

N9LYA TECHNICAL GUIDES

Build a Homelab

Design, Build, and Actually Run It

Jerome M. Kutche

N9LYA

Second Edition · September 2026

About This Book

There are two kinds of homelab books. The first is a shopping list with delusions of grandeur — buy this, plug that in, admire the blinking lights. The second is a 400-page enterprise architecture text that assumes you have a change advisory board and a budget line item.

This is neither. This is the book I wish someone had handed me before I spent money on hardware I didn't need, wired a rack I had to tear out six months later, and learned what a UPS runtime curve actually means at 2 a.m. during a thunderstorm.

A homelab is a working system you own, break, and repair on your own terms. It is the single best way to learn infrastructure, because nothing teaches you like being the person who has to fix it. But it is also a real system, drawing real power, holding real data, and connected to a real network with real people on the other side of it. Treat it that way and it will teach you more than any certification course. Treat it casually and it will teach you something anyway — usually about backups.

Who This Book Is For

You should read this if you are:

- **A complete beginner** who keeps hearing the word “homelab” and wants to know where to start without wasting a thousand dollars finding out.
- **An intermediate builder** running a NAS and a couple of containers, who suspects there is a more disciplined way to do this and is right.
- **An experienced engineer** who wants a lab that behaves like production — clustering, monitoring, backup verification, change control — without the corporate overhead.

The book moves in that order. Part I and II assume nothing. By Part VI you will be doing GPU passthrough and high availability. You can skip forward, but the chapters build on each other deliberately, and the expensive mistakes are mostly made in the early chapters.

What You Will Need

Nothing, to start. Chapter 2 is about deciding what you want before you spend a dollar, and that chapter alone will save most readers more than this book costs.

Eventually: a computer that can run a hypervisor, a network you control, somewhere to put it that doesn't make the household hate you, and a willingness to read logs.

A Note on Electricity

I spent thirty years working on industrial electrical and control systems — motor control centers, PLCs, switchgear, the kind of equipment where a mistake is not a reboot. I bring that up exactly once, here, because it colors how this book handles power.

You will find more attention paid to circuit loading, grounding, thermal behavior, and UPS sizing than you will in most homelab material. That is not padding. A homelab can create electrical and fire hazards through overloaded circuits, damaged cords, poor connections, blocked airflow, or equipment used outside its rating. Chapter 6 covers the planning boundaries. Please read it before you plug in a server.

Scope and safety: This book is educational and does not replace equipment instructions, applicable electrical or building codes, manufacturer environmental limits, or work by a qualified professional. Do not remove service-panel covers, modify fixed wiring, defeat protective devices, or work on energized equipment based on this book. If the circuit, grounding, heat, equipment condition, or required work is uncertain, stop and consult a licensed electrician or other qualified professional.

How to Use This Book

Each chapter ends with a **Bench Check** — a short set of questions or tasks that confirm you actually absorbed the chapter rather than just read it. Do them. The appendices contain build sheets by budget, an IP addressing template you can fill in, and a command reference you’ll come back to.

Where I recommend specific software I will say why, and I will name the alternatives. Where I recommend specific hardware I will give you the class of hardware and what to look for, not a model number that will be discontinued before this book is a year old.

About the Author

Jerome M. Kutche holds the amateur radio callsign N9LYA, first licensed in 1991. He spent over three decades in industrial electrical and controls work — Allen-Bradley and Siemens PLC systems, motor controls, plant electrical maintenance and supervision — across manufacturing, materials, and heavy industry. He holds NFPA 70E and lockout/tagout certifications among thirty-nine professional credentials.

He serves as Indiana’s 44Net/AMPRNet coordinator, administering the state’s allocation of the amateur radio IP address space, and is trustee for two amateur radio clubs. He operates a packet radio node, a fifty-panel solar array with custom monitoring, and a homelab built on enterprise hardware that has been rebuilt, re-racked, and re-thought more times than he cares to admit.

He wrote this book because the good information was scattered across a hundred forum threads and none of it was written by someone who had also been the person responsible when the power went out.

Part I — Foundations

Chapter 1: What a Homelab Actually Is

A homelab is a computing environment you own, operate, and are responsible for, built primarily to learn rather than to serve customers. That’s the whole definition. Everything else — the rack, the fiber, the three-node cluster, the ten-gigabit switch — is implementation detail.

This matters more than it sounds. The word “homelab” gets used for at least four different things, and people get into trouble by building one while thinking they’re building another.

The Four Things People Call a Homelab

The appliance. A single NAS or a Raspberry Pi running a handful of services. Plex, Pi-hole, maybe Home Assistant. It works, it’s useful, it requires almost no operational discipline. There is nothing wrong with this and many people never need more. But an appliance is a product you operate, not a lab you learn from. If you never open it up, you never learn what’s inside.

The learning rig. One reasonably capable machine running a hypervisor, where you spin up virtual machines to practice things: install Linux forty times, break Active Directory, configure a firewall, tear it all down. This is where most genuine learning happens, and it can be done on a used desktop for under three hundred dollars. If your goal is skills, start here and possibly stay here.

The self-hosted household. Services your family actually depends on — file storage, media, home automation, password management, DNS. The moment your spouse notices when it’s down, you have crossed from lab into production, and the rules change completely. Uptime matters. Backups matter. Change windows matter. Most people cross this line without noticing and then wonder why their hobby feels stressful.

The miniature datacenter. Multiple nodes, shared storage, clustering, redundant networking, real monitoring. This is where you learn the things that only appear at scale: quorum, failover, split-brain, the difference between a service being up and a service being available. It costs real money and real electricity.

One Rack, Two Risk Classes

Separate experiments from household production

LAB / EXPERIMENT

- Break it deliberately
- Frequent rebuilds
- Short retention
- Learning is the outcome



HOUSEHOLD PRODUCTION

- Backups verified
- Changes scheduled
- Dependencies documented
- Recovery is the outcome

Do not share the only copy, disk, or recovery path.

N9LYA TECHNICAL GUIDES

Figure 1: Separating experiments from household production

Most serious homelabs are two or three of these at once, running on the same hardware. That’s fine, but be honest about which parts are which. The family photo archive is production. The Kubernetes cluster you rebuild every weekend is a lab. Do not put them on the same disk without a plan.

Why Build One

The honest reasons, roughly in order of how well they hold up:

Because reading doesn’t work. You can read about VLANs for a year and still not understand them. Configure one, misconfigure it, lock yourself out of your own switch, and drive to the closet with a console cable — you will understand VLANs permanently. Infrastructure knowledge is procedural, not declarative. It lives in your hands.

Because the failures are the curriculum. In a lab, a failure costs you an evening. In production, it costs you a job. The whole point of a lab is to accumulate failure experience cheaply. A homelabber who has recovered from a corrupted array, a failed upgrade, a bad certificate, and a network loop is more valuable than one who has read about all four.

Because you own it. Cloud services change terms, raise prices, discontinue features, and read your data. Self-hosting is not always cheaper and is never easier, but the thing you run is the thing you control.

Because it’s genuinely fun. This is a legitimate reason and I’m not going to pretend otherwise. Some people restore cars. This is that, with better documentation and worse ergonomics.

The reason that does not hold up: “it’ll save money.” It won’t, at least not for years, and probably not at all once you count electricity and your own time at any nonzero hourly rate. Build a homelab because you want the knowledge or the control. If you build one to save fifteen dollars a month on cloud storage, you will be disappointed and you will have a rack in your basement.

What a Homelab Is Not

It is not a production datacenter, no matter how much it looks like one. You have no redundant utility feed, no on-call rotation, no vendor support contract, and no second person who knows how it works. Design accordingly — which mostly means design for recovery rather than for never failing.

It is not free. Chapter 3 does the arithmetic properly, but a modest always-on lab drawing 150 watts costs somewhere between \$150 and \$250 a year in electricity depending on your rate. A rack of enterprise gear can easily triple that.

It is not a substitute for backups of things you care about. A RAID array is not a backup. A second copy in the same building is not a backup. We will return to this until you’re sick of it, because it is the single most common catastrophic mistake in this hobby.

The Progression

Here is the honest arc most people follow, and roughly what each stage teaches:

Stage	Typical Build	What You Learn
Curiosity	Raspberry Pi, one or two services	Linux basics, SSH, services and ports
Foundation	Used desktop or mini PC, hypervisor	Virtualization, snapshots, storage basics
Expansion	Dedicated NAS, managed switch	Storage design, VLANs, DNS, reverse proxy
Consolidation	Rack, UPS, real backup strategy	Power, cooling, backup verification, monitoring
Maturity	Multi-node, clustering, automation	High availability, IaC, change discipline

Most people accelerate through the first two stages and then stall between Expansion and Consolidation — the point where the lab has become genuinely useful, genuinely complicated, and genuinely undocumented. That stall is what Part V of this book exists to fix.

Bench Check

1. Which of the four categories best describes what you want to build? Write it down.
2. Is there anything in your planned lab that a family member would notice being down? If yes, that part is production. Name it.
3. What is one thing you currently pay a cloud provider for that you would genuinely rather run yourself — and why?

Chapter 2: Mission Before Metal

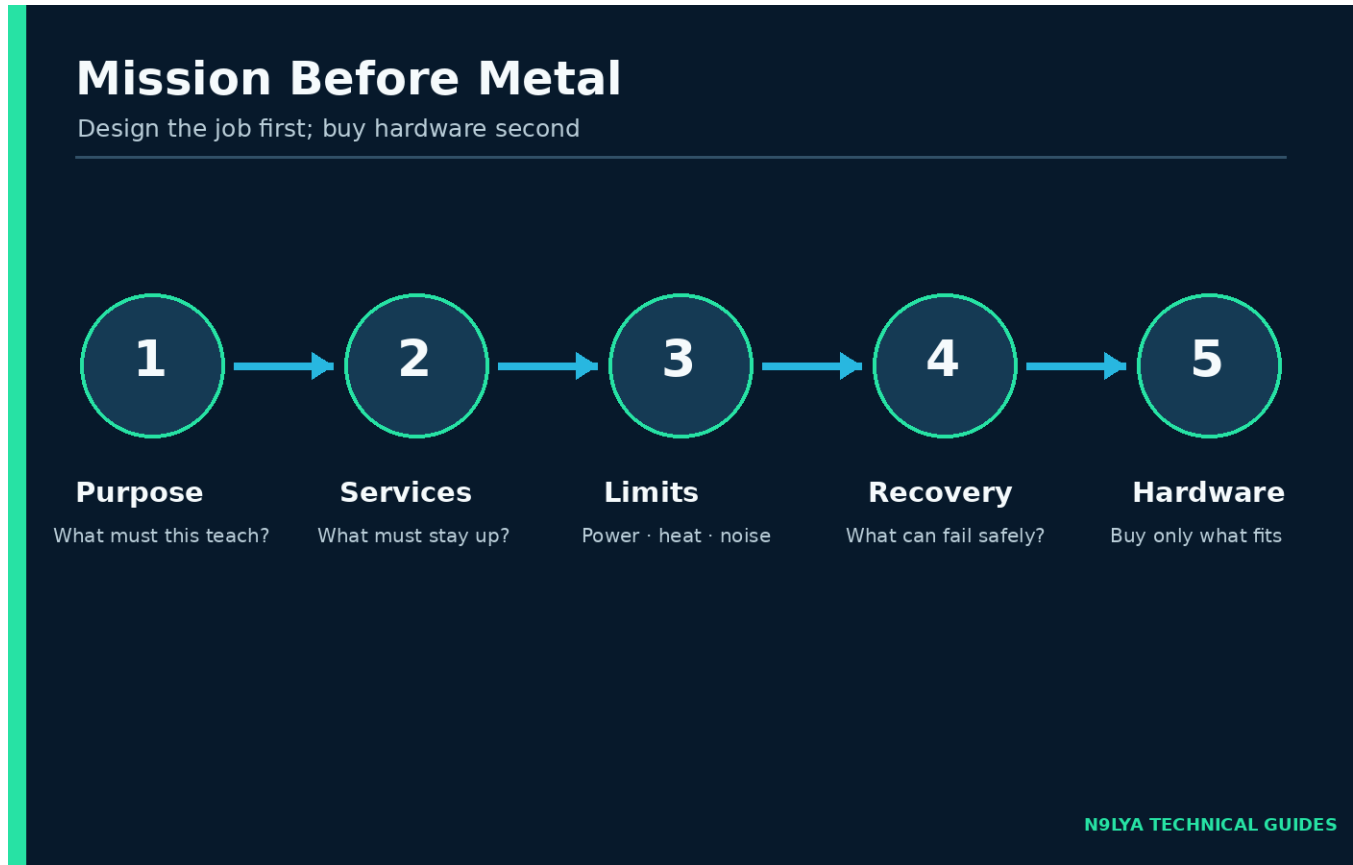


Figure 2: Mission-before-metal planning sequence

The most expensive mistake in this hobby is buying hardware before deciding what it's for. It is also the most common, because buying hardware is fun and deciding what it's for is not.

I have watched people spend eighteen hundred dollars on a dual-socket server to run a Plex instance that a five-year-old mini PC would have handled without effort. I have watched other people try to run a twelve-VM lab on a Raspberry Pi and conclude that virtualization is unreliable. Both errors have the same root cause: no defined mission.

Write the Mission Statement

Before you spend anything, write down — actually write, not think — the answer to these questions:

What do I want to be able to do in twelve months that I cannot do now?

Be specific. “Learn Linux” is not a mission. “Be able to build, break, and rebuild a three-node cluster with shared storage” is a mission. “Host my family’s photos with off-site backup and be able to prove I can restore them” is a mission.

What services will run continuously, and who depends on them?

This list defines your uptime requirements and therefore your power, cooling, and redundancy budget. A lab with zero always-on services can live on a machine you turn on when you use it. A lab hosting household DNS cannot go down without the house noticing within about ninety seconds.

What am I willing to spend, up front and monthly?

Two numbers. The up-front number is hardware. The monthly number is electricity plus any subscriptions or domain costs. People plan the first and ignore the second, which is how a “free” self-hosting project quietly becomes a twenty-five dollar a month habit.

What is my noise and space tolerance?

This kills more homelabs than budget does. A 1U rack server at full fan speed is roughly as loud as a vacuum cleaner. If the only place you have is a bedroom closet, that decision has already been made for you, and it eliminates most used enterprise gear from consideration. Decide now, not after the pallet arrives.

How much of my time is this worth per week?

An honest answer here changes everything. Four hours a week supports a genuinely ambitious lab. One hour a week supports a small, stable, well-documented one — and you should build accordingly, favoring boring reliable choices over interesting fragile ones.

Common Missions and What They Actually Require

“I want to learn enough to change careers.” You need virtualization and networking, not storage capacity. A single machine with 32 GB of RAM and a managed switch beats a big NAS every time. Prioritize RAM and the ability to run many VMs simultaneously. Budget: \$400–800.

“I want to get out of the cloud.” You need storage, backup, and remote access — in that order, and the backup part is non-negotiable because you have just made yourself the only thing standing between your family and data loss. Prioritize drive capacity, a second physical copy, and an off-site target. Budget: \$700–1,500 plus ongoing off-site cost.

“I want to run home automation and media reliably.” You need low power, high uptime, and quiet. This is the strongest case for mini PCs and against used enterprise servers. Prioritize efficiency and silence. Budget: \$300–700.

“I want to build something that behaves like production infrastructure.” You need multiple nodes, a real network, and the discipline chapters in Part V. This is the most expensive path and the most educational. Prioritize node count over per-node power — three modest machines teach you far more than one powerful one. Budget: \$1,200–3,000.

“I want to run local AI models.” You need GPU VRAM above all else, then system RAM, then everything else. This is a genuinely different build with genuinely different constraints, and Chapter 20 covers it specifically. Budget: highly variable; \$500 for modest inference on used datacenter cards, several thousand for anything ambitious.

The Requirements Table

Turn your mission into numbers before you shop. Fill in this table honestly:

Requirement	Your Answer
Always-on services	
Peak simultaneous VMs/containers	
Usable storage needed (year 1)	
Usable storage needed (year 3)	
Acceptable downtime for a planned change	
Acceptable downtime for a failure	
Maximum noise level	
Maximum monthly power cost	

Requirement	Your Answer
Up-front budget	
Hours per week available	

The two storage rows exist because storage growth is the most underestimated number in homelabbing. Whatever you think you need, you will use it faster than you expect, and expanding an array is much harder than building it right the first time. Chapter 5 goes into why.

The two downtime rows exist because they are different numbers and they drive different decisions. Tolerating four hours of planned downtime is easy. Tolerating four hours of unplanned downtime while you figure out what broke is a much bigger ask, and it's the one that pushes you toward redundancy.

The Scope Discipline

Here is the rule that will save you the most money and frustration:

Build the smallest thing that satisfies the mission, then expand when a real constraint bites.

Not when you imagine a constraint. Not when a forum post says you'll want more RAM eventually. When you actually run out.

This feels wrong to technical people, who like to build headroom. But headroom in a homelab has a specific cost — you pay for it in dollars up front and watts forever, and it frequently goes unused because your mission changed. The person who bought 128 GB of RAM “for later” and still uses 40 GB three years later spent several hundred dollars on a hunch.

The exception is anything genuinely painful to change later: chassis size, drive bay count, power supply capacity, and network cabling. Buy headroom in the physical layer, where expansion is disruptive. Buy exactly what you need in the compute layer, where expansion is a screwdriver and twenty minutes.

Writing It Down

Create a document. It can be a text file, a wiki page, a notebook — the format matters far less than its existence. Put in it:

1. The mission statement, in a sentence or two.
2. The requirements table above, filled in.
3. A list of services you intend to run, and for each, whether it is lab or production.
4. A “not doing” list — things you have explicitly decided are out of scope.

That last item is the one people skip and the one that does the most work. A written “not doing” list is what lets you read an exciting forum thread about Ceph and think “not this year” instead of ordering three more drives.

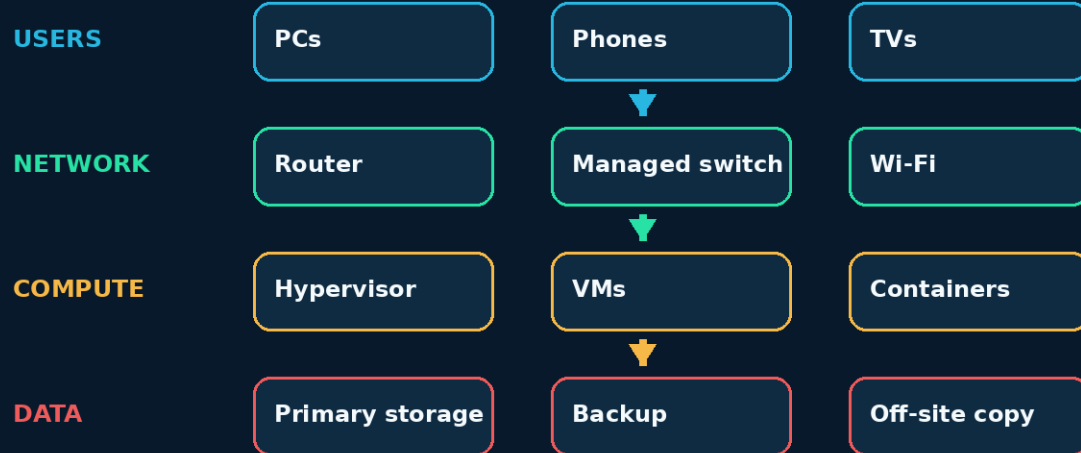
You will revise this document. That's expected and healthy. But revising a written plan is a decision; drifting without one is just accumulation.

Bench Check

1. Write your mission statement now, in two sentences or fewer.
2. Fill in the requirements table. If you can't answer a row, that's information — it means you need to research that dimension before buying.
3. Write three items on your “not doing” list.
4. For your top-priority service, answer: if it's down for a day, who cares and how much?

Practical Homelab Reference Architecture

Simple layers with clear recovery boundaries



N9LYA TECHNICAL GUIDES

Figure 3: Practical homelab reference architecture

Chapter 3: Budget, Power, Space, and the Household

Money is the constraint people plan for. Power, space, noise, and household tolerance are the constraints that actually end homelabs. This chapter covers all of them, with arithmetic.

The Real Cost of a Homelab

There are four cost categories, and most people only count the first.

Up-front hardware. The obvious one. Server or desktop, drives, switch, cables, rack or shelf, UPS.

Ongoing electricity. The one that surprises people. Calculate it before you buy, not after.

Consumables and replacement. Drives fail. Fans fail. Batteries in a UPS need replacing every three to five years and cost \$50–150. Budget roughly 10–15% of your hardware value per year for replacement over the long run.

Ancillary services. Domain registration (\$12–20/yr), off-site backup storage (\$3–7 per TB per month for object storage, more for consumer services), possibly a VPS for a jump host (\$5/mo). Small individually, real in aggregate.

Doing the Power Math

This is the calculation nobody runs and everybody should. It takes two minutes.

Annual cost = (watts ÷ 1000) × 8,760 hours × your rate per kWh

The 8,760 is hours in a year. Your rate is on your electric bill; the U.S. average sits somewhere around \$0.16/kWh at the time of writing, but it ranges from about \$0.11 to over \$0.30 depending on where you live. Use your actual number.

Worked examples at \$0.16/kWh:

Setup	Typical Draw	Annual Cost
Raspberry Pi 5	7 W	\$10
Mini PC (N100 class)	15 W	\$21
Small NAS, 4 drives	45 W	\$63
Used desktop as hypervisor	80 W	\$112
Tower server, moderate load	150 W	\$210
2U enterprise server (DL380 class)	200 W	\$280
Two-node cluster + switch + NAS	400 W	\$561
Full rack, several nodes	700 W	\$981

That last row is why people quietly decommission racks. A thousand dollars a year is a car payment.

Two things follow from this table. First, idle efficiency matters enormously for always-on gear — the difference between a 15 W mini PC and a 200 W server, run continuously, is about \$260 a year, every year. Second, gear you only power on when using it is nearly free to own, which is a strong argument for splitting your lab into an always-on tier and an on-demand tier. Chapter 4 develops that idea.

Measure, don't guess. A plug-in energy meter costs about \$25 and will tell you the truth about your actual draw, which is frequently lower than nameplate ratings and sometimes higher than forum estimates. Buy one before you buy a server.

Circuit Loading — Read This Part

A nominal U.S. 15-amp, 120-volt branch circuit has an arithmetic maximum of 1,800 volt-amperes, and a 20-amp circuit has 2,400. Continuous-load and conductor/protective-device sizing rules can require 125% sizing, commonly expressed as keeping the continuous portion at or below 80% of the applicable rating. The correct calculation depends on the actual circuit, connected loads, equipment

ratings, and locally adopted code. Treat 1,440 VA on a 15-amp circuit and 1,920 VA on a 20-amp circuit as planning ceilings for the continuous portion, not permission to fill an unknown shared circuit to that level.

That sounds like plenty. It usually is, for the lab itself. The problem is that you almost never have the circuit to yourself. The outlet in your basement office is very likely on the same branch as several other outlets, possibly a lighting circuit, possibly a freezer.

Practical guidance:

- **Find out what else is on the circuit.** Turn the breaker off and walk the house noting what died. Do this once; write it down.
- **Keep your lab under 50% of the circuit rating** if you're sharing it with anything else, especially anything with a motor. Compressors and pumps draw large inrush currents on startup that can trip a marginally loaded breaker.
- **Never run a homelab through a daisy chain of power strips.** Power strips are not rated for series connection, and the failure mode is a fire, not a trip.
- **Use one properly rated extension cord if you must, sized for the load and the length,** and understand that voltage drop over a long thin cord is real and causes power supplies to draw more current to compensate.
- **Consider a dedicated circuit when the lab is a large continuous load, shares a circuit with motor loads, or leaves inadequate margin.** Have a licensed electrician evaluate the existing circuit, panel capacity, grounding, and locally adopted requirements. Cost varies too widely by building and jurisdiction to estimate responsibly here.

A word on heat. Loose, damaged, or corroded connections can develop resistance and excessive heat under load. Unusual heating, discoloration, odor, arcing sounds, damaged insulation, or a plug or outlet that becomes progressively hotter is a stop condition, not something to monitor casually.

You may observe accessible equipment externally without opening a panel or removing guards, but temperature alone is not a diagnosis and emissivity can mislead infrared readings. If you find abnormal heat or damage, shut down the connected load if it is safe to do so, do not work inside energized equipment, and call a licensed electrician. Utility-side, line-side, and service-panel work is not homelab territory.

UPS Sizing

An uninterruptible power supply does three jobs: rides through brief outages, allows graceful shutdown during long ones, and conditions incoming power against sags and surges. The third job is the one people forget and the one that quietly protects your hardware.

Sizing. UPS units are rated in volt-amps (VA) and watts, and the two are not the same. For modern switching power supplies, assume a power factor of about 0.6 for cheap units and 0.9 for better ones — meaning a 1500 VA unit may only deliver 900-1350 watts. **Size on the watt rating, not the VA number on the box.**

Target 50-70% loading of the watt rating. Above that, runtime collapses and battery life shortens dramatically. Below about 30%, you're paying for capacity you don't use.

Runtime. Do not expect to ride out a long outage. A typical 1500 VA unit at 300 watts of load gives you roughly fifteen to twenty-five minutes. That is not "keep working through the storm" time — it is "shut down cleanly" time, and that is what you should design for.

Topology matters if your power is dirty. Standby ("offline") units switch to battery only when power fails and do nothing about voltage variation. Line-interactive units add automatic voltage regulation, correcting sags and swells without touching the battery — this is the right choice for most homelabs and the one to buy if your utility power sags when the air conditioner starts. Double-conversion (online) units continuously regenerate the output waveform and are the best protection, at higher cost and higher idle power draw.

Signal your servers. A UPS that isn't connected to your machines by USB or network is only half a UPS. Configure automatic graceful shutdown — Network UPS Tools (NUT) on Linux is the standard

answer and is well supported. An unsigaled UPS just delays the hard crash by twenty minutes.

Test annually. Pull the plug on purpose, with someone watching, and confirm the shutdown sequence actually runs. Batteries fail silently and a UPS with a dead battery is a very expensive power strip.

Space, Heat, and Noise

Heat. Nearly every watt your lab consumes eventually becomes heat in the room. A 400-watt continuous load adds 400 watts of heat. The resulting temperature rise depends on room volume, insulation, airflow, and HVAC; do not rely on a universal degree-rise estimate. Follow each manufacturer’s published inlet-temperature and airflow limits, measure at the equipment intake, and leave operating margin rather than designing to the maximum rating.

Ventilation is not optional. At minimum, a closet door with a louvered vent or a gap at top and bottom to allow convection. Better, a small exhaust fan. Best, a room with actual HVAC.

In summer, a homelab makes your air conditioner work harder, which is a second electricity cost most people never account for. In winter, it’s free heat, which is genuinely a small consolation.

Noise. Measured roughly, in dBA:

Equipment	Approximate Noise
Mini PC, fanless	Silent
Consumer NAS	25-35 dBA
Tower server, low load	35-45 dBA
Desktop with quality fans	30-40 dBA
2U rack server, idle	50-60 dBA
1U rack server, under load	65-80 dBA

For reference, normal conversation is about 60 dBA. A 1U server at full fan speed is not something you can be in a room with. This is the single most common reason people sell enterprise gear at a loss three months after buying it.

If your only available space is living space, buy quiet hardware. There is no amount of fan-curve modification that makes a 1U chassis pleasant, because the fans are small, fast, and physically required to move that much air through that little space.

Space. Plan for more than the equipment footprint. You need clearance in front for airflow and access, clearance behind for cabling, and somewhere to put a monitor and keyboard when things go wrong. A rack needs about three feet of working room in front of it. A shelf build needs somewhere to stage a drive swap.

The Household Factor

This is the section people skip and it is the most predictive of whether your lab survives.

Do not make the household’s internet depend on your experiments. If your homelab runs DNS or routing for the whole house, every maintenance window is a household outage, and every mistake at 11 p.m. is a conversation. There are two good answers: either build a separate lab network that can break freely, or accept that your infrastructure is now production and adopt real change discipline. Chapter 9 covers the segmentation approach and Chapter 19 covers the discipline approach. Do one of them.

Be honest about the noise and the heat before you buy. “You won’t even notice it” is a claim you should verify with a decibel meter app before it becomes a claim you have to defend.

Make it visibly useful. A lab that provides something the household values — reliable media, working photo backup, an ad-free network — is a lab that survives budget conversations. A lab that only benefits you is a hobby that competes with every other hobby for money and space.

End of the free sample

Build a Homelab — Second Edition · by Jerome M. Kutche

That was the first 12 of 80 pages. The full book continues with the practical detail, worked examples and recovery steps that don't fit in a sample.

Get the full book (\$9.99, PDF + EPUB):

<https://booksrus.biz/product/build-a-homelab/>

Also on Gumroad:

<https://n9lya4.gumroad.com/l/wvmmdl>



All 17 Second Edition guides in one bundle for \$89.99:

booksrus.biz/product/n9lya-technical-guides-complete-bundle

This sample may be shared freely in unmodified form. © Jerome M. Kutche. Second Edition.