Episode Book

EPISODE #5904

Multi-WAN Beyond Failover: OPNsense vs OpenWrt

PUBLISHED	12 October 2026 at 13:22 UTC
DURATION	20:03
LISTEN	myweirdprompts.com/episode/multi-wan-opnsense-openwrt-routing-03166/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.4
SOURCES	12 cited by the research (listed at the end)
BOOK GENERATED	19 October 2026 at 17:41 UTC

SYNOPSIS

You've built the gateway group, set the tiers, and failover works. So why does a dropped link still kill your VPN, your game, and your VoIP call? This episode separates true bonding from load balancing, then walks through what the "advanced" version of multi-WAN actually looks like in open source. In OPNsense, that means gateway group tiers, trigger levels, weight-based distribution, sticky connections, and the one silent subnet-mask mistake that breaks per-device routing. On the OpenWrt side, it means mwan3's firewall-mark packet classification, the iptables-to-nftables transition, and a five-hundred-line Lua alternative called Mini-MWAN that does failover and nothing else. Along the way: why failover speed was never the bottleneck, why the public IP change is the real problem, and why people who've done load balancing properly still go looking at bonding.

TRANSCRIPT



Corn

Daniel's been down the multi-WAN road already. He runs OPNsense, he's built a WAN group, assigned priorities, and failover works. But he stopped there, because he didn't want to break the one thing that finally worked. So his question is what the next level actually looks like. He's asking about QoS-style approaches, whether anything out there can automatically spot a recurring MAC address, sort devices by type, say IoT versus laptops, and then permanently pin them to one WAN or another. And what the advanced version looks like, whether it's automatic or whether you're doing it by hand.



Herman

And the pitch he's working from is the one everybody starts with, which is you take a few connections, you stack them, and you get one fat pipe.



Corn

Right. And that's where the disappointment sets in.



Herman

It's where the vocabulary sets in, first. Because bonding and load balancing are different things, and almost nobody who starts this project wants the one they end up with.



Corn

Define the difference, because I think people use them interchangeably and then wonder why their download didn't get faster.



Herman

True bonding splits traffic at the packet level. A single download, a single video call, gets chopped into packets and spread across every connection at once, and it all comes out the other end through a tunnel that presents one stable public IP. That's the crucial part. One IP, one session, and when a link dies, the tunnel just absorbs it and the session never notices.



Corn

And load balancing?



Herman

Load balancing works at the session or flow level. Each new connection gets assigned to one WAN. So your aggregate capacity goes up, because different sessions are riding different links at the same time. But any single download still rides one link at that link's native speed. Two hundred megabit lines do not give one download four hundred megabit. They give one download two hundred, and a second download two hundred alongside it.



Corn

Which is fine, unless the thing you actually wanted was one big download.



Herman

Unless the thing you wanted was one big download. So why does everybody end up with load balancing? Because bonding is a project. You need a relay, usually a VPS somewhere, and your bonded speed is capped by that server's throughput. You need compatible hardware, you need multiple SIM plans if you're doing cellular, and you need to maintain all of it. The line I keep coming back to is that DIY bonding can work well, but it needs to be a hobby to be feasible, and it gets expensive quickly.



Corn

So the honest summary is you wanted a fire hose and you got a traffic cop.



Herman

You got a very good traffic cop. And that's the thing. Load balancing is more versatile in practice. Failover is trivially easy with it. Spreading clients across links so no single pipe saturates is useful. It's just less exciting on the box.



Corn

So the real question, then, is what the sophisticated version of the traffic cop looks like in open source. Because that's what Daniel is actually asking. He's already got failover. He wants to know what's above it.



Herman

And OPNsense is the right place to start, because that's his platform.



Corn

Walk me through it. He's already built a gateway group, so what's the layer above that?



Herman

Tiers. This is the mechanism people miss. OPNsense gateway groups have tiers, and the tier number is a priority, where a lower number means higher priority. If two gateways sit in the same tier, OPNsense load balances between them. If they sit in different tiers, the lower one is primary and the higher one is failover. And you get five tiers, each of which can hold multiple gateways.



Corn

So you can balance within a tier and fail over between tiers, in the same group.



Herman

You could have two fiber links in tier one balancing against each other, and an LTE modem sitting in tier two that only wakes up if both of those drop.



Corn

And how does it decide a link is dead?



Herman

There's a trigger level per gateway. Member Down is a hundred percent packet loss. Then there's Packet Loss above a threshold, High Latency, or the combination of packet loss or high latency. Plus gateway monitoring, which is the part people forget to turn on and then wonder why nothing switches.



Corn

So the failover logic is decent. What about the balancing itself, because equal distribution isn't always what you want.



Herman

That's the weight setting, and it's the lever most people never touch. If you've got a ten megabit link and a twenty megabit link, you don't want them taking equal shares, because the ten megabit one saturates first. So you give them weights of one and two, and the faster link gets twice the traffic.



Corn

Ten and twenty. Weights of one and two. That's a clean way to put it.



Herman

It's the QoS lever people are looking for without knowing the name. You're not shaping traffic, you're shaping distribution.



Corn

Now, sticky connections. This is the one that bit Daniel's kind of setup, I'd guess.



Herman

Sticky connections keep a given source's requests to the same website pinned to the same gateway. You need it because some sites don't react well to your request IP changing mid-session. Think about a banking site or anything with a session cookie tied to an IP. If request one goes out fiber and request two goes out LTE, the site sees two



Herman (continued)

different clients and the session falls over.



Corn

But it's not free.



Herman

It is very much not free. There are threads going back years where people fixed weird behavior by turning sticky connections off. Firesticks pausing with a no internet message, connections dropping for no obvious reason. And then there are other people in the same threads saying it's absolutely needed for their HTTPS sessions and everything broke when they disabled it.



Corn

So it's one of those settings where the correct answer depends on what your network is actually doing, and the documentation can't tell you which one you are.



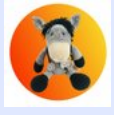
Herman

And nobody has resolved that. It's a genuine tension, not a settled question.



Corn

There's something grimly funny about a feature called sticky being the thing that makes your connection unstuck.



Herman

There's also a related setting people should know about, which is killing firewall states on failover. If your backup link is a metered LTE or 5G connection, you don't want stale states holding sessions open and burning data on the expensive link. You can have OPNsense kill states when it fails over. It needs gateway monitoring and gateway switching enabled to work.



Corn

Practical money question, that one. Now get to the per-device part, because that's what Daniel actually asked about.



Herman

Per-device assignment is done by hand and only by hand. You create a firewall rule, and it has to sit above the gateway-group rule in the order, you set the device as the source, and you set the specific WAN as the gateway for that rule. The device then always goes out the WAN you picked.



Corn

And there's a specific mistake people make here.



Herman

There's a very specific mistake. You set the source as a single host and you accidentally leave the subnet mask at slash twenty-four instead of slash thirty-two. Which means your rule doesn't apply to one device, it applies to your entire subnet. There's a forum thread from a couple of years back where the person confirmed per-device routing worked fine once they fixed exactly that, and moved the DNS rule to the top of the list.



Corn

So the feature works. It's just that the failure mode is silent and total.



Herman

Which is the story of this whole topic, honestly.



Corn

Here's the part I want to push on. Failover works. We've established that. The switching itself is fast. So why does a properly configured multi-WAN setup still feel disappointing when a link goes down?



Herman

Because the failover speed was never the bottleneck. The IP change is. Your connection moves to the other WAN, your public IP changes, and every session-based application notices. Your VPN drops. Your game drops. Your VoIP call drops. Load balancing doesn't protect active sessions when a WAN goes down, any more than bonding does. It just gets you back online faster.



Corn

So the thing that actually breaks is not the pipe, it's the identity of the pipe.



Herman

It's the identity. And this is exactly why bonding exists, and exactly why people who've done load balancing properly still go and look at bonding setups. You solve failover and you discover the real problem was never failover.



Corn

That reframes the whole question. Daniel's fear of breaking a working setup is well-placed, but the setup he has is arguably as good as session-level balancing gets on the session level.



Herman

On the session level, yes. Which is why the interesting question is what happens above the session level, and what the OpenWrt world does differently.



Corn

So that's the OPNsense side. What's the other major open-source implementation?



Herman

mwan3. It's OpenWrt's multi-WAN policy routing framework, and it's built differently. It classifies packets using firewall marks, and then it uses routing rules to send marked packets through per-interface routing tables.



Corn

Marks and rules. Explain why that's a different philosophy.



Herman

Because it's packet classification rather than session assignment. You tag packets at the firewall, then the routing layer obeys the tags. Which means you can get more granular than OPNsense's gateway groups, in principle. It supports up to two hundred and fifty WAN interfaces, it's hotplug-driven so it reacts when an interface comes up or goes down, and it does any combination of primary, secondary, failover, and load balanced or not.



Corn

Two hundred and fifty WANs.



Herman

Nobody has two hundred and fifty WANs. But the number tells you the design assumption is that this is a general framework, not a two-link home setup.



Corn

What's the state of it right now? Because I have a feeling there's a complication.



Herman

There's a significant complication. The classic mwan3 build is iptables-based, and iptables is on the way out in OpenWrt. There's an nftables port, version three point six point twelve, latest commit in August. It replaces iptables and ipset with nftables, it lives in its own table independent of the main firewall, it uses a configurable mark bitmask with a default value, and it includes sticky routing plus a LuCI app with a load balancing simulator.



Corn

A load balancing simulator.



Herman

Which is nice, because you can see whether your distribution is actually behaving without guessing.



Corn

So what's the complication?



Herman

mwan3 is not available in OpenWrt twenty-four ten. The legacy build is iptables-based and the nftables port targets twenty-five twelve. So there's a gap where the flagship multi-WAN tool for OpenWrt simply isn't there on the current release.



Corn

That's a fracture, not a bug.



Herman

It's an ecosystem fracture. And it produced something interesting, which is Mini-MWAN. Someone wrote a roughly five-hundred-line Lua daemon that does failover and basic weight-based load balancing using standard kernel routing metrics instead of firewall rules.



Corn

No firewall rules at all?



Herman

No firewall rules, no iptables or nftables dependency, works with any firewall backend. It's about five hundred lines of Lua against mwan3's ten thousand or more, and four dependencies against mwan3's many.



Corn

And the trade?



Herman

No sticky sessions, no per-protocol routing, no custom rules. It does failover and basic weighting and nothing else.



Corn

So it's the honest version of the thing. Which, I'll say, is a category I approve of.



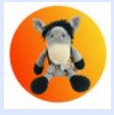
Herman

I knew you'd like that one.



Corn

I like anything that admits what it doesn't do. Now, what about actual bonding in open source, because Daniel said he'd been down the bonding road before.



Herman

OpenMPTCProuter is the closest open-source thing to real bonding. It aggregates multiple connections using Multipath TCP on top of OpenWrt, and it terminates over any VPS you run. It's ISP-independent, WAN-type-independent, latency-independent. Fiber, VDSL, ADSL, 4G, 5G, mixed. And it can be configured for either aggregation or failover.



Corn

But it needs the relay.



Herman

It needs the VPS. That's the toll you pay for a single stable public IP. And that's the honest architecture of the whole field. If you want one IP and one fat pipe, somebody has to be the meeting point, and that somebody is a server you pay for.



Corn

So the answer to what's above failover in open source is, you either accept per-session assignment and configure it carefully, or you rent a relay and do it properly.



Herman

That's roughly it.



Corn

And that gets us to the actual question Daniel asked, and I want to be direct about it. He asked whether anything can identify a list of recurring MAC addresses, sort them by type, and assign them persistently to a WAN. What's the answer?



Herman

Nothing in open source does that. There is no tool that automatically identifies recurring MAC addresses, classifies devices by type, and persistently assigns them to WANs. OPNsense doesn't have it. Mwan3 doesn't have it. Mini-MWAN certainly doesn't have it.



Corn

At all?



Herman

The closest thing in OPNsense is manual aliases plus per-rule gateway selection. You build an alias for your IoT devices, you build one for work machines, and you write rules that send each alias out a chosen WAN. That's the mechanism. There's no automatic classification layer sitting on top of it.



Corn

So the intelligent version is the well-organized manual version.



Herman

That's exactly what it is. The advanced tier is described in the guides as work laptops always on fiber, IoT on LTE, gaming on the most stable path. Which sounds intelligent. But the warning attached to it is that rules interact, edge cases appear, and the more you configure, the more you maintain.



Corn

There's no magic. There's a list.



Herman

There's a list, and there's someone maintaining the list.



Corn

And the feature requests to make this easier went nowhere, I assume.



Herman

There's a specific one worth mentioning. A feature request asked for sticky connections to work on source and destination rather than just source, which would give better balancing. It was closed as not planned after six months of inactivity.



Corn

Not planned is the most honest answer in software.



Herman

It's the most honest answer in software.



Corn

So the thing Daniel imagined, a system that learns your devices and sorts them, is a commercial SD-WAN feature. It's not an open-source one. Not because it's technically impossible, but because nobody has built it.



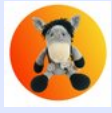
Herman

And the reason nobody has built it is worth thinking about. MAC classification sounds simple until you're the person who has to maintain the device database. Somebody has to decide what a device is. Manufacturer prefixes get you part of the way and then lie to you. A phone and a tablet from the same vendor look identical. A device that randomizes its MAC address per network is deliberately unclassifiable.



Corn

The automatic version has a maintenance problem underneath it that nobody wants to own.



Herman

The commercial vendors own it, and they charge for it, and that's the actual product. The classification is the product. The routing is the easy part.



Corn

That's a useful thing to understand. The clever feature isn't the routing. It's the database.



Hilbert

I tried to do this and got stuck on the sticky connections setting.



Herman

Stuck how?



Hilbert

I read the entire OPNsense documentation twice. Twice. Could not figure out why my wife's iPad kept dropping Zoom calls.



Corn

Did you resolve it?



Hilbert

I turned sticky connections off. It stopped dropping the calls. I still don't know why. I've stopped asking why.



Herman

That tracks with everything we've said.



Hilbert

Here's my complaint, though. My setup is intelligent. I assigned the smart TV to the slower WAN so it wouldn't hog bandwidth while I'm on work calls.



Corn

That's just a firewall rule.



Hilbert

I call it the Hilbert method. It's a QoS strategy. I've got a spreadsheet with every MAC address in the house, colour-coded by priority level. I update it weekly.



Corn

Weekly.



Hilbert

Weekly. And I did try to assign the refrigerator to a specific WAN at one point, because it was using too much bandwidth on the IoT VLAN. That one didn't take.



Herman

The refrigerator.



Hilbert

It kept reassociating. I let it go.



Corn

You've got a separate WAN for the doorbell.



Hilbert

The doorbell camera has its own WAN because it needs the lowest latency. I called the ISP at one point because the doorbell was buffering. They sent a technician. He looked at it for about twenty minutes and said the doorbell was fine.



Herman

Was it?



Hilbert

He didn't understand the QoS requirements of a video doorbell. I've kept the WAN. It's still on it.



Corn

What does that WAN cost you a month?



Hilbert

Forty-one shekels. Worth it.



Corn

Right.



Herman

The misconception to kill here, and it's the one that sends people down this road in the first place, is that stacking connections gives you one faster pipe.



Corn

It doesn't. Session-level load balancing gives you more total capacity, and any single download rides one link at that link's native speed. Two hundred megabit lines give you two hundred megabit downloads, plural.



Herman

The second one, which is the bridge too far for a lot of people, is that anything open source will learn your devices and sort them for you.



Corn

It won't. You're building aliases and writing firewall rules above the gateway-group rule, and you're remembering to use a single host and not a slash twenty-four.



Herman

You're accepting that the thing you're actually protecting against was never failover speed.



Corn

It was the public IP changing. That's what kills your sessions, and that's what bonding solves, and that's why the whole field exists.



Herman

If the intelligent version is just tidy manual policy routing, the gap between what commercial SD-WAN vendors advertise and what open source actually ships is basically the word intelligent, plus a device database nobody wants to maintain.



Corn

Which is going to keep mattering, because the number of devices on a home network is going up, not down, and a spreadsheet updated weekly doesn't scale to a house with sixty things on it.



Herman

There's a real question underneath that, which is whether automatic classification ever arrives in open source, or whether it stays permanently commercial because the upkeep is the product.



Corn

My honest guess is it stays commercial for a long time, because the routing is easy and the database is the hard part, and Hilbert's spreadsheet is proof that even one man with one house can't keep it current.



Herman

Forty-one shekels a month for the doorbell. That's the whole economy of this topic in one number.



Corn

Thanks to Hilbert Flumingtop for producing. If you enjoyed this one, try episode two, Local STT For AMD GPU Owners; episode three, Safetensors or something else; and episode eleven, How Does Fine Tuning Work Anyway. This has been My Weird Prompts. If you want to send us your own prompt, find us on Telegram at t dot me slash MWP listener bot. We'll be back soon.

SOURCES

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

1. **OPNsense docs, gateway groups/multi-WAN [primary]**
   https://docs.opnsense.org/manual/multiwan.html

2. **OPNsense docs, Multi WAN how-to (sticky, weight, failover states) [primary]**
   https://docs.opnsense.org/manual/how-tos/multiwan.html

3. **mwan3 nftables port, v3.6.12 (2026-08-11) [primary]**
   https://github.com/dl12345/mwan3

4. **mwan3 original, up to 250 WANs [primary]**
   https://github.com/Adze1502/mwan

5. **Mini-MWAN, created 2025-10-20 [primary]**
   https://github.com/alex-schwartzman/mini-mwan

6. **OpenMPTCProuter, MPTCP aggregation [primary]**
   https://github.com/Ysurac/openmptcprouter

7. **pfSense policy routing docs [primary]**
   https://docs.netgate.com/pfsense/en/latest/multiwan/policy-route.html

8. **OPNsense forum, per-device gateway assignment (2024-09-06/07)**
   https://forum.opnsense.org/index.php?topic=42693.0

9. **OPNsense core issue, src/dst sticky (closed not planned 2022-10-06)**
https://github.com/opnsense/core/issues/5688

10. **Waveform, Definitive Guide to Multi-WAN Internet, updated 2026-05-05**
https://www.waveform.com/guides/multi-wan-guide

11. **session-level load balancing explainer**
https://homelabrouter.com/load-balancing-two-internet-connections/

12. **Speedify, bonding vs. load balancing**
https://speedify.com/blog/combining-internet-connections/bonding-vs-load-balancing/