older installs must reach 4.1.0 before 5.0.0). Always snapshot before upgrading, and keep the previous working OVA so you can rebuild from scratch if an upgrade fails.

Laptop-as-server & power

An old laptop is an excellent host: built-in screen and keyboard (KVM) and a battery (a free UPS). Two changes stop it sleeping when you close the lid. Apply these on the **host** OS (Ubuntu for Track B; the Debian host for Proxmox):

```
sudo sed -i 's/^#\?HandleLidSwitch=.*/HandleLidSwitch=ignore/' /etc/systemd/logind.conf
sudo sed -i 's/^#\?HandleLidSwitchExternalPower=.*/HandleLidSwitchExternalPower=ignore/' /etc/
systemd/logind.conf
sudo sed -i 's/^#\?HandleLidSwitchDocked=.*/HandleLidSwitchDocked=ignore/' /etc/systemd/
logind.conf
sudo systemctl restart systemd-logind
sudo systemctl mask sleep.target suspend.target hibernate.target hybrid-sleep.target
```

PRACTICAL NOTES

Don’t run it shut inside a drawer — laptops need airflow; prop the lid open or keep vents clear. Ensure the VM autostarts on boot (`qm set --onboot 1` / `virsh autostart`) so it recovers after a power cut. For a desktop or NUC, add a small UPS to get the graceful ride-through a laptop gets for free.

Security hardening

- **Patch the host.** Ubuntu: enable `unattended-upgrades`. Proxmox: apply updates regularly from the no-subscription repo.
- **Minimise exposure.** Keep management surfaces (Proxmox 8006, Cockpit 9090, SSH, and the cnMaestro admin UI) on a management VLAN. Don’t port-forward them from the internet. A host firewall with nftables allowing only needed ports is a sensible default.
- **Strong, unique credentials** everywhere, and change the appliance’s default console password immediately.
- **TLS:** replace the appliance’s self-signed certificate with one from your internal CA (or a public cert) so device connections and admin logins are properly trusted.
- **Watch disk space.** The database and statistics grow over time; alert on the data volume filling, and test a restore at least once.

Appendices

A. Command quick-reference

Eligibility

```
grep -qw avx /proc/cpuinfo && echo "AVX yes" || echo "AVX NO"
grep -qwE 'vmx|svm' /proc/cpuinfo && echo 'virt yes' || echo 'virt NO/BIOS'
```

Proxmox (Track A)

```
tar xvf cnmaestro*.ova
qm list
qm importovf 105 ./cnmaestro_*.ovf local-lvm
qm set 105 --name cnMaestro --memory 8192 --cores 4 --sockets 1 \
  --cpu host --scsihw virtio-scsi-pci --net0 virtio,bridge=vmbr0 --onboot 1
qm start 105
```

Ubuntu + KVM (Track B)

```
sudo apt -y install qemu-system-x86 qemu-utils libvirt-daemon-system \
  libvirt-clients virtinst cockpit cockpit-machines
sudo usermod -aG libvirt,kvm "$USER"     # then log out/in
tar xvf cnmaestro*.ova
qemu-img convert -p -O qcow2 disk1.vmdk /var/lib/libvirt/images/cnmaestro-os.qcow2
qemu-img convert -p -O qcow2 disk2.vmdk /var/lib/libvirt/images/cnmaestro-data.qcow2
sudo virt-install --name cnmaestro --memory 8192 --vcpus 4 \
  --cpu host-passthrough --import \
  --disk path=/var/lib/libvirt/images/cnmaestro-os.qcow2,bus=virtio \
  --disk path=/var/lib/libvirt/images/cnmaestro-data.qcow2,bus=virtio \
  --network bridge=br0,model=virtio --os-variant generic --noautoconsole
sudo virsh autostart cnmaestro
```

Appliance post-import

```
sudo blkid                              # get data-partition UUID
# /etc/fstab:  UUID=<uuid> /mnt/data ext4 noatime,errors=panic,nofail 0 2
```

B. Verifying your download

Prove a copy is intact and untampered — essential for files passed around the community after the portal is gone.

```
# create a checksum when you first download from the official portal
sha256sum cnmaestro_on-premises_5.x.x-rx_amd64.ova > cnmaestro.ova.sha256

# anyone can verify a later copy against that value
sha256sum -c cnmaestro.ova.sha256      # prints "OK" if it matches
```

COMMUNITY SUGGESTION

If you're sharing files, publish a signed checksum manifest (SHA-256 for each OVA/firmware) so others can confirm they have a genuine, unmodified copy. Never trust an unverified image with access to your network.

C. Troubleshooting

Symptom	Likely cause	Fix
VM won't boot; "illegal instruction"; Mongo won't start	AVX absent or masked by the virtual CPU	Set `--cpu host` / `host-passthrough`. If the CPU truly lacks AVX, use an older build (Part 0) or the cloud.
Web UI shows "Service Temporarily Unavailable"	`/mnt/data` not mounted (device-name fstab)	Apply the UUID fstab fix (3.1).
`tar` "finds nothing" in the OVA	Wrong directory or naming members	`tar xvf cnmaestro*.ova` in the OVA's folder.
Blank console at first boot	Long first-boot initialisation	Wait 10–15 min; if still nothing, reboot the VM once.
Static IP resets after reboot	Appliance rewrites its netplan	Use the UI, or the immutable-file method (3.2); requires 3.0.3+.
Devices don't appear	Onboarding/reachability/time skew	Check Option 43/DNS/manual URL, firewall path, and NTP sync.
Locked out after netplan edit (host)	Bridge misconfig dropped SSH	Use console; `netplan try` auto-reverts in 120 s.
Only one disk imported (Track B)	Forgot the second (data) disk	Convert and attach both VMDKs on the VirtIO bus.

D. Sources & further reading

These community threads and documents informed this guide. They live on Cambium's infrastructure and **may go offline** — the procedures above are reproduced so you don't need them, but they're listed for provenance and credit.

- Installing cnMaestro on Proxmox — community thread (the Proxmox procedure this guide reproduces): `community.cambiumnetworks.com/t/installing-cnmaestro-on-proxmox/106133`
- Deploying cnMaestro NOC on KVM — community thread: `community.cambiumnetworks.com/t/deploying-cnmaestro-noc-on-kvm/47437`
- cnMaestro 5.0.0 / 5.2.0 (On-Premises) Release Notes — Cambium Community
- cnMaestro On-Premises User Guide & Quick Start — Cambium Networks
- cnMaestro AWS AMI (Marketplace) & AMI release notes — Cambium Community
- Device onboarding & DHCP options (Option 15/43) — Cambium Community
- MongoDB 5.0+ requires AVX — MongoDB documentation / docker-library/mongo issue #619

- Proxmox VE package repositories (no-subscription) — `pve.proxmox.com/wiki/Package_Repositories`
- Cambium Networks enters administration (Sept 2026) — ISPreview, Data Center Dynamics, Total Telecom

---

Community survival guide · v1.0 · September 2026 · Unofficial and unaffiliated · Provided as-is with no warranty, express, implied, or emotional · Verify everything against your own environment.

Networking & Wi-Fi Business

Extreme networks

extreme@bluechipit.com.au

Extreme Networks offers high-performance, enterprise-grade switching and Wi-Fi solutions, managed primarily via its ExtremeCloud IQ platform. Replacing Cambium's high-density Wi-Fi and enterprise switching, Extreme excels in complex, multi-site environments like education, healthcare, venue, and municipal projects. It sits at the top tier of the enterprise market, competing directly with Cisco and Aruba by providing advanced AI-driven analytics, zero-touch deployment, and granular network management.

Allied Telesis™

matthewh@bluechipit.com.au

Allied Telesis delivers ruggedised, highly secure, and resilient switching and routing solutions designed for harsh operational environments. Replacing Cambium's industrial and enterprise switching lines, it shines in industrial automation, smart cities, defense, and mission-critical infrastructure where high uptime is non-negotiable. It occupies the mid-to-high enterprise market, differentiating itself through robust hardware, Autonomous Wave Control (AWC) for smart wireless, and self-healing network security features.

Omada by tp-link

tplink@bluechipit.com.au

TP-Link Omada provides a cost-effective, cloud-managed networking ecosystem that includes access points, managed switches, and security gateways. Serving as a direct replacement for Cambium's cnPilot and lower-tier cnMatrix lines, Omada targets the price-sensitive SMB, hospitality, and residential multi-dwelling unit (MDU) markets. It sits in the budget-to-mid SMB space, offering feature parity for core networking needs alongside free or low-cost cloud controller options that drastically lower total cost of ownership.

HUAWEI eKit

huawei@bluechipit.com.au

Huawei eKit is a tailored networking portfolio offering Wi-Fi 6 access points, unmanaged/managed switches, and routers designed specifically for small-to-medium enterprises. It serves as a drop-in replacement for Cambium's SMB Wi-Fi and switching tiers in regions where Huawei infrastructure is deployed. Positioned squarely in the competitive SMB to mid-market space, it delivers high-spec performance and easy mobile-app management at an aggressive price point.

Contact your Bluechip Infotech representative
P 1300 72 74 74 www.bluechipit.com.au

Fixed Wireless Business (PTP / PTMP)

RADWIN

leightonr@bluechipit.com.au

Radwin provides high-capacity, carrier-grade sub-6 GHz Point-to-Point (PTP) and Point-to-Multipoint (PTMP) wireless broadband solutions. It is a direct 1:1 replacement for Cambium's ePMP, PTP 670/700, and cnWave series in challenging Line-of-Sight (LOS) and Non-Line-of-Sight (NLOS) environments. Positioned at the premium end of the wireless market, Radwin serves WISPs, critical infrastructure, rail/transit, and video surveillance networks that demand guaranteed SLA performance and heavy interference mitigation.

CERAGON

leightonr@bluechipit.com.au

Ceragon is a global specialist in ultra-high-capacity wireless backhaul, delivering high-frequency microwave and millimeter-wave links. It replaces Cambium's high-capacity backhaul portfolio (PTP 820/850 series) for long-haul and high-throughput requirements. Positioned in the top-tier carrier and enterprise backhaul market, Ceragon targets telecommunication operators, large ISPs, offshore facilities, and utility networks needing multi-gigabit throughput over long distances.