Your Proxmox Home Server

Build a Personal Cloud with VMs, Docker, Pi-hole, Jellyfin, and Home Assistant on Any Mini PC

Nova X Lab

Your Proxmox Home Server

Marque éditoriale : Nova X Lab

ISBN : 978-2-489108-09-5

Éditeur : Nova Publishing SAS, 01280 Prévessin-Moëns, France

Contents

Read This First — The Full-Stack Build Contract

This book is written as a full-stack build. You can skip optional services, but the default path assumes the whole homelab is built in sequence. Do not jump directly to a later service unless its dependencies are already working.

The address plan has one rule: backend IPs describe where a service actually runs; friendly .home names describe how you reach it after Pi-hole and NGINX Proxy Manager are in place. For example, Jellyfin runs at 192.168.1.21:8096, but jellyfin.home points to NPM at 192.168.1.20, and NPM forwards the request to Jellyfin.

Important version boundary: Proxmox VE 9 uses Debian 13 Trixie on the host. The Pi-hole LXC in Chapter 8 deliberately uses Debian 12 Bookworm until future Pi-hole-on-Trixie compatibility and stability are confirmed for this beginner stack. Do not upgrade the Pi-hole LXC to Trixie unless a future edition tells you to.

Do not copy MAC addresses from examples. Proxmox generates MAC addresses for VMs and LXCs. Only IP addresses, VM IDs, CT IDs, hostnames, ports, and storage paths are part of this book's fixed design.

Before you begin the build, use the reference sheets in Appendix A. They collect the IDs, IP addresses, DNS names, ports, build paths, hardware values, and worksheet fields that you will refer back to throughout the chapters.

About the **optional Config Pack**: This book teaches the complete build sequence, architecture, settings, commands, and verification checks. Some long Docker Compose files, environment templates, and helper scripts are provided separately in the optional Nova X Lab Config Pack, available from store.novaxlab.eu. The Config Pack is a time-saving shortcut for readers who want ready-to-edit files instead of creating them manually. When a chapter uses a Config Pack shortcut, the chapter explains what the file does, which values must be reviewed, and how to verify the result.

The Fifteen Services You Will Run

This book builds a practical 15-service homelab: DNS, reverse proxy, container management, files, photos, passwords, media, smart home, monitoring, backups, notifications, dashboarding, remote access, and local PDF tools. The numbered list below gives the basic purpose of each service and the chapter where it is built. Optional Config Pack file references are collected in Appendix B.

1. Portainer (Chapter 7) — web interface for managing Docker containers and stacks.
2. Diun (Chapter 7) — Docker image update notifier that replaces Watchtower for this stack.
3. Pi-hole (Chapter 8) — network-wide DNS and ad blocking for every device on your home network.
4. NGINX Proxy Manager (Chapter 9) — reverse proxy and HTTPS front door for local services.
5. Jellyfin (Chapter 10) — personal media server with Intel iGPU hardware transcoding.
6. Nextcloud (Chapter 11) — self-hosted file sync, personal cloud storage, calendar, and contacts.
7. Immich (Chapter 11) — private photo and video backup with local machine-learning features.
8. Vaultwarden (Chapter 12) — Bitwarden-compatible password manager hosted on your own server.
9. Uptime Kuma (Chapter 12) — uptime monitoring and alerting for your homelab services.
10. Homepage (Chapter 12) — dashboard landing page for all homelab services.
11. ntfy (Chapter 12) — self-hosted push notifications for alerts and scripts.
12. Stirling-PDF (Chapter 12) — local PDF merge, split, compress, and OCR tools.
13. Home Assistant (Chapter 13) — local smart home hub for devices, integrations, and automations.
14. Proxmox Backup Server (Chapter 14) — encrypted, deduplicated backups for VMs and containers.
15. Tailscale (Chapter 16) — secure remote access VPN without opening router ports.

CHAPTER 1

Why Proxmox? The Case for Running Your Own Server

Something changed for a lot of people in 2024. Broadcom acquired VMware and immediately restructured licensing in a way that sent subscription costs sharply upward for businesses of every size. IT departments began migrating to alternatives. And in parallel, in living rooms and spare bedrooms, a different but related shift was happening: people who had been running their files on Google Drive, their photos on iCloud, their passwords in LastPass, and their media through Plex were starting to ask what it would take to run those things themselves.

The answer, for a growing number of them, was Proxmox VE running on a small, quiet, inexpensive mini PC.

This book is a practical guide to building exactly that. By the end of it, you will have a real home server running fifteen production-quality self-hosted services — a personal cloud for files, photos, passwords, media, home automation, DNS, monitoring, backups, and remote access — on hardware that fits in a drawer and costs less than a year of the subscriptions it replaces.

But before the first command, it is worth understanding what Proxmox VE actually is, why it is the right foundation for a home server in 2026, and what you are committing to when you build one.

What Proxmox VE Is

Proxmox VE is a Type 1 hypervisor. That means it runs directly on the hardware, not inside another operating system. When you install Proxmox VE on a mini PC, the mini PC becomes a server that can run multiple isolated operating systems simultaneously — each one in its own virtual machine, each believing it is the only system on the hardware.

If you have used VMware Workstation or VirtualBox on a Windows or Mac desktop, those are Type 2 hypervisors — they run on top of an existing OS. Proxmox VE is different. There is no Windows or Mac underneath it. The hardware boots directly into Proxmox, and everything else runs on top of that.

Proxmox VE is open source, built on Debian Linux, and free to use without a subscription. It has been in active development since 2008 and as of 2025 powers

more than 1.5 million deployments globally. The homelab community has grown around it specifically because it gives serious infrastructure capability — the same virtualization technology used in enterprise data centers — to anyone willing to spend an afternoon learning how it works.

PVE 8 vs PVE 9. This book targets Proxmox VE 9.x, which is based on Debian 13 (Trixie). PVE 9 ships with an updated installer, deb822-style repository files, a newer Linux kernel generation, and OpenZFS 2.3.x. If you are running PVE 8.x, the majority of this guide applies — differences are noted where they matter for a beginner.

Two Types of Container, One Platform

Proxmox VE can run two distinct types of isolated environment: virtual machines (VMs) and LXC containers. Understanding the difference is useful before you build anything, because you will use both — and the decision about which to use for each service is one of the first real choices the book asks you to make.

Virtual machines use KVM (Kernel-based Virtual Machine) to fully virtualize hardware. Each VM gets its own kernel, its own CPU allocation, its own RAM, its own virtual disk. A VM does not share a kernel with the Proxmox host or with other VMs. This isolation makes VMs the right choice for anything that needs its own operating system, has complex hardware requirements, or where strong security isolation matters — Home Assistant OS, the Docker host that runs most of your services, and Proxmox Backup Server all run as VMs in this book.

LXC containers share the Proxmox host kernel but run in their own isolated namespace. They start faster, use less RAM, and are more efficient than VMs for lightweight services — but they do not have the same level of isolation. Pi-hole runs as an LXC container in this book because it is a simple, well-understood service that benefits from the lower overhead.

A useful rule of thumb: if a service has an official operating system image (like Home Assistant OS), use a VM. If it is a single Linux service with modest resource requirements, an LXC container is often the cleaner choice. The book explains this decision in detail in Chapter 6.

The Architecture You Are Building

Every chapter of this book adds one layer to the same structure. By Chapter 16, you will have built a five-layer homelab that looks like this:

Layer	Role	Components
Layer 1	Infrastructure	Proxmox VE, ZFS storage, Linux bridge networking
Layer 2	Compute	Docker host VM, Home Assistant VM, PBS VM, Pi-hole LXC
Layer 3	Services	Jellyfin, Nextcloud, Immich, Vaultwarden, Home Assistant, Pi-hole, NGINX Proxy Manager, Portainer, Uptime Kuma
Layer 4	Access	NGINX Proxy Manager for HTTPS and subdomains; Pi-hole for local DNS; Tailscale for remote access
Layer 5	Security and backup	Proxmox Backup Server, rclone offsite, Uptime Kuma monitoring, SSH hardening, PVE firewall

This model is the spine of the book. Each chapter builds one part of it. Chapter 16 returns to this diagram as a checklist — by that point, every row should be filled in on your own hardware.

The Hardware This Book Is Written For

The primary hardware target for this book is an Intel N100 mini PC — specifically the Beelink S12 Pro, Minisforum UN100, and similar units that have established themselves as the default homelab starter platform as of 2025–2026.

The N100 is a 2023-generation Intel processor designed for low-power continuous operation. It draws 6–10 watts under typical homelab load. It has four efficient cores, 16–32 GB of DDR5 RAM depending on configuration, an integrated Intel GPU suitable for hardware video transcoding, and a single NVMe slot for storage. A new unit at the recommended specification is widely available on Amazon and Newegg — search "Beelink EQ12" or "Beelink S12 Pro" and "Minisforum UN100" for current listings. Prices change frequently so search current listings rather than relying on any figure printed here.

Everything in this book has been written for Intel N-series mini PCs, especially N100 and N150 systems. You must still verify device names, storage pools, IP addresses, BIOS settings, and credentials against your own hardware.

If you have a different machine. This guide works on any x86-64 hardware that meets Proxmox VE's requirements: a 64-bit CPU with hardware virtualization (Intel VT-x or AMD-V), at least 8 GB RAM, and at least 64 GB storage. Older desktops and small-form-factor workstations are common alternatives. Chapter 2 covers hardware

selection in detail, including what to check before you start and where single-CPU older machines differ from the N100 setup.

What You Need to Know Before Starting

This book assumes you are comfortable with a few things. Not expert — comfortable. If you can open a terminal and type commands without anxiety, you have enough.

Linux command line basics. You should be able to navigate directories (cd, ls), edit a file (nano is fine), copy and move files, and run a command with sudo. You do not need to know bash scripting. Every command in this book is explained before it is run.

Home networking fundamentals. You should know what an IP address is, what a router does, and roughly how DNS works. You do not need to know subnetting in detail — Chapter 3 covers everything you need for the networking steps in this guide.

SSH. You should be able to open an SSH connection from your laptop or desktop to another machine. If you have done this before, even once, you are set. If you have not, Windows 10 and 11 include a built-in SSH client — open PowerShell or Windows Terminal and type ssh root@[ip-address]. Mac and Linux include SSH by default.

What you do not need: any previous Proxmox experience, Docker experience, or Linux server administration background. Those things are built up progressively from Chapter 2 onwards.

A note on Windows users. Chapter 2 covers the first SSH connection step by step. If you have never opened a terminal before, that is fine — it is one command from PowerShell.

How This Book Is Structured

Each chapter ends with a Lab Checkpoint — a specific, testable statement of where your homelab should be at that point. The checkpoints are written so you can verify them before moving on. If the checkpoint condition is not true, the chapter has something that needs resolving before the next chapter builds on it.

Chapters 1–6 build the foundation: Proxmox is installed, networking is configured, the storage layout is set up, and you understand VMs and LXC containers well enough to make your own decisions about what runs where.

Chapters 7–12 deploy the main service stack: Docker host, Pi-hole, reverse proxy, Jellyfin, Nextcloud, Immich, Vaultwarden, Uptime Kuma, Diun, Homepage, ntfy, and Stirling-PDF.

Chapters 13–14 deploy the specialist VMs: Home Assistant OS and Proxmox Backup Server, including the offsite backup job that turns the backup strategy into a genuine 3-2-1.

Chapter 15 hardens the complete stack and reviews the full architecture as a checklist.

Chapter 16 adds remote access via Tailscale — the service that makes the entire homelab accessible from anywhere in the world.

> **LAB CHECKPOINT** - Chapter 1 checkpoint: You understand what Proxmox VE is and why it is appropriate for homelab use. You have confirmed your hardware meets the minimum requirements (64-bit CPU with VT-x or AMD-V, 8 GB+ RAM, 64 GB+ storage). You are ready to install.

CHAPTER 2

Before You Start — Hardware, Storage, and Setup

Chapter 1 described what you are building. This chapter covers what you need before you build it: the right hardware, the right storage configuration, and a short list of accounts and items to have ready. Getting these choices right before touching a keyboard saves time and avoids the most common dead ends.

Use the reference sheets before you build

The appendices are not optional reading. They keep the build chapters lighter, but you should use them before installing Proxmox so the IP addresses, storage choices, DNS names, and hardware values are already decided.

Reference	Use it for	Use before
Appendix A.1 — Quick Reference	Service IPs, ports, DNS names, VM/CT IDs, and routes.	Configuring services, NPM, Pi-hole, Tailscale, and troubleshooting.
Appendix A.2 — Build Paths and Dependencies	Choosing Starter, Privacy, Security, Smart Home, or Full Build.	Deciding which chapters to follow.
Appendix A.3 — Numbering, Networking, Resources, Storage	ID ranges, IP ranges, startup order, RAM/disk sizing, and storage layout.	Creating VMs/LXCs and assigning storage.
Appendix A.4 — Worksheets	Your own router IP, subnet, usernames, domains, ports, disk names, and hardware values.	Starting Chapter 2 and before running any command with a device path or IP.
Appendix B — Optional Companion Config Pack	Optional ready-to-edit scripts and configuration files.	Only if you purchased the separate Config Pack.

Pre-build checklist — do this before Chapter 2

Before installing anything, confirm the items below. This prevents the most common stop-start frustration in a first homelab build.

Required before	Have ready	Why it matters
Chapter 2	Mini PC, wired ethernet, USB installer, keyboard/monitor access, SSH key on your workstation	You need console access for the installer and recovery if networking is wrong.