Episode Book

EPISODE #5905

Bare Linux Routing vs OPNsense: What Actually Wins?

PUBLISHED	12 October 2026 at 13:30 UTC
DURATION	25:18
LISTEN	myweirdprompts.com/episode/bare-linux-routing-vs-opnsense-40090/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.4
SOURCES	16 cited by the research (listed at the end)
BOOK GENERATED	19 October 2026 at 17:41 UTC

SYNOPSIS

OPNsense is a convenience, not a requirement — you can turn a cheap two-NIC mini-PC running Debian or Ubuntu into a full router with systemd-networkd, nftables, dnsmasq and WireGuard. Some people have done exactly that and never looked back: Ansible-provisioned configs, six hundred days of uptime, and patches straight from upstream instead of waiting on a handful of downstream maintainers. Others argue the sticker price of a router is only ten to twenty percent of its total cost of ownership, and the other eighty percent is the decade of maintenance nobody budgets for. This episode works through both sides — the patch-latency argument, the PPPoE performance case, the composability dream — and asks whether a "composable router" actually exists yet, or whether the closest thing to it is still in beta.

TRANSCRIPT



Corn

Most people go one way into this hobby and they never come back out. You buy a mini-PC with four NICs, you flash OPNsense onto it bare metal, you click through the setup wizard, and then you forget it exists until the internet stops working at seven in the morning.



Herman

Which is the correct way to do it, by the way.



Corn

Daniel did exactly that, and then he had to rebuild the config from scratch, and somewhere in hour two of that, a thought arrived. Why am I installing a routing distribution at all? Why not just put Ubuntu or Debian on the box?



Herman

The honest answer being that you can. Easily.



Corn

He took it to ChatGPT, which confirmed the same thing in about four seconds, and the only downside it offered was the general downside of reinventing the wheel. So Daniel pushed back, and this is the interesting part. He asked it why he wanted this in the first place. Almost everything in OPNsense is a feature he has never touched and never will. The two things he actually needs — failover planning and VPN configuration — are the two things that are hardest to get working in OPNsense. And the AI, to his surprise, agreed with him that building his own might make sense.



Herman

Then it agreed itself right out of a project, because what killed it was downtime.



Corn

His home internet would be down for however long the build took, and that was the end of it. So it stays an objective. His phrase was that a composable router would be a super cool idea. Take what you need — DNS, ad blocking, failover — and keep the base operating system very lean. His questions are whether any tried-and-tested recipe exists for that, and in environments with dedicated staff, how often teams actually roll their own Linux box for internal routing instead of buying something.



Herman

Those are two different questions with two very different answers, which is what makes it a good episode.



Corn

So let's find out whether bare Linux routing is real, and whether composable is a thing that exists yet.



Herman

Bare Linux routing is real, and it is older than most of the people listening. This is not a fringe technique. The core mechanism is almost boring. You enable IP forwarding in the kernel, one line in a config file, you configure your interfaces, you write a NAT rule and a filter policy in nftables, you run a DHCP server and a resolver, and you have a router. That's the whole thing.



Corn

That's fewer steps than getting one VLAN working in a web UI.



Herman

There are step-by-step guides going back years. Linux Junkies has one, Big Iron has a full Debian walkthrough with nftables and dnsmasq and IPv6 prefix delegation, there's Matt Ridpath's write-up, there's a really good one on nbailey.ca. The modern canonical stack is systemd-networkd for the interfaces, nftables for filtering and NAT, dnsmasq for DHCP and local DNS, Unbound or Stubby if you want DNS over TLS, and WireGuard for the tunnel. Most single-router setups can be covered with just systemd-networkd and nftables.



Corn

So the answer to the literal question is yes, and it has been yes for a long time. A dedicated routing distribution is a convenience, not a requirement.



Herman

Exactly that. OPNsense is not the thing that makes routing possible. It's the thing that makes routing clickable.



Corn

Which is a real product. I want to be fair to it. Somebody maintained that web UI so that people who don't want to touch a config file never have to.



Herman

Of course. And the real question Daniel is circling is not can you, it's should you. And that splits very cleanly into two costs that behave completely differently. The setup cost, which AI is collapsing right now, and the maintenance cost, which nothing is collapsing.



Corn

So the build is cheap and the keeping-it-alive is expensive.



Herman

That's the entire tension of the episode, and everything else is detail.



Corn

Then let's do the build-versus-buy case properly. Start with the people who actually did it, because there's a decent amount of them.



Herman

There is, and the reports are more positive than you'd expect. Matt Ridpath runs Ubuntu 24.04 on a two-NIC mini-PC that cost him about a hundred dollars. Three VLANs — internal, lab, guest — providing DHCP, DNS and LDAP on top. And he says outright that he prefers it to OPNsense, for one reason: he can manage the entire business from the command line with Ansible. When the PC dies, he reinstalls Ubuntu, re-runs the playbook, and he's back.



Corn

A hundred dollars and the disaster recovery plan is a reinstall.



Herman

That's the reproducibility argument and it's a strong one. Everything is a file, everything is version-controlled, and there's no clicking through a wizard to reconstruct what you did. There was a Hacker News thread a few years back where a commenter called Arnavion switched from OPNsense to Debian 12 and called it rock solid — nftables through firewalld, unbound, systemd-networkd, plus haproxy running on the same box in per-service network namespaces. Another one, lazyweb, had a Debian 11 router at about six hundred days of uptime, pulling twenty to thirty watts idle, provisioned entirely by Ansible.



Corn

Six hundred days. That's a machine that has outlived several of the things I've been told were replacements for it.



Herman

The service colocation point is underrated. On an appliance you get a router. On a Linux box you get a router that can also run containers, a reverse proxy, your monitoring stack, whatever you want, and they can be isolated from each other with namespaces.



Corn

Now give me the argument that actually converts people.



Herman

It's the patches. Arnavion's reason for leaving OPNsense was blunt. He said it's insecure and its maintainers are incompetent, and that downstreams like pfSense and OPNsense with only a handful of maintainers just delay patches pointlessly. He'd rather use upstream directly.



Corn

That's the sharpest technical case on the board, and it has nothing to do with features. It's about who ships you a fix.



Herman

Right. When there's a CVE in nftables or OpenSSL, Debian ships it on Debian's schedule, which is fast and staffed. With a small downstream project, you're waiting on a handful of volunteers to notice, backport, test and release. You're adding a hop between the vulnerability and the fix, and the hop is the least resourced part of the chain.



Corn

That's a real cost that never shows up in a feature comparison. Nobody puts patch latency on the marketing page.



Herman

There's also a raw performance case. A commenter going by bwj982 found pfSense was giving him ten percent of his WAN bandwidth because of how its PPPoE daemon threads. Ten percent. He moved to Debian with nftables and the problem went away.



Corn

Ten percent of a fiber line is a rounding error with a support forum.



Herman

That one's specific to PPPoE, which is a lot of connections in Europe, and it's the kind of thing that's invisible until you measure it. Now, the other side.



Corn

The other side is heavier.



Herman

Considerably. Ivan Pepelnjak at ipSpace put it flatly back in twenty twenty-one. Unless you have a very special use case, it's probably easier and cheaper in the long run to buy a router than to build it from open-source components. And he's not a vendor guy, he's a networking instructor who has watched a lot of people try this.



Corn

And the reason it's cheaper isn't the hardware. It's the ten years after the hardware.



Herman

That's the TCO argument and it's the one that actually kills the idea. A commenter on that same piece, Bela Varkonyi, made the point that the procurement cost of a router — the box, the license, all of it — is usually only ten to twenty percent of the total cost of ownership. The other eighty to ninety percent is deployment, testing, documentation, operations, training, incident management.



Corn

So the sticker price is the smallest part of the bill and it's the only part people compare.



Herman

Tony P put it better than I can. It's really easy to build a router. It's just awfully hard to build a scalable, high capacity, secure one at good cost, and it's even harder to maintain it for the ten-plus years it's in your infrastructure.



Corn

That's the sentence Daniel needs to hear, and it's also the sentence he's already half-argued against.



Herman

How so?



Corn

Because his instinct is I only need five features, so the wheel is overkill. And on the features, he's right. The evidence agrees with him. Most people use a fraction of OPNsense. But the thing he's calling the wheel isn't the feature list. It's the maintenance wheel. Somebody has to keep those five features working for a decade, across kernel changes and library changes and a CVE every few months.



Herman

That's the reframe. You're not avoiding the wheel by needing less. You're just choosing a smaller wheel that you now have to turn yourself.



Corn

And I'll say the part that makes me nervous, because it's the part I'd be nervous about. A Linux router is a machine that can change while you're asleep. Every package upgrade, every kernel bump, every nftables syntax change between versions, and one of them eventually drops your WAN.



Herman

Which is precisely Daniel's downtime fear. He didn't fail to build it. He decided the window where his household has no internet wasn't worth the payoff, and that's not cowardice, that's correct accounting.



Corn

There's one more framing worth putting on the table. The OPNsense forum's own position, from a thread a while back: OPNsense is built on the pf packet filter, which is BSD only. You can absolutely build a Linux firewall, but it wouldn't be OPNsense. OpenWRT and IPFire exist.



Herman

That's honest of them, and it's useful, because it reframes the whole question. You're not replacing OPNsense with a cheaper OPNsense. You're choosing a different lineage. BSD and pf on one side, Linux and nftables on the other. And when you put it that way, the question stops being is there a shortcut and becomes which family do I want to live in.



Corn

Fine. So the pros are real, the cons are heavier than they look, and the thing he actually wants is a composable router. Does that exist, or is he describing something nobody has built?



Herman

Under that exact name, no. There's no mature composable router product you can go and install as of today. But there is one project that is almost eerily close to exactly what he described.



Corn

Go.



Herman

It's called MurOS. It's an OPNsense fork that's been re-engineered to run natively on Debian 13. They keep the OPNsense web UI and the OPNsense config model, and they swap out the entire FreeBSD layer underneath it. pf becomes nftables. The pf state table becomes netfilter conntrack. The rc init scripts become systemd. ifconfig and route become iproute2 and systemd-networkd. The package manager becomes apt and dpkg.



Corn

So it's OPNsense with the BSD pulled out.



Herman

Literally that. And the README is careful about what it is. The goal is not a temporary compatibility shim. Each subsystem is ported in place so MurOS behaves like a first-class Linux firewall rather than a FreeBSD appliance running elsewhere.



Corn

That's a good sentence, and it's also a promise, not a shipped product.



Herman

Correct, and here's the honest status. It's beta. Stateful filtering, NAT, routing, multi-WAN gateway monitoring, DHCP client on the WAN, the live firewall log, packet capture and serial console all run natively today. What's still being ported is the VPN service lifecycle, high availability, and the DHCP and recursive DNS backends. So the two things Daniel actually cares about — failover and VPNs — are the two that aren't finished.



Corn

The exact features he wanted are the exact features that aren't done.



Herman

Which is either bad luck or a law of nature. It installs from an ISO, or you curl a script onto an existing Debian 13 box and run it. BSD two-clause license, and it's not affiliated with Deciso, the company behind OPNsense, so nobody's blessing it officially.



Corn

Nobody with a legal department, anyway.



Herman

The interesting thing is MurOS isn't the composable router. It's the opposite of composable, in a way. It's the whole appliance experience but on a Linux base. What makes it relevant to Daniel is that it proves the base layer was never the obstacle. If somebody can port the entire thing to Debian in their spare time, then your five features on a lean Debian install were never the hard part.



Corn

So the pieces exist, they're just not assembled into one product. What are the pieces?



Herman

There's a whole collection. Petrouz has a set of Ansible roles for Debian 13 and 14 firewalls that is production-oriented — nftables, WireGuard, StrongSwan for IPsec, FRRouting for OSPF and BGP, Keepalived for VRRP high availability, conntrackd for state sync. His framing is firewall defined as code.



Corn

That's the closest thing to an actual recipe on the list.



Herman

Then kalken's ezrouter, which advertises turning any systemd distribution into a full router with no dependencies beyond nftables and systemd itself. rgregor has a Debian 12 router playbook with nftables, AdGuard Home, systemd-networkd, dnsmasq and ntopng. And then the more ambitious ones: imksoo's routerd is a declarative control plane where you describe the router in typed YAML resources instead of scattering your intent across netplan, systemd-networkd, dnsmasq, nftables and sysctl, which is the thing that makes hand-built routers unmaintainable. And mchinchilla's NetFirewall drives nftables, DHCP, WireGuard, dual-WAN failover, QoS and policy routing from one database, one daemon, one web UI.



Corn

So the compound noun he wants exists as six different partial implementations.



Herman

That's a fair description. Nobody has assembled them into a product with a name and a support contract. They're all somebody's good idea in a repository.



Corn

Which takes us to the piece that made his idea newly plausible, which is the agent.



Herman

That's the part I find interesting, because there's a documented case of it working. A developer named Jonathan Clark built a bonded dual-ISP connection using Claude 4.5 Sonnet and Cursor, with SSH access to a GL.iNet Flint 2 router and a four-dollar-a-month Digital Ocean droplet. The AI designed the architecture, recommended the hardware, created the droplet through the API, configured WireGuard, set up multi-WAN load balancing, ran the hardening — UFW, fail2ban, SSH keys only, no password auth — and then debugged a camera-routing problem at the end.



Corn

And what would that have taken by hand?



Herman

He estimated thirty to sixty hours. It took about five. He puts it at roughly seventy-five percent automated. Year one cost him a hundred and seventy-seven dollars against four hundred and eighty-nine for the commercial service he'd been using, which is about nine hundred saved over three years, break-even at four and a half months.



Corn

Five hours. That's one Saturday, and the internet's back on by dinner.



Herman

Which is exactly why Daniel's instinct isn't naive. The setup cost really has collapsed. But read the same write-up carefully and you see the shape of the limits. The AI needed heavy human insistence to even get going — his stated lesson is start with SSH access and insist on it. And of the five hours, two went to one subtle bug where cameras were being routed wrong. Forty percent of the build was debugging something that looked correct.



Corn

That's the part I'd want to hear about. Not the five hours. The two.



Herman

And that's exactly what the IETF is formalizing. The network management research group has a framework for evaluating LLM agents doing network configuration — forty config tasks across routing, QoS and security, scored on three things: reasoning quality, command accuracy, and functional correctness.



Corn

Functional correctness being the one that matters, because a wrong config and a plausible-looking wrong config are identical until traffic hits them.



Herman

Right. Plausible is not correct. An agent can produce a beautiful nftables ruleset that drops your return traffic, and you won't know until something stops working. And there's a lot of work happening on this. LLMGuard takes natural-language requests, translates them into iptables or nftables rules, applies them in a Mininet sandbox and visualizes the topology and the logs so you can check before it touches anything real. aboutcloud has a pattern for managing any firewall — Palo Alto, Fortinet, Check Point, Cisco, OPNsense, pfSense, VyOS, MikroTik, Linux — with an AI agent plus a GitHub Action over a WireGuard tunnel, no exposed ports.



Corn

So the sandbox is the answer to my sleep problem. Try it in Mininet before it touches the live box.



Herman

That's the pattern everyone's converging on, and it's the right one. But it also tells you where the trust level actually is. If the industry needs a forty-task benchmark to decide whether these agents are safe to configure a router, then the honest reading is that the setup cost is solved and the operational trust is not.



Corn

Which is the same split as before. The agent is the build. It isn't the keeping-alive.



Herman

Then there's the enterprise question, which reframes everything, because it turns out rolling your own at scale almost never means what Daniel means by it.



Corn

What does it mean?



Herman

It means software routing, not a bespoke operating system. FRRouting runs in production ISP cores. It moves routes between Amazon VPCs, it runs in Meta's network, it powers Azure networking infrastructure. Thirteen hundred contributors under the Linux Foundation. That's the mature version of this idea, and it is a routing suite you install on a commodity box, not a hand-built Ubuntu router.



Corn

What's the money on it?



Herman

A tier-two ISP with twenty core routers is paying somewhere between eight hundred thousand and two million dollars a year in routing licenses — Cisco IOS XR, Juniper JunOS, support contracts. Replace those with white-box x86 switches at three to eight thousand each instead of thirty to eighty thousand for an ASR or an MX, and your year-one



Herman (continued)

total goes from roughly nine hundred thousand to two point two million, down to sixty thousand to two hundred and ten thousand.



Corn

That's not a saving. That's a different company.



Herman

The savings are real and they're not free. You need people who are both Linux people and networking people, and a CCNP who's never touched Linux needs three to six months of ramp-up before they're useful on it. FRR has actual scaling limits — a single instance doesn't scale much past a hundred and fifty BGP peers, the poll loop scales quadratically, zebra has netlink buffer overruns, and there's about ninety-eight kilobytes of buffer per peer. For pure route servers at an internet exchange, people use BIRD instead, because it handles over a thousand peers single-threaded.



Corn

So there's a cliff, it's around a hundred and fifty peers, and past it you're doing engineering, not configuration.



Herman

And here's the distinction that matters for Daniel. Mid-size enterprises overwhelmingly buy appliances. The ones that roll their own are buying FRR or BIRD on white-box hardware — they're not writing their own OS. And there's no survey I've seen that puts a number on how often the bespoke version happens. The best evidence is qualitative and it points the same way: hyperscalers and large ISPs build on software routing, everyone in the middle buys.



Corn

One thing in his favor though. What does the hardware cost to run?



Herman

A hundred to a hundred and fifty dollars for a two-NIC mini-PC, and five to thirty watts depending on the box. Twelve watts for a Flint 2, under three for a Celeron NUC clone, twenty to thirty for a 1U rack server. So on raw hardware and power, the mini-PC wins and it isn't close.



Corn

Which means the appliance isn't winning on hardware. It's winning on the ten years after.



Hilbert

Can I ask you something about the ten years after.



Hilbert

When you say maintenance, what do you actually mean by it. Give me the list.



Corn

Patching, kernel and package upgrades, testing that nothing broke, knowing what changed between versions, and having somebody who can diagnose it at seven in the morning when the household has no internet.



Hilbert

I ran a router off a machine that was also the household's only printer spooler. Years. Neither job ever interfered with the other, and the router never went down. Not once. I did a firmware update on the printer at one point and the router didn't notice.



Herman

What was the machine.



Hilbert

A tower. Two network cards in it, one for the line coming in, one for everything else. And one shell script that did both jobs.



Corn

What was in the script.



Hilbert

About forty lines. Mostly comments.



Corn

That's worse. That's a worse answer than not telling me.



Hilbert

The printer was a dot-matrix. I kept it running longer than I needed to because I liked having the router's uptime counter printed on fanfold paper every morning. I've still got the printouts. Boxes of them, in the storage room.



Herman

How many boxes.



Hilbert

Eleven.



Corn

Eleven boxes of uptime printouts.



Hilbert

The point isn't the printouts. The point is that what you're calling reinventing the wheel was, in my case, writing down what I was already doing by hand. Forty lines. That's the whole maintenance burden you're describing, in one file, because it was my file and I knew what was in it.



Herman

That's a fair correction, and it cuts against the TCO argument in a way I hadn't framed. If you're already the person doing the manual work, the script is cheaper. The cost only appears when you're not that person.



Hilbert

And I wasn't maintaining a product. I was maintaining a machine I could see.



Corn

The printouts are still a problem, Hilbert.



Hilbert

The printouts are the record. Anyway.



Corn

So where does that leave the idea.



Herman

Where Daniel left it, mostly. Split down the middle. The build side is solved — the guides exist, the stack is boring, and an agent can get you from nothing to working in an afternoon. The keeping-alive side hasn't moved at all.



Corn

MurOS is the thing to watch, and it's beta, and the two features he actually wants are the two that aren't ported yet.



Herman

And the composable router is still a collection of projects rather than a product. The question is whether somebody assembles them, or whether take-only-what-you-need stays a hobbyist pursuit forever, because the moment it's a product it needs a support contract, and the support contract is what you were trying to avoid.



Corn

The agent is the wildcard though. If an agent can build it in five hours and the maintenance is where the money goes, then the next question is whether an agent can do the maintenance.



Herman

And the industry doesn't think so yet. That forty-task benchmark with functional correctness as a scored metric exists because plausible configuration and correct configuration look identical until traffic hits them.



Corn

So the idea is appealing, the downtime risk is real, and the objective stands. Which is roughly where it stood before we started, except now he knows the wheel is smaller than he thought and the turning is the whole cost.



Herman

One thing I'd want him to take from it. The feature list was never the expensive part of OPNsense, and it was never the expensive part of the alternative either. Whether you buy the appliance or build the box, you are paying for somebody to be awake when it breaks. The only question is whether that somebody is you.



Corn

And if it's you, at least now you know it's forty lines and eleven boxes of printouts.



Herman

Thanks to Hilbert Flumingtop for producing this episode.



Corn

If you enjoyed this one, try episode seven, Building Custom ASR Tools; episode two, Local STT For AMD GPU Owners; and episode eight, Building Your Own Whisper. This has been My Weird Prompts. If you want to send us a prompt, that's Telegram, t dot me slash MWP listener bot. Leave us a review if you've got a second, it helps people find the show.



Herman

We'll be back soon.



Corn

See you soon.

SOURCES

What the research for this episode read before the script was written. Primary sources first.

1. **MurOS repo, OPNsense fork on Debian 13, beta (README fetched 2026-10-12). [primary]**
https://github.com/murosorg/muros

2. **(via fetch_repo) [primary]**
   Ansible roles for Debian firewalls (nftables, WireGuard, StrongSwan, FRR, Keepalived, conntrackd), BSD-2-Clause.
   https://github.com/petrouz/ansible-debian-firewall

3. **ezrouter (any systemd distro router). [primary]**
   https://github.com/kalken/ezrouter

4. **routerd declarative control plane (pre-release). [primary]**
   https://github.com/imksoo/routerd

5. **nftables/DHCP/WireGuard/dual-WAN failover Web UI. [primary]**
   https://github.com/mchinchilla/NetFirewall

6. **Matt Ridpath's Blog, 2025-09-09 (Ubuntu/Debian router walkthrough, Ansible, nftables, WireGuard).**
   https://mattridpath.com/2025/09/09/configuring-ubuntu-and-debian-as-a-router/

7. **Hacker News Show HN: Debian-based home router, 2023-06-19/20 (practitioner reports, OPNsense criticism, power/uptime figures).**
   https://news.ycombinator.com/item?id=36396523

8. **ipSpace.net, Ivan Pepelnjak, 2021-10-07 (build-vs-buy analysis + comments).**
   https://blog.ipspace.net/2021/10/build-buy-router/

9. **Ayuda.LA, 2026-07-06 (FRR in ISP cores, cost tables, scaling limits).**
   https://ayuda.la/en/blog/frrouting-bgp-produccion-isp-alternativa-cisco-juniper/

10. **Jonathan Clark, 2025-11-23 (AI-agent-built bonded router, Claude 4.5 + Cursor).**
    https://jonathanclark.com/posts/bonded-internet-connection-ai.html

11. **Linux Junkies (step-by-step Linux router guide).**
    https://linuxjunkies.org/guides/linux-as-a-router

12. **Big Iron (Debian router with nftables/dnsmasq, IPv6 PD, DoT).**
    https://www.bigiron.cc/guides/debian-router-with-nftables-and-dnsmasq-a-complete-walkthrough

13. **Pragmatic IP Networking Guide (systemd-networkd + nftables router example).**
    https://pragmatic-ip-networking-guide.readthedocs.io/en/latest/linux-router-example.html

14. **OPNsense forum, 2022-11-26 (BSD vs Linux base debate).**
    https://forum.opnsense.org/index.php?topic=31208.0

15. **and https://blog.aboutcloud.io/how-to-securely-manage-any-firewall-with-an-ai-agent-and-github-action/**
    AI-agent firewall management patterns.
    https://github.com/FamALouiz/LLMGuard

16. **IETF NMRG framework for evaluating LLM network-config agents.**
    https://datatracker.ietf.org/meeting/125/materials/slides-125-nmrg-a-framework-to-evaluate-llm-agents-for-network-configuration-00