Episode Book

EPISODE #5816

Why Linux Still Runs on a 1986 Mailing List

PUBLISHED	9 October 2026 at 16:23 UTC
DURATION	24:56
LISTEN	myweirdprompts.com/episode/linux-mailing-list-transparency/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.2
SOURCES	15 cited by the research (listed at the end)
BOOK GENERATED	19 October 2026 at 17:40 UTC

SYNOPSIS

Every Linux kernel release candidate still ships through email. In September 2026, Linus Torvalds announced Linux 7.3-rc5 on the Linux-Kernel Archive, and a twenty-patch kbuild series got reviewed inline on the same list. This episode traces the listserv from its 1984 human-administered origins through Éric Thomas's 1986 automation breakthrough to the present day, asking why the most advanced computing community on earth never abandoned a forty-year-old tool. Along the way: Python's unmoderated 1991 mailing list, the invention of double opt-in and the first spam filter, and what Slack, Discord, and GitHub pull requests actually give up compared to plain-text email with a public archive.

TRANSCRIPT



Corn

Here's the question that's been rattling around my head since Daniel sent this in. If you're at the absolute frontier of computer science, why are you still arguing on mailing lists?



Herman

And he's got a point. He listened to the Python episode and got hung up on one detail, this listserv post that helped ignite the language back in the day, and his question is basically, is that a relic or is it still happening?



Corn

Right, and Daniel being Daniel, he doesn't stop at the tech question. He notices that the people at the bleeding edge do things like keep hand-coded HTML personal sites and have animated discussions on listservs, and he finds something wonderful in the persistence of those communities.



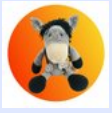
Herman

Especially the ones around foundational Linux computing, where every exchange just sits on the open internet for anyone to read.



Corn

He calls it radical transparency, and he makes the case that it's particularly well suited to open source, which might be exactly why the community has never let go of it. Then he asks the two things we're here to answer: how did this technology get started, and are listservs actually still a phenomenon this year, or is that just a limitation of his search experience?



Herman

Which is a beautiful question to ask, because the answer is yes, and the receipts are spectacular.



Corn

One flag before we start. Daniel asserts the hand-coded HTML personal website thing as a fact. We went looking and couldn't substantiate it as any kind of documented pattern, so treat it as his impression rather than a verified phenomenon.



Herman

Fair. Though I will say the man has a nose for these things.



Corn

So let's take the whole thing apart. The machine, the culture, and the receipts from this year.



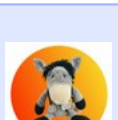
Herman

Take it from the top. What is a listserv, exactly, because the word gets thrown around like it means any old email list, and it doesn't.



Corn

Technically it's a product. LISTSERV, all caps, one specific piece of software. But it's been genericized the way Kleenex and Xerox were, so when people say listserv now they mean any email-based mailing list with a subscription mechanism and an archive.



Herman

And it's the archive that makes the difference. That's the thing everyone skips past.



Corn

Explain that, because I think it's the whole episode in one mechanic.



Herman

Here's the architecture. One address to post to. One address to subscribe. And a public archive that persists after everyone involved has moved on, changed jobs, or died. That third piece is what turns a contact list into a community.



Corn

So the distinguishing feature isn't the email. It's the memory.



Herman

It's the memory, and the fact that the memory is readable by anyone with a browser and no account.



Corn

Which is exactly the tension Daniel is poking at. These are tools from 1986. Thirty-nine, forty years old depending on how you count, and the Linux kernel is shipping release candidates through them.



Herman

Not as a historical exercise either. As the actual production channel. That's not nostalgia. Something in the design is load-bearing.



Corn

So we've got an arc to run. Where the technology came from, why a language's early growth depended on it, what the modern alternatives actually give up, and then the receipts from this year that settle Daniel's question.



Herman

And to understand why any of this survived, you have to go back to the moment it almost didn't.



Corn

Bitnet. Start there.



Herman

The original Bitnic Listserv ran from 1984 to 1986. Built by Ira Fuchs, Daniel Oberst, and Ricky Hernandez to implement mailing lists on IBM VM mainframes on the BITNET network, which was this academic network, kind of the scholarly cousin of the early internet.



Corn

And the crucial detail is how you subscribed.



Herman

You wrote to a human. There was a person at an address called INFO at BITNIC, and you sent them a message asking to be added to a list, and they added you by hand.



Corn

That's remarkable. The automation we take for granted was a person with a stack of paper.



Herman

L-Soft's own history of the thing says it plainly. The process became so slow and cumbersome that it threatened the viability of mailing lists altogether. That's their phrasing.



Corn

So the bottleneck wasn't the network. It wasn't the computers. It was the administrator.



Herman

The administrator was the single point of failure for the entire idea of group communication.



Corn

Enter Eric Thomas.



Herman

1986. Éric Thomas, an engineering student at École Centrale Paris, builds Revised LISTSERV, and this is the actual invention that made the listserv a phenomenon. First automated mailing list manager. You could join or leave without a human in the loop.



Corn

And that's why the word exists. That's why it's his name on the software.



Herman

It's worth naming him because almost nobody does. Every time someone says listserv, they're saying his product.



Corn

What happened after that is a scale story.



Herman

BITNET peaked around 1991 at roughly one thousand four hundred organizations across forty-nine countries. By 2000, LISTSERV was managing more than a hundred and seventy thousand lists with over a hundred million subscriptions.



Corn

A hundred million subscriptions before most people had email at home.



Herman

LISTSERV was freeware from 1986 to 1993, then went commercial through L-Soft, which was founded in 1994. And two features came out of that era that are now everywhere. Double opt-in, 1993. And the first spam filter, 1995.



Corn

Every confirmation email you've ever clicked was invented for a mailing list manager.



Herman

Every one. Nobody thinks about it.



Corn

Now the Python thread, because that's what Daniel actually asked about. The thing he heard on the other episode.



Herman

Guido van Rossum's welcome message to the Python mailing list is dated Friday, the twenty-second of November, 1991.



Corn

Read me the interesting part.



Herman

Here's what he wrote. The list is not moderated. Everything you send to it is immediately forwarded to anybody else on the list. So be a little careful in what you write.



Corn

That's a nineteen ninety-one project lead telling his first users there's no safety net and no editor. You're on your own.



Herman

And it was the deliberate design choice for a brand-new language with essentially no users.



Corn

Why would you do that? Why not gate it?



Herman

Because the alternative was a bottleneck. He'd just watched what a human filter does to a conversation. An unmoderated list is fast, it's honest, and it's public. You can see everything, including the mistakes.



Corn

The plumbing detail is nice too.



Herman

Messages went to python-list at cwi dot nl. Subscription requests went to python-list-request at cwi dot nl. That dash-request convention survives in mailing list software to this day.



Corn

A thirty-five-year-old naming convention still sitting in postfix configs somewhere.



Herman

Everywhere. And there's an earlier breadcrumb. Python 0.9.1 was posted to Usenet on the twentieth of February, 1991, to alt dot sources. And when 1.0.0 was announced, the message said, quote, there's a mailing list, write to python at cwi dot nl to subscribe, no LISTSERV commands please. That parenthetical is a tiny gem. It means by 1991 the LISTSERV command syntax was already a known annoyance to users. The thing was popular enough to be irritating.



Corn

So the tool had already made the leap from infrastructure to household annoyance.



Herman

Which is a kind of success.



Corn

How did the list actually help the language grow, though? That's the part I want to understand.



Herman

It gave a scattered, tiny population of interested people a single shared room with a permanent record. Before that, you had individuals, and you had Usenet newsgroups, but a mailing list is something you subscribe to, which means you're announcing to yourself that you're part of it.



Corn

It's a membership, not a feed.



Herman

And by 1994 van Rossum had archived almost all messages posted to the Python mailing list and newsgroup as hypertext using Kevin Hughes' hypermail. And he noted that the newsgroup, comp dot lang dot python, contains the same messages.



Corn

So he treated the archive and the list as one artifact.



Herman

The conversation was the documentation. That's the thing I keep circling back to. There was no docs site. There was no wiki. There was a mailing list, and the mailing list was the knowledge.



Corn

And it's all still readable. Somebody today can go read the nineteen ninety-one thread where a decision got made.



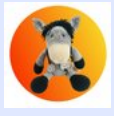
Herman

Which is the whole transparency argument, made concrete. It's not an abstraction. It's a URL.



Corn

That's the origin. Now here's the part that answers Daniel's actual question. What's happening this year.



Herman

So let me just answer it directly, because he wondered if he was seeing a search artifact. He isn't. Linux 7.3 release candidate 5 was announced by Linus Torvalds via email to the Linux-Kernel Archive on the twenty-seventh of September, 2026.



Corn

That's two weeks ago.



Herman

Two weeks. And on the seventeenth of September, Torvalds posted to the linux-kernel list reviewing a twenty-patch kbuild series, explicitly to significantly speed up kernel builds.



Corn

A twenty-patch series reviewed in public email.



Herman

On the twenty-fifth of August, Cong Wang announced mklinux version 7.0-mk2, the first public release of the multikernel Linux tree, via an email to that same list. And on the twenty-fourth of July, Greg Kroah-Hartman posted a reply on the 6.18.40 stable release.



Corn

That's a straightforwardly live working list. That's not an archive being maintained for sentimental reasons.



Herman

It's carrying the actual work of the kernel. Right now.



Corn

Give me the volume.



Herman

About one thousand four hundred messages a day, most of them kernel code patches. And going back, a study of the list covering 1995 to 2000 found fourteen thousand five hundred and thirty-five people from more than thirty countries had posted to it.



Corn

Robert Love has the line, doesn't he?



Herman

He does. If the Linux kernel community had to exist somewhere physically, it would call the Linux Kernel Mailing List home.



Corn

That's all of it in one sentence.



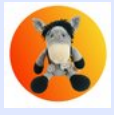
Herman

And here's the bit that surprises people who've only ever used GitHub. Kernel patches are sent as plain-text emails. You use git send-email or b4 send, and the patches get reviewed inline in the email thread.



Corn

So the code is in the message body.



Herman

The code is the message. And someone running this in production put it well, that it's superior to GitHub's pull requests because everything is sanely threaded since it's all email.



Corn

Sanely threaded.



Herman

Threading is a data structure. Email has had it since the beginning. Pull requests had to reconstruct it on top of something that didn't want it.



Corn

That's a non-obvious argument. It's not sentimentality, it's information architecture.



Herman

The mailing list stores a conversation as a tree. The web forum stores a conversation as a series of pages, and then tries to reconstruct the tree with quote blocks.



Corn

And quote blocks are a horror show. Everyone knows this.



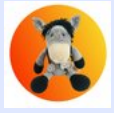
Herman

Everyone knows this and nobody fixes it, because the fix is to use email.



Corn

So now do the honest comparison. What do you give up with the modern tools?



Herman

Slack and Discord. No message history when you join. You see today, and maybe last week if the plan allows it. Invite-gated, so you have to know someone. Proprietary, so the archive belongs to a company. Not indexed by search engines, so nobody can find the



Herman (continued)

answer you already gave. And not durably exportable.



Corn

That last one is the killer.



Herman

Someone arguing against the nginx shutdown put it bluntly. None of those are reachable without an account and in many cases an invite. They're not indexed by search engines. They're proprietary, cannot be exported or archived. It's asking for knowledge to be lost.



Corn

Asking for knowledge to be lost. That's a strong way to frame it, and it's accurate.



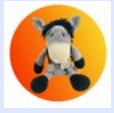
Herman

It's the strongest single sentence in the whole debate, and it's from a comment thread rather than a white paper.



Corn

Now the counter-case, because it's real and we shouldn't wave it away.



Herman

It's real. The newcomer problem. Percona launched something called Hackorum in February of 2026, and what it is, is a forum-style read-only web view of the PostgreSQL hackers mailing list.



Corn

Just a view.



Herman

And their stated reason is that this list is the main communication channel for PostgreSQL core development, and that decades-old email threads are, quote, not exactly welcoming to newcomers.



Corn

So somebody looked at a mailing list and said the content is fine, the interface is hostile.



Herman

And then in May, Hackorum and pginbox joined forces on the same mission.



Corn

Here's what I want to flag, because it's the tell. They didn't replace the mailing list.



Herman

They didn't. They built a view on top of it. The email substrate persists, and only the interface modernizes.



Corn

Which tells you what they actually believe about where the truth lives.



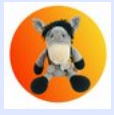
Herman

The list is the source of record. The forum view is a reading aid.



Corn

That's a much bigger statement than it looks.



Herman

It is, because it's a project saying we're not migrating. We're just building a window.



Corn

Now the governance angle, which I think is the sharpest thing in all of this.



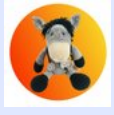
Herman

September 2025. nginx dot org announces it's shutting down its mailing lists by the end of that month. Sparked a real community argument.



Corn

And freenginx.



Herman

Freenginx, the fork started in February 2024 by Maxim Dounin after F5 took over nginx, kept its mailing list support. One more reason to look into the fork, as people were saying.



Corn

And the detail I love is where Dounin's fork announcement was sent from.



Herman

From the nginx mailing list. He announced the fork on the list he was leaving.



Corn

Which is almost too on the nose.



Herman

And his stated reason. I no longer see nginx as a free and open source project developed and maintained for the public good. I'm starting an alternative project, which is going to be run by developers, and not corporate entities.



Corn

So the mailing list became a proxy for who controls the project. The corporate parent killed it, the community fork kept it.



Herman

That's not a technical decision wearing technical clothes. That's a governance decision wearing an email address.



Corn

And that's the thing I don't think Daniel fully anticipated when he wrote this. He's asking about tech persistence and the answer is partly about politics.



Herman

It's a lot about politics. Which list you're on tells you which project you're actually part of.



Corn

Now the wrinkle that closes the loop on his radical transparency idea.



Herman

lkml dot org now runs Anubis proof-of-work anti-scraping protection, because AI companies aggressively scraping the archives were causing downtime.



Corn

So the openness that makes these lists valuable is now expensive to maintain.



Herman

Radical transparency has a bandwidth bill, and somebody has to pay it. That's the knock-on effect nobody talks about when they praise open archives.



Corn

The archive is free to read and expensive to serve.



Herman

And the more valuable it becomes as training data, the more expensive it gets to keep it free.



Corn

Do you want to do the survey of what's dying and what's adapting?



Herman

Sure. PostgreSQL core development runs entirely through the pg-hackers list. Guix debated proposals that committers are supposed to read and acknowledge messages coming through the guix-devel list at least weekly. Fedora's discussed retiring its scm-commits list.



Corn

So some lists die, some adapt, some are load-bearing.



Herman

And the ones that survive are the ones where the archive is the project's memory. When the list is where the reasoning lives, you can't turn it off without losing the reasoning.



Corn

When it's just notifications, it dies quietly.



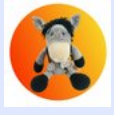
Herman

Right. Nobody fights to keep a notification feed.



Corn

Okay. So the archive, the cost, and the newcomer. That's where we've ended up.



Herman

It's a lot to hold at once.



Corn

The thing I find funny about all of this is that the most cutting-edge infrastructure on the planet routes through a 1986 email manager that was written by a student in Paris who just didn't want to add people to lists by hand.



Herman

That's how a lot of the important stuff happens. Somebody gets tired of doing something manually.



Corn

Herman, you had a thought you didn't say.



Herman

I had a thought about the threading argument that I wanted to test on you.



Corn

Test away.



Herman

If threading is the actual reason, then any modern tool that implements real threading should be just as good. And nothing does.



Corn

So the argument isn't that email is better. It's that nobody's bothered to rebuild the one feature email got right.



Herman

Nobody's bothered. And I don't fully understand why, honestly. It's not hard.



Corn

Because the incentives go the other way. A proprietary tool profits from making its history hard to export.



Herman

That's darker than where I was heading but I think you're right.



Corn

So to Daniel's point about openness, the mail list isn't just old. It's the only option that has no commercial reason to lock the door.



Herman

That's the design position.



Corn

And the reason it stays around, I think, is that when you've got nothing to sell, you've got nothing to hide.



Herman

Which is why the freenginx fork kept it and F5 didn't.



Corn

Now hold on, because let me go back to the newcomer thing for a second.



Herman

Go.



Corn

The complaint is that a thirty-year archive is unfriendly. But the archive is also the only reason a newcomer can learn anything. Where's the line?



Herman

The line is discoverability. The old threads are all there but you can't find the right one, and you can't tell what's outdated, and there's no index. The information is complete and unusable.



Corn

So the problem isn't too much. It's no map.



Herman

That's a much better way to say it than anything I had.



Corn

How would you fix it?



Corn

I don't know. And I don't think anybody does, which is what makes Percona's approach interesting. They didn't solve the map problem. They just made the pages readable.



Herman

A view without an index is still better than nothing.



Herman

It is. But it's a reading aid, not a solution.



Corn

Meanwhile the list keeps running, and the newcomers keep arriving, and the archive keeps growing.



Herman

At one thousand four hundred messages a day.



Corn

That's a lot of memory being written.



Herman

That's a lot of memory being written and almost none of it being read, at least at the time it's written. But it's there. For whoever needs it later.



Hilbert

The binding had cracked, so I had it rebound in the winter of eighty-two, which cost about eleven dollars, and it's still the original pages inside.



Hilbert

Sorry. The ledger. This was when I was adding subscribers by hand. Before Thomas wrote the automation. There was a person who did that, and for a while the person was me, at the institution I won't name because it took them eighteen months to pay the invoice and I haven't forgotten.



Herman

So you were the bottleneck.



Hilbert

I was the bottleneck, and I want to correct the framing you two have been running with for forty minutes.



Corn

That automation was unqualified good.



Hilbert

I'm not saying it should have stayed manual. I'm saying something was lost and nobody put it in the changelog. The manual step was the filter. When a human had to read your request, people wrote better requests.



Herman

Better how?



Hilbert

You could tell who was serious. The person who sent three lines naming the list, their institution, their department, and why they wanted in. And the person who sent a paragraph with no subject line.



Corn

And you had a three-ring binder of these.



Hilbert

Alphabetized. I could tell you which department at which institution a person was in by the handwriting on the envelope.



Herman

By the handwriting.



Hilbert

You stopped reading the name after a while. You knew the hand. The binder was the original database index, and I'll defend that to anyone.



Corn

What happened to the binder?



Hilbert

I've been meaning to digitize it since the mid-nineties.



Corn

Since the mid-nineties.



Hilbert

I have the scanner. I bought it and I set it up and it's still in the box. Well, it's in the box and the box is on the shelf and the shelf is behind the door, so it's mostly a layout question now.



Herman

What kind of scanner?



Hilbert

A flatbed. The box has the original foam inserts, which is why I haven't opened it, because once you take the foam out it never goes back the same way.



Corn

That is a real problem with foam.



Hilbert

Thank you. Nobody else gets that.



Herman

So the ledger is just... on a shelf, behind a door, in a box, next to a scanner that's in its own box.



Hilbert

Correct. I use it now and then. Mostly I don't.



Corn

What's in it that's worth digitizing?



Hilbert

Names. Four hundred and something of them on the first list alone. Every one of them wrote in by hand and every one of them got a reply from me, usually within the week.



Herman

Now a subscription takes four seconds and nobody reads the request.



Hilbert

Nobody reads it. I'm not saying it's worse. I'm saying the reading was doing something.



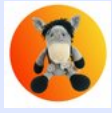
Corn

The filter that automation removed was the only thing making people think before they typed.



Hilbert

I don't know about only. But it was making them think.



Herman

I'd like to push back on one thing, because I don't think the filter is gone. It just moved.



Hilbert

Where'd it move to?



Herman

Into the archive. Nobody reads your request before you post now, but everybody can read it afterward, forever, with your name on it.



Corn

The filter isn't before. It's after.



Hilbert

That's a real answer. Which means the request is still written by someone who knows it'll be read.



Herman

Just not by an administrator. By everyone who comes later.



Hilbert

Which is a longer wait. The reply you got from me was in a week. The reply from the internet might be in thirty years.



Corn

There's no envelope to read the handwriting on.



Hilbert

There's no handwriting. That part's just gone. I'm not going to pretend that's a loss worth reversing. It's a loss. I'll leave it there.



Corn

The ledger stays on the shelf.



Hilbert

The ledger stays on the shelf. The scanner stays in the box. Somebody should do something about that, and it won't be me. I've got a show to produce.



Corn

The archive and the cost are actually the same problem in two directions, aren't they?



Herman

They are. The thing that makes the list valuable is the thing that makes it expensive, and the same openness that lets a newcomer read a thirty-year-old thread is the openness that lets a scraper take the entire business.



Corn

The freenginx fork kept its list and nginx dot org didn't, and that wasn't a decision about email. That was a decision about who the project belongs to.



Herman

Every project with a list is going to keep making that decision, and I don't think there's a stable answer. The economics move.



Corn

The newcomer problem is real and nobody has solved it. Hackorum and pginbox are views, not replacements. Nobody's figured out how to hand a newcomer thirty years of context without drowning them or hiding the archive behind a login.



Herman

The only thing everyone agrees on is that losing the archive is the loss you can't undo.



Corn

Which brings me back to what Daniel actually noticed. The people at the frontier keep choosing the old tool, and it's not nostalgia.



Herman

It's a set of tradeoffs. Openness, persistence, and who owns the conversation.



Corn

The listserv isn't a relic. It's a design position, and it's the position these people keep taking on purpose.



Herman

Thanks to Hilbert Flumingtop for producing this one, and for the reminder that the reading was doing something.



Corn

If you enjoyed this one, try episode thirty-five, The Privacy Gap; episode four twenty-five, The Arc of Deprecation; and episode ten twenty-one, The Python Paradox. This has been My Weird Prompts.



Herman

If you're enjoying the show, leave us a review, it helps people find us.



Corn

If you've got a question about something everyone else has written off, send us your own prompt on Telegram at t dot me slash MWP listener bot.



Herman

We'll be back soon.

SOURCES

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

1. **L-Soft, History of LISTSERV [primary]**
   https://www.lsoft.com/corporate/history-listserv.asp

2. **Python Archives, Guido van Rossum welcome message, 1991-11-22 [primary]**
   https://legacy.python.org/search/hypermail/python-1992/0000.html

3. **Python Archives in hypertext, 1994-09-05 [primary]**
   https://legacy.python.org/search/hypermail/python-1994q3/0428.html

4. **Linux-Kernel Archive (Oct 2026 index) [primary]**
   https://lkml.iu.edu/

5. **Linus Torvalds, kbuild patch review, 2026-09-17 [primary]**
   https://lists.openwall.net/linux-kernel/2026/09/17/2592

6. **mklinux v7.0-mk2 announcement, 2026-08-25 [primary]**
   https://lists.openwall.net/linux-kernel/2026/08/25/1911

7. **IETF list, Why use mailing lists?, 2025-09-26 [primary]**
   https://mailarchive.ietf.org/arch/msg/ietf/q6A_anL1u-Y9iXe-vboiOYamsl0/

8. **Guido van Rossum, The History of Python blog index [primary]**
   https://gvanrossum.github.io/python-history/index.html

9. **Wikipedia, LISTSERV (stable release 17.5, Nov 20 2024)**
   https://en.wikipedia.org/wiki/LISTSERV

10. **Wikipedia, LKML**
    https://en.wikipedia.org/wiki/Linux_kernel_mailing_list

11. **BigGo News, nginx mailing list shutdown, 2025-09-27**
https://biggo.com/news/202509270116_Nginx_Shuts_Down_Mailing_Lists

12. **HN discussion of nginx shutdown, 2025-09-26**
https://news.ycombinator.com/item?id=45390398

13. **HN, freenginx announcement, 2024-02-14**
https://news.ycombinator.com/item?id=39373256

14. **Percona, Hackorum launch, 2026-02-02**
https://percona.community/blog/2026/02/02/hackorum-a-forum-style-view-of-pg-hackers/

15. **Percona, 2026-05-13**
https://percona.community/blog/2026/05/13/two-projects-one-mission-hackorum-and-pginbox-join-forces/