EPISODE 1328

[INTRODUCTION]

[00:00:00] ANNOUNCER: In the late 1970s, a printer at MIT kept jamming, resulting in regular pileups of print jobs in the printers queue. Some frustrated computer scientists wrote a software program that alerted every user in the backup queue with a message, “The printer is jammed. Please fix it.” Richard Stallman reached out asking for a copy of the faulty software. But he was refused a copy of that program. He resolved to create a publicly available operating system, and the open source movement was born. Over 50 years later, open source has become a coding philosophy practiced by millions of software engineers around the world. Why is open source so popular? What difference has it really made in software engineering? And what major projects are open source? In this episode, we talk with William Morgan, CEO of Buoyant, and creator of the open source service mesh, Linkerd.

[INTERVIEW]

[00:00:57] JM: William, welcome back, once again.

[00:01:00] WM: Thank you very much, Jeff. Great to be here.

[00:01:02] JM: First question, what is your favorite restaurant in Austin?

[00:01:06] WM: Oh, man. Okay. So, yes, I did move to Austin recently from San Francisco, which I never thought we would do. We were there for 15 years, and definitely did not expect to end up anywhere else. But here we are. What's my favorite restaurant? It's weird, because we moved in the middle of the pandemic and did a whole lot of takeout. And then as soon as things started opening up, they started closing down again. And now we're back to doing takeout. So I actually don't have a great answer for you.

[00:01:36] JM: Wait. Are things closed down in Austin again?

[00:01:38] WM: Our attitudes towards eating out have closed down, let me put it that way. Yeah, there're all sorts of drama. So yeah, I don't know, man. I kind of like Uber Eats. And I pick something random each time. And that's kind of my experience, which is really horrible. Because I know there's a lot of really good food here. Really, that's been our style so far.

[00:01:57] JM: Hmm. What's your favorite service mesh these days?

[00:02:01] WM: Oh, gosh, this is really a good one. Linkerd. Have you heard of it?

[00:02:06] JM: I've heard of it. I hear it's lightweight. I hear it comes with a social network for your services.

[00:02:12] WM: Yeah, that's right. When one service wants to recruit another service, it uses –

[00:02:18] JM: It's a friend request, right? It makes a friend request to the other service.

[00:02:22] WM: Yeah, I want to send you some friendly HTTP traffic for you to deal with for me. Yeah, no, Linkerd. That's the best one, and my favorite.

[00:02:34] JM: I think we did our first show five years ago. I'm not 100% sure, but I think it was five years ago. Yeah, because you did your first round April 22nd, 2015. My first show is July 15th, 2015. And I think our first show was like a year into my podcast, which that was July 2016, roughly. I distinctly remember where I was when it was recorded. And there's a whole lot of evolution that's taken place over that time. Actually, here's a question, how much code remains from what Buoyant was five years ago?

[00:03:11] WM: Yeah, that's a great question. So on the Linkerd side, we went through this big rewrite that happened kind of starting in 2018, and so that 1.X version. So right now we're on 2.X, and 11 is around the corner. We should talk about that, because it's super cool. But the 1.X version that we had back then, it's still there. We're still kind of maintaining it. So that code, I guess, remains, but none of that made it into 2.X. And then in terms of like the non-open source stuff, probably very little. Actually, I think, what I heard was that there's a little bit of code that I wrote way back when I'm still in there for like generating cookie session, randomized IDs or

something like that. Probably only because people were too horrified to do anything other than just leave it and hope to never have to touch it.

[00:04:04] JM: And today, what's your go to market looking like? What are you spending your time on? Is it like sales and marketing, product development, closing deals? What are you focused on?

[00:04:16] WM: My gosh, that is a cute cat. That is suddenly grabbing my attention. Yeah. So, for us, yeah, it's been – You said evolution. It's been a big evolution for us. Early on, when you and I were first talking, our focus was all around open source adoption. And how do we take this fledgling project, because it was this like tiny little project with this weird name in this weird category that no one had ever heard of before? How do we turn that into like something that people care about into like an object that's in the brain? And then you fast forward five years or whatever, and now service mesh, at least if you're in the Kubernetes world, like that is a category of things. And Linkerd is definitely a word that's in in your vocabulary despite some pretty intense marketing competition from very well-funded other projects. Linkerd is definitely a word in your brain. So a lot of what I'm doing these days is less about how do we grow the open source community? Because I think it's very healthy, and it's inclusive, and it's engaged, and it's kind of self-perpetuating. A lot of what I'm doing is how do we help – This is going to sound super boring, but how do we help companies, especially large companies, successfully adopt Linkerd, right? And that is less of a technology challenge. It turns out, and this is like why this kind of this model works, it's less of a technology challenge. It's more about, "Gosh, we have this weird set of constraints that only make sense in the enterprise. And we need the service mesh to now fit into that." So that's mostly what I'm spending my brain juices on.

[00:05:53] JM: What I really liked about the last time I saw your product back when we could actually visit offices, back when there were offices, was you were taking what you had with Linkerd and – Actually, going back a little bit further, Istio had done what I would consider is kind of an unfair move and sort of steal the service mesh branding from the market leader, basically, in order to gain control almost as a cut out of Google. I mean, we had a pretty long conversation about this. I still pretty much feel about the same way I did back then, that you have the CNCF, which masquerades as something that is not Google, when it actually is sort of Google. And they're able to just tip the scales and make everybody adopt Istio while sort of like vaguely

asserting that they're neutral by virtue of the fact that Istio is not a part of the CNCF, or not a project or whatever. But it's all this kind of like fake gamesmanship, and it's really appalling. But we don't have to talk about that today. Or we can if you want to. But what I liked was, basically, you were you're engaging in a strategy that didn't end run around that, which was you were building higher level UX-based differentiation. Istio was this kind of like clunky thing that still looks like a clunky thing. It looks like a big clunky enterprise product. And I called you the hipster service mesh before the show. That's a great place to be, because everybody wants to build with hipster products these days. You don't want to use Microsoft Excel . You want to use air table. You don't want to use Istio. You want to use Buoyant.

[00:07:32] WM: Yeah, yeah. Well, that's a really nice way of putting it actually. I think, for us, it was we had this moment of panic, I think, when Istio came out. And I know we've talked about this in the past. It's like, “Holy crap! Here's this thing. It comes from Google, which not only has like a trillion dollar market cap and can like invest incredible amounts of marketing and engineering into this.” But, also, Google just invented Kubernetes. And like Kubernetes kind of had this very public success story kind of crushing its container orchestrator competition. And the way that we emerged from that moment of panic was like, “Oh, my God! What are we going to do?” We said, “Well, do we actually feel comfortable?” Like we looked at Istio and there was stuff that we really liked about it. But we said, “Do we really feel comfortable telling our Linkerd users to go use Istio?” And like the answer was no. There's stuff that we really didn't like, and we came out of that conversation with the idea that the thing that we really needed to do was providing a service mesh option that was simple, right? Like that was the challenge. The challenge wasn't having a million features, right? That's the easy part. The challenge was if you are a human being who actually has to install a service mesh, what kind of world are you living in? And how do we make that like a tractable thing to do? So that focus on kind of the end user and having a little bit of empathy for them drove everything that we did from that moment onwards. And I guess that has resulted in Linkerd having this very particular view about simplicity and kind of a very – I'm happy to see this like wonderful community of very enthusiastic people growing up around that, because of that focus on them, right? The product focused on them.

[00:09:17] JM: Am I wrong about CNCF being a Google cut out?

[00:09:21] WM: I don't think it's really that like intentional. I think the CNCF is part of the Linux Foundation, and Linux Foundation takes a whole lot of money from Google, and IBM, and Red Hat, and like all these other companies that –

[00:09:36] JM: Do we know the distribution? Do we know the distribution?

[00:09:39] WM: It might be public. The Linux Foundation is pretty public with their data. It might be out there.

[00:09:46] JM: I'm going to see if I can do a little bit of Carmen San Diego style research right now while you're talking. But continue talking. I'm just going to see if I could find how much CNCF funding.

[00:09:56] WM: So that has an influence on things, I'm sure. But most of the people that we interact with at the CNCF are like TOC, which is a Technical Oversight Committee, which comprises like a bunch of grumpy engineers who like aren't in that world, right? They're like trying to make decisions about, "Well, is a project mature? Or how mature is this project? And what level, what tier should to be?" and stuff like that. So I don't think it's that bad. It's not as bad as you make it out to be. It is annoying at times. Yeah, it's not like a giant conspiracy, or at least not one that I'm privy to.

[00:10:34] JM: Well, then let's talk about a little bit more specific examples. So did Kubernetes actually win because of superior technology or because of marketing?

[00:10:43] WM: Well, that's a great question. And I could only give my, probably, naive impression there, which is what I think, and I was not like – When this was going down, when the whole Mesos, versus Kubernetes, veruss Docker Swarm, versus Nomad, versus whatever stuff was happening, a lot of that was happening while I was still on Twitter. And we kind of came into that pretty late, kind of at the tail – Well, not quite the tail end. That first version of Linkerd, the 1.X that I talked about that, actually, one of our goals was to treat each of these as like an equal kind of thing. So it would integrate with Mesos and talk to Marathon. And it would integrate with Kubernetes. And it would integrate with Zookeeper. And like we treated them all as equal kind of things and providing this abstraction layer on top of it. And that sounded really

nice at the time. It actually turned out to be horrifically complex. And we kind of developed a lot of empathy for our users for struggling to go through that.

But by the time we kind of reset the clock and said, “We're going to develop a Linkerd 2.X, and it's going to be a rewrite,” which is like a scary, crazy, bad idea, most of the time to do, we decided to focus just on Kubernetes. Because at that point, it's like two years later, Kubernetes was clearly the one that was going to kind of take over. So to answer your question, I think it was a couple things. Number one is I think there was really good design in Kubernetes. In terms of like where it fits in the spectrum of nothing is provided for you and you have to do everything yourself, to like, “We're going to do everything for you, and you don't get a lot of choice.” It sat in this really nice sweet spot that I don't think Mesos quite hit. My impression of – And I cut my teeth on Mesos at Twitter. My impression was, at least back then, the answer to everything, “Can I do this on Mesos?” They’re like, “Yes, you can. Write your own scheduler in C++,” and like that's how you do it, right? And then on the other end of the spectrum was like App Engine, it was like, “Can you do this with App Engine?” “No .You can only do these three things.” And Kubernetes kind of fit this nice spot in that spectrum where the answer was often, “Yes, you can do that. And you can do that with YAML,” right? And like the result is that you have YAML madness today, and like that's not great. And actually, I think that was a big part of why it was successful, was it gave you just enough that you could start building stuff on to it without having to devolve to like writing code.

For the SRE audience, especially, and the DevOps audience, like that's a nice sweet spot to be in. I think the other part of it was the Google name definitely was helpful, right? Like there's a big difference. When a project comes from like a well-established brand, it starts off with like a huge amount of interest, right? So you get this big head start. And Google was known and is known for its like really good infrastructure, especially, right? So that helped it.

I think there's some real magic there with folks like Craig, and Joe, and Brendan. I think a lot of magic behind Kubernetes was those three beautiful minds kind of working in harmony to make that thing work? And, yeah, like I said, that's my impression of why it took off. But I do think it was rare, right? Like I think in most open source kind of ecosystems, you don't end up with one really dominant one. And when you do, it's kind of remarkable and noteworthy thing. So coming into that world with that history fresh in everyone's mind coming in with Linkerd and then having

Istio drop for Google, that was like a really scary thing, right? And I remember everyone was just telling us, “Oh, Istio is going to win. It's going to be just like Kubernetes. You guys are like Docker Swarm. Oh, you like simplicity? That sounds like Docker Swarm to me.” And I was like, “Man, it doesn’t help me if that analogy really works.” Now, I'd like to go find those people and like haha in their face, but of course I don't do that.

[00:14:43] JM: They're listening.

[00:14:46] WM: Yeah, haha!

[00:14:47] JM: They're shamed.

[00:14:47] WM: One person is.

[00:14:48] JM: They're feeling shame. Actually, they've probably forgotten everything, everything they ever said about that topic. Probably most of them have never even deployed a service mesh or care about a service mesh.

[00:15:01] WM: Yeah. I’m the idiot in that exchange where like that's been living in my head rent free. But maybe they should feel proud, because that actually kind of spurred me on. I was like, “It doesn't seem right.” I’m like, “I'm going to prove that person wrong.” So here we are.

[00:15:16] JM: Service mesh adoption is still super early, right?

[00:15:20] WM: I'd say in the grand scheme of things, yeah, it is. It is. Because even in the grand scheme of things, I think Kubernetes adoption is still pretty early, right? And like service meshes trailing, like very much trailing with that and kind of requires Kubernetes to get into place before a service mesh can really be effective, right? And there's this weird symbiotic – Well, somewhat symbiotic relationship, right? Where the reason why – And I wrote this – I wrote a meshifesto like a couple years ago where –

[00:15:50] JM: I remember. I was ground shaking for me.

[00:15:52] WM: Oh, good. Okay. Wow! Wow!

[00:15:55] JM: Okay, that's exaggeration. It was great, though. It was great.

[00:15:58] WM: Okay. It was.

[00:16:01] JM: I think we did a show around that time.

[00:16:04] WM: Yeah, that's right. But one of the points I made was that the reason why the service mesh is possible is basically because of Kubernetes. Because it allows you to do things like, “Hey, I want to deploy 10,000 proxies, and I kind of want to just do it everywhere,” and not have that be like an insane statement, right? You can do that. It's not trivial. It's not like Ctrl+C, Ctrl+V. But you have that kind of power. And that has allowed the service mesh, which has this very weird deployment pattern of like tens of thousands of proxies everywhere to actually be a plausible way of deploying these features.

[00:16:41] JM: Have you looked seriously at HashiCorp Nomad?

[00:16:45] WM: No.

[00:16:46] JM: That's the answer that most people give. I haven't either. I'm going to do a show on it soon. But they're still working on it. They're still pushing it. They haven't given up, which I like. Think about like if I'm going to blindly pick an open source infrastructure tool from HashiCorp or Google, I'm honestly not sure which one I want. HashiCorp has such a good track record at this point. Vagrant, Console, Terraform. That’s it, right? Those three, those are the big ones. Vault, all of those are extraordinarily successful. HashiCorp doesn't really seem to miss.

[00:17:26] WM: Yeah. They have a really – What is amazing to me is they have such a great product sense. When they're building these things, like they know how to speak to the engineering audience that is going to adopt this, right? That's going to form the basis of the open source adoption. I also think they're really interesting from the open source perspective, because the CNCF for a long time was like, “Okay, these multi-vendor projects are like – That's kind of the model, right? That's what we want, right? That's how Kubernetes is. You got a

steering committee, and you've got the maintainers. And they all come from like these 12 different companies." And HashiCorp took a very different model. Yes, it's open source, but like it's open source in a very different way. We control everything, and like you can contribute. That's nice. Thank you. But like we're not going to let Red Hat come in and have a seat in the Vault steering committee or whatever." It sounds like, I guess, a nice contrast for us, because the Linkerd model is also, I guess, by accident, more than by design is very focused around Buoyant and doesn't have the Red Hats and Microsofts and Google of the world sitting in on the steering committee and kind of participating, which is another interesting topic to talk about.

[00:18:41] JM: While we're on the subject of open source, single vendor versus multi-vendor, and tinfoil hat related conspiracy theories, I don't know if you've seen any of my campaigning around React. Have you?

[00:18:58] WM: Absolutely.

[00:19:00] JM: What do you think of my perspective there?

[00:19:04] WM: This is a perspective that Facebook is going to change it from underneath us?

[00:19:09] JM: It's basically the perspective that React has become as important to the frontend as Linux is to the backend. And if you had a modern day Linux, it would basically be of national security level interest that this became a big multi-vendor collaborative project, and therefore we must push React to do the same.

[00:19:33] WM: Interesting. Okay. I had seen some tweets, but I hadn't seen those tweets.

[00:19:38] JM: Yeah, I mean, everybody uses React, right? It's the Linux of the frontend. Do you really feel comfortable with Facebook owning the project that controls all your frontends?

[00:19:49] WM: Yeah. I mean, I think that is a really good question to ask. And I think there are many related questions where, I think, I don't know if it's for the first time in history. But it sure seems like there is large portions of people's lives are often controlled at least nominally by specific corporations, right? Google and kind of the monopoly on like kind of search, right? And

like the basic way that people get information is either through Google or through Facebook, right? That doesn't seem great.

[00:20:24] JM: No, it's not. Actually, it's actively dangerous. It's very, very dangerous.

[00:20:30] WM: Yeah. So I think I'm sympathetic to that argument. I probably don't have anything really intelligent to say. I spent all my waking hours like smelling service meshes, inhaling deeply the surface mesh fumes. And like I guess there's a world of websites out there that have frontends and things.

[00:20:51] JM: Maybe if you could put me in touch with – If there's any Twitter OGs that are still they're working on a frontend stuff, I'd love to talk to some people about this, because they'd have a nice little dog in the fight. But just to pause a little bit more on this question. I mean, just ideologically speaking, it doesn't make you uneasy that Facebook is essentially able to layer in, whether it's open source or not, they're able to layer in a piece of infrastructure that they control into pretty much every other prominent technology company.

[00:21:24] WM: Yes, yes. Yeah, when you say it that way, yeah, it sounds terrifying.

[00:21:29] JM: Yeah, exactly. Exactly. And by the way, like nobody understands how they do state management. So React Hooks, or this big mysterious state management thing that people kind of understand how it works. But I seriously doubt that there have been a lot of people who have gone so deepest to vet React Hooks for any and all security flaws. And I just think if there's one open state channel in React, then that's a huge problem. If you have if you have an open state channel in React, that basically means that there's the vulnerability to have your entire UI changed by rogue actors, right? And we know how insecure the browser is. It's terribly insecure. So it's kind of bone chilling to imagine a vulnerability on your frontend that leads to your UI being potentially open to – I call it a UI attack, where basically your UI can be changed by rogue network elements on the page.

[00:22:29] WM: Where does that rank with you among kind of a similar set of threats around like the browser itself? Chrome had some huge portion of web traffic, and Firefox below that,

and Safari. And also, on the other side, like Facebook's just kind of control of information dissemination in general.

[00:22:53] JM: The problem with Facebook is that they do so much that is clearly deceptive, and mischievious, and malevolent that they've shown time and time again that they're not trustworthy. And it's kind of hard to encapsulate that. We really do need like legal proceedings. We need some massive legal proceedings to scrutinize the company for all the stuff that they've done. Obviously, Google does its fair share of stuff too. But you and I both know that Google is just much more well-respected from the departments of honesty and integrity of the Internet, right? It's just widely known. It's widely agreed upon that Facebook's done enough stuff. And this is coming from somebody who spent two and a half years writing a laudatory book about Facebook. My book about Facebook is completely laudatory. I love the company. They're inspirational. I use their products every single day. I love them. I'm addicted to them. And I love React, by the way. But, again, like I kind of like Coca-Cola. I don't drink it that often. And I know Coca-Cola is kind of a nasty company in a lot of ways, right? Like they pollute places, and they basically poison children, and poison older people too. And it's just – I don't know. It's just like, ultimately, you don't really want Coca-Cola in charge of the nutrition infrastructure. And you don't want Facebook in charge of your frontend. I feel much more comfortable with Google controlling Chrome. I would not use a browser controlled by Facebook, right? Like I wouldn't trust that at all. I wouldn't trust that at all.

[00:24:24] WM: Yeah. For you, it's like the fact that it's Facebook specifically. It's less about like there's a single –

[00:24:30] JM: I mean, Facebook has an internal thing called push karma, where basically the developers get a karma status based on how many times they push like a broken build. If you push a broken build, you lose push karma. It's sort of the internal developer social network. If Facebook itself had some sort of push karma relative to the entirety of the Internet, they'd be very, very low ranked, because Facebook has broken so many things for so many people, including sanity, right? And including integrity. It's so obviously a deceptive and vaguely evil company that they need to be prevented from controlling the Internet infrastructure.

[00:25:11] WM: For you, did this fight start like after you wrote the book or during the book?

[00:25:20] JM: So I went to DEF CON. I went to DEF CON a week or two ago, and I got massively hacked. Because you go to DEF CON to get hacked, right? I took three cell phones, and I took a laptop, and all my devices got hacked. And I went through some horrifying hacker victim experiences. But that's ultimately why you go to DEF CON. And I have theories about how I was attacked. I was attacked – Again, I describe as a UI attack. My user interfaces were changed on the fly in very creepy manners. If you can imagine somebody essentially taking over your native applications and changing, like repainting the UI in order to horrify you in a targeted way. That's what I experienced. It basically brought me to tears. And it sort of just made me realize that security is probably the thing that I have been under reporting on the most in the entirety of Software Engineering Daily. I've been focused on the cool companies like Buoyant, because that stuff's fun. It's light hearted. It feels like a game. But there are elements of software engineering that are not a game. And that's like in the integrity of your software. Honestly, not to sound like a conservative person, but there're norms, right? There're norms around what you can and can't do in the software community.

And the Linux Foundation is kind of the closest thing we have to the cabal of super friends, right? Where everybody comes to the table and just says, “Look, what can we agree on? What can we actually agree on as a software community?” The Linux Foundation has got to be the closest thing. Like it's not the EFF, right? Who comes close to the Linux Foundation in terms of impartiality, right? Is there anything else?

[00:26:56] WM: I mean, EFF was the one I was going to suggest.

[00:27:00] JM: EFF does not make cogent arguments around technology. They're like not engineering arguments. They're polemic.

[00:27:07] WM: Interesting. Okay.

[00:27:09] JM: So I just feel like I don't think Linux Foundation is by any means perfect. But it is the best thing we have in terms of arbitration of norms around software, which we do need as a community. And Facebook has just shown repeated violation of those norms. So we essentially need to bring them into the UN of software and put them under trial. And I think that includes

forcing them to donate React to either the Linux Foundation or an independent organization that is like the Linux Foundation. And if they don't, then we need to fork it. And we need to just do a groundswell adoption of that forked React version. And if nobody else does it, then we're going to do it.

[00:27:50] WM: What's been the reaction to this? I saw tweets, but I didn't spend too much time.

[00:27:56] JM: Yeah, so first, the “open source Facebook community” was engaging with me. And then they just went silent, because that's what they do, right? They're ultimately kind of cowardly. They can't engage, because they don't really have a valid defense, right? They can't really tell me that their software is secure, because they know it's not, because they know the web isn’t secure, right? So okay, look, I grant you. The web is insecure. Therefore, if you build a JavaScript framework and you make it the dominant JavaScript framework, that's also going to be insecure. That's fine. But again, if you become dominant, you're going to get regulated. That's the way the world works. That's what we need to do. And this is not a realm where we can impose any trust laws. So the closest thing we have is force you to donate your project to the Linux Foundation. Say what you will about Linus Torvalds. The guy has a kind of integrity, and his foundation also has a kind of integrity.

[00:28:57] WM: Interesting. Have you talked to the Linux Foundation about this? Are they like onboard with the idea?

[00:29:04] JM: I don't know. I mean, I'm just like tweeting at people, and like my in my inbox is overflowing right now, and I’m like trying to start some technology companies, and just like, in the meantime, I'm suddenly waking up feeling irritated about Facebook. I don't really know why it's happening. But it probably has something to do with pandemic and lockdowns and stuff also just irritating me generally. But yeah, I mean, you see this company in your life for like however long it's been around, 16 years or whatever, and you just recognize at a certain point that this company keeps bugging me in various ways. It just keeps irritating me. And it doesn't matter that all of those irritations are legal and that I could technically opt out of all of these products. It just doesn't matter, because the way that they're acting is so inappropriate and so offensive to me as a human being, as a software engineer, as a – I don't know, as a person who sees their

family corrupted and divided by their greedy technology. I don't know. It's frustrating. I don't know. Does that resonate with you at all?

[00:30:07] WM: I don't use Facebook. So I'm probably inoculated in that –

[00:30:10] JM: I mean, why does Twitter feel so much more pure compared to Facebook?

[00:30:17] WM: Oh, gosh, I don't know. You mean in terms of like the ethics of the company? Or in terms of the like the Twitter experience and who you're interacting with and like –

[00:30:25] JM: Kind of both. It's like I know that both feeds are algorithmic, and yet I trust Twitter 10X more.

[00:30:32] WM: Yeah. Yeah, that is interesting question.

[00:30:36] JM: And by the way, for those who don't know, you worked at Twitter for four years.

[00:30:40] WM: I did. Yeah, that's right. Although I was like a cog in the wheel. I wasn't like up there making strategic decisions. In fact, mostly, when I was there, Twitter was characterized by the fact that so many people would come and go that like there wasn't even kind of the staying power to assemble any kind of long-term strategy. So maybe that's the secret is that because there's been so much churn among the founders and everyone else, no one has been able to actually enact any kind of evil plans.

[00:31:09] JM: Yeah. Right. It does feel kind of leaderless for better or worse.

[00:31:12] WM: Yeah. I mean, Jack is at the helm. And he did that after I left. And he's been there at least 50% of the time. And I have opinions. But these are now the opinions of someone who has been out of Twitter for much longer than I ever was in Twitter. So yeah, to answer your question, like neither I nor my immediate family members are actually really on Facebook all that much. So I get inoculated from a whole lot of this stuff. Obviously, I'm exposed to React kind of everywhere I go. Some of your statements are worrisome from that.

[00:31:47] JM: That's the thing. That's the thing, is you're not on Facebook, but Facebook's on you. I agree, we don't have the Chinese social credit score, but are we sure that we don't have the Facebook social credit score?

[00:32:02] WM: Yeah, yeah. Well, I'm sure my score is close to zero. Yeah.

[00:32:07] JM: It may not be. It's probably not zero. It's probably either negative or positive, right? Or assuming zero is somewhat neutral.

[00:32:15] WM: I think these are great questions. Yeah, I'm glad someone's asking them.

[00:32:18] JM: And it's creepy that we can't answer them.

[00:32:22] WM: It is creepy. And I think it's an interesting class of –

[00:32:27] JM: And by the way, who else can ask these questions, right? Is it really the software podcaster? It's the only guy that can ask these questions?

[00:32:34] WM: Yeah, right. I don't know. Maybe that's where it has to start.

[00:32:41] JM: Because most of the journalists I see who tried to report on this, they get so wrapped up in these blatantly subjective arguments about like ethics. I mean, I guess that's what I'm doing also, but maybe they're just less convincing arguments. I don't know. I feel like a lot of the journalists get wrapped up in the same kind of polemics stuff that I haven't heard people make – Well, I guess there are valid criticisms that people have made against Facebook. I mean, I feel like the most valid one that you can jab at is React is not open source, right? And that, to me, just seems like – This is a place to start. Like I've kind of decided that I'm going to interrogate this company as a journalist, if I am a journalist, whoever the fuck I am, but I'm going to interrogate this company, because I wrote a book about them. And I love the engineers there. I would even say I really like Mark Zuckerberg, and I respect the guy, but I think he's – I don't know, he just seems kind of rapacious. And whether or not that's like kind of a joke, you have to take it seriously, because the guy has so much power. He's got an unprecedent amount of power. And, to me, it's pretty serious. It's pretty creepy and serious. Because if you just look at

the kind of behavior that has been essentially enforced and propagated by social networks over the last 18 months, whether or not that behavior has been justified, that behavior has been instigated by social media, which is eerie.

[00:34:08] WM: The other thing I really don't like about Facebook is they don't use Linkerd.

[00:34:13] JM: Are you sure? Do you do even know? Do you even know? They probably forked it?

[00:34:18] WM: Yeah, probably. Probably fork it and messed it up, man.

[00:34:22] JM: Probably. They wouldn't call it a service mesh. They would call it a service network or service social network.

[00:34:29] WM: I think we did talk to some Facebook engineers in the very early days of Buoyant, and there were equivalent to Linkerd that they had in place, which maybe now have been totally replaced. The pattern that Linkerd plays is a common one, especially for those companies that built those microservices architectures early on. But in terms of like, yeah, if they were heavy LinkerD users, maybe we could help you take them down somehow.

[00:34:56] JM: Oh, yeah. Oh, yeah.

[00:34:56] WM: Sadly, I don’t know how much power I have.

[00:34:59] JM: No. No. No. No. No. We can threaten to increase in their LinkerD license fees if they don't release React to the Linux Foundation.

[00:35:08] WM: We’ll have to quickly make a LinkerD license. Yeah, I'm sure this is where the Linux Foundation will start getting involved.

[00:35:14] JM: Okay, let's talk about like enterprise software sales and stuff.

[00:35:18] WM: Oh gosh! Oh! So much more boring. Okay.

[00:35:21] JM: Or we don't have to. I don't know. Hey, by the way, are you going to KubeCon LA?

[00:35:26] WM: My current plan is to go, but there's a big question mark that seems to be growing bigger every day around Delta variant. And is there going to be an in-person KubeCon or not? Right now, last I heard, Hybrid is still the plan. And I'm like desperate for an in-person event.

[00:35:41] JM: Oh, me too.

[00:35:43] WM: But yeah, I'm kind of watching –

[00:35:45] JM: Listen, if it gets canceled, let's just rent like a Luby's cafeteria and have a mini KubeCon at the cafeteria in LA.

[00:35:53] WM: That's a great idea. I don't know if they have Luby's in LA. Also, Luby's went out of business.

[00:35:59] JM: Probably.

[00:35:59] WM: Yeah, I think it did recently. There's like –

[00:36:02] JM: Why did we like Luby's? Why did anybody like Luby's?

[00:36:06] WM: I mean, I went there when I was a kid. And that's why I think that your brain gets programmed to like things that you experience when you're kids. Yeah, I don't know if it was ever any good. Never went back when I was an adult.

[00:36:21] JM: Yeah, it was fried okra, macaroni and cheese, chocolate milk, fried chicken. It's pretty disgusting. Black eyed peas, pretty disgusting. Jello pudding for dessert.

[00:36:33] WM: Cod stick.

[00:36:36] JM: God! Technology is not the only thing that's improved over the last, whatever, 25 years.

[00:36:42] WM: I think our understanding of nutrition has improved a fair amount.

[00:36:47] JM: By the way, I think I sent you an email. Did you go to Schlitterbahn yet?

[00:36:51] WM: No. It's on the list. I haven't been there yet.

[00:36:54] JM: Oh, okay. What about Enchanted Rock?

[00:36:57] WM: No. We haven't done anything. We just like stayed in our house, man.

[00:37:01] JM: Enchanted Rock is outdoors. It's an outdoors place.

[00:37:04] WM: Oh, all right. Well, that's **[inaudible 00:37:06].**

[00:37:07] JM: Enchanted Rock is awesome.

[00:37:07] WM: It's also like a thousand degrees here in Austin.

[00:37:12] JM: You better get used to it. You moved there.

[00:37:14] WM: Well, I'm going to wait for it to cool down a bit before I go outside, I guess.

[00:37:19] JM: What about stand up paddleboarding?

[00:37:21] WM: Looks really fun.

[00:37:23] JM: It's so fun. It's not even hard. It looks hard. It's not even hard. I did it a few weeks ago. I did it for the first time a few weeks ago. I never did when I was in Austin. But now, whenever I go to Austin, I'm going to stand up paddleboard. It's amazing.

[00:37:34] WM: We kayaked down the river, and that was a lot of fun. I saw people stand up paddleboarding, yeah. It's easy?

[00:37:41] JM: It's actually easy. It's actually as easy as kayaking and much more fun.

[00:37:45] WM: Great. Kayaking was easy. Was it fun? It's fun-ish.

[00:37:49] JM: Are you in a in a town lake adjacent area or like a further from town lake adjacent area, or further from town lake area, or Ladybird Lake I should say?

[00:37:57] WM: It's probably 10, 15 minute drive from where we are.

[00:37:59] JM: Okay. Okay. Got it. Got it. I feel like North Austin is the best place to set up shop these days. South Austin is pretty good to Central Austin is super congested. Anyway, what should we talk about? What's Buoyant related that we can talk about?

[00:38:16] WM: We can talk about enterprise sales and stuff, man, if you want to? I'll try and make it exciting.

[00:38:20] JM: I mean, what does what does like a large Linkerd deal look like, or Buoyant deal look like?

[00:38:26] WM: Yeah. So for us, the thing that –

[00:38:29] JM: It's all bottoms up, right? They install the agent. Some stuff happens. And then eventually they realize, "Okay, we need actually best in class enterprise support."

[00:38:37] WM: Yeah, that's right. As much as possible, LinkerD adoption as bottom up. That's by design, and it's also like that's the model that I understand. It's the model that works well with

like open source infrastructure. Possibly the only model that makes sense with open source infrastructure. So the trick for us has been, “Okay, with Linkerd, we want to give you – We don't want to hold back any features. We want to give you everything you need to run this thing in production.” And it should never be like I could only get this far. And to go from here, I have to buy Linkerd+ or whatever. That's just not the model that we want to do.

But then it turns out, there are actually these challenges, very specific challenges for like enterprise operation. Like one obvious one is like, a lot of times, companies will not adopt open source unless there's support in place. Okay, great. Well, like that's something that you can provide, and that's something you can and should charge for, because it's not like a free service. Human beings have to provide that support. And then there's other features, too, that are like, “Well, gosh, we've got this weird security, like the set of security requirements, and like the open source doesn't really quite match into that. It’s not because Linkerd is insecure. It's just because we have these requirements that have been developed in a very specific way. Can you give us a way of meeting those requirements with Linkerd?” Okay, great. Like that's something we can do.

And then the stuff that I get excited about is when people really start adopting Linkerd and they're like, “You know what? This is awesome.” And it's great that it's so like simple, whatever. But we don't actually want to do any work here, right? We want to have the service mesh, and we don't want to operate it. We don't want to have a person who we have to worry about if they go on vacation, or whatever. So can we actually allow you to operate this for us? Can you like take over? And that's where things get really interesting for us because –

[00:40:29] JM: Self-driving. Self-driving –

[00:40:31] WM: Yeah, that's right. That's right. And you look at open source and the business models that make sense, like the one that makes a lot of sense, especially in the modern world is like the SaaS kind of hosted model, right? We'll host this product for you. For Linkerd, we can't really host it entirely for you, because there are bits that has to run on a cluster. We can host a whole bunch of it for you, right? And then we can make it so that operational burden, which is a real cost to software, right? That comes on us. And we're experts, by the way. We're really good at running Linkerd. And then you get the value, right? And you pay us money, which

is good for everyone. But we're incurring the operational burden, right? And that way, you as a company can focus on what you really want to focus on, which is not being service mesh experts, right? You want to focus on your business, right? What are the things that you should be doing to advance the mission of your company, to advance your agenda, to advance your products and to serve your customers? And unless you're a company like Buoyant, that is not the service mesh, right? Being service mesh experts is like should be no one's job except for companies that are selling service mesh expertise, which is like a handful. So that's kind of where we end up being able to provide commercial value. It's not in like the raw technology, but it's like how do I fit this technology into this existing business structure that has these constraints and it's not trying to become a service mesh expert anyways? Moments of profound – When you're exposed to moments of profound genius, sometimes it takes –

[00:42:05] JM: I know. No. No. No. I'm just right-sizing my question for the next six minutes. Time always flies with you. I hope we can do this again in LA, by the way. So let's talk a little bit about like kind of go to market strategy relative to fundraising in the last six minutes. So you've been around for more than six years, and you've only raised an A, which is unusual. Just tell me about your fundraising strategy, or your just burn rate strategy, because you've clearly kind of taken your own path. This is a little bit different than the typical 18 month burn rate strategy. What do you think in there?

[00:42:46] WM: Yeah. So fundraising is not a goal for anyone. It should never be a goal. It's a means to an end, right? What you are doing, what you should be doing, is trying to build a sustainable, large scale business, right. And that's is my goal with Buoyant, is to build a company, IPO it, have a massive impact on the world in a positive way. Don't do any of the Facebook stuff. And fundraising is one tool along that path. Other tools include like making money, generating revenue. For us, for open source especially, I think the model is like every 18 months, we fundraise, we fundraise, we fundraise. I think that makes sense for SaaS companies, or at least that has come out of like the world of SaaS companies, because you have a burn rate that you're operating, you're paying your AWS bill and whatever else. Like you almost have to do that, right? And you're being judged by these all scientific standards at each point, right?

For open source, it's really different, and it's much less of a science. It's much more of like you've got some of that consumer magic in there. Like you don't want to get the lightning in the bottle. You want to see that thing spark, right? Either the community is like growing rapidly and self-perpetuating, or it's just like shriveling up and dying. And so early on, especially, we kept the burn rate super, super low, because what we were doing was not like anything revenue generating. I was like investing in the open source community. Okay, that's going to pay off in the long run, but that's a long term play. Doing a long term play, and like you don't want to end up hitting against some short term walls. So that's really what it comes down to. For open source, especially infrastructure open source, it takes time for that to happen. And we're seeing it now with Linkerd. We just graduated in the CNCF. It's like the top tier of project maturity. So it's up there with like Kubernetes, and Prometheus, and Envoy. And like the community is going gangbusters. We're having people come to Linkerd from Istio like in these very public ways. So all of that is just happening right now. And that has a very different dynamic from where we were in the early days when a lot of what we were doing, like I was saying, isjust investing in community with the idea that it was going to pay off in a couple more years. We took all that beautiful money. And we've been very fortunate and that it's been relatively – Relative to other companies, it has been easy for us to fundraise for a variety of reasons. Obviously, my visionary genius just kind of awes and astonishes anyone who ever talks to me. So like –

[00:45:26] JM: You know, you're deflective. But everybody that knows you knows you're good.

[00:45:31] WM: Okay, thanks.

[00:45:34] JM: I mean, come on, how many companies have stuck around for six – Whatever, six years and change? I'm sure you could have sold by now, right? I mean, you must have a bigger vision, which is basically networking and beyond it. Your TAM is just gigantic. It's really, really big.

[00:45:55] WM: Yeah, that's right. I think what we're doing with Linkerd is so fundamental to how people build today and build future software.

[00:46:02] JM: And the thing is, like today, it looks like, "Oh, I have to integrate with –" Or maybe today, it doesn't even look this way. Three years ago, four years ago, whenever we start

talking about this stuff, integrate with Linkerd. Eventually it's going to be like one click, you choose Linkerd. Or one click, choose Istio. Once it's at that point, then it's a UX thing. And which UX do you want the most? Like at this point, I mean, I love Varun. I just talked to Varun over at Tetrate. I think he's kind of got a different demographic for who he is marketing to. You look at the Tetrate website versus the Buoyant website, it's quite obvious they're marketing to a different kind of customer. I mean –

[00:46:42] WM: Yeah, I think it all relates to kind of my core hypothesis, which is that the future of software adoption even in the enterprise is bottom up adoption. If you don't have that sorted out, and if you don't have that in a place that you can control it, then you're going to have a short term company, not a long term company. And I think that's a big difference. This is, again, another thing I really liked about HashiCorp is they control – The projects are open source. But they control that core technology, right? For the most part, Console Connect uses Envoy and like they have no control over kind of that. But in contrast to the Istio vendors, like they don't have any real control over the core technology, right? They're kind of at the whim of Google's roadmap or whatever, IBM's roadmap.

[00:47:28] JM: In here, just to bring it back, I know we're out of time, basically. But I think it's worth pointing out that if HashiCorp had React.js, if HashiCorp made React.js, I don't think we would have much of a problem here, because everybody trusts them. It's just simply a matter of trust. You look at those guys, you look at the way they've operated their company. You can listen back to their hundreds of podcast interviews, and you know that they're trustworthy.

[00:47:52] WM: There's a deep thing here where the way that you become a successful open source company involves having a very trusting relationship with your engineers and with your adopters. And that's very different from kind of the standard consumer relationships. Like I have cellphone service from Verizon. Do I trust Verizon? No. And like I don't trust them to ever look out for my best interest. But if I'm adopting like open source technology, if someone adopts Linkerd, they're not going to do that if it's sketchy, right? Like you're running this code on your system. So there's a very different kind of relationship that you have to develop with your adopter in the open source world. That comes down to trust. I think there's something profound there.

[00:48:34] JM: Okay, last question, just because we didn't really tackle any tough engineering stuff. What's the hardest technical problem that you've had to work on since we last spoke? I think I talked to you last you were in that office, the top Market Street. So that was probably two and a half years ago or something like that. It was a long time.

[00:48:50] WM: Yeah, hardest technical challenge.

[00:48:53] JM: Yeah. And I think back then, by the way, your biggest challenges were UX related, because you were working on the service social network. Which is a huge product category that we don't even had time to talk about today.

[00:49:06] WM: Right. Yeah, I think for us, the hardest technology challenges continue to be at the proxy layer for us. Linkerd is very unique. It has this RUST-based micro proxy just called Linkerd proxy. I've written articles about this. It's like it's very different from the Envoys of the world. And it's very specifically designed to be different, because a lot of the complexity in many other service meshes comes from their choice of Envoy, not because Envoy is bad, it's because Envoy is complex, right? So we want to avoid all that. So we have this amazing proxy, but like that is hard technology .You need to be a serious technologist in order to make this proxy that is like ultra-lightweight, and ultra-fast, and secure, and sitting at kind of the very leading edge of RUST, and like network programming, and Kubernetes, and like running things containerized environments. That is a very, very advanced – I would say, and I have said, I think Linkerd 2 proxy is the most technologically advanced project anywhere in the CNCF. I think it's just like the brain power that goes into that thing is really astonishing. So that's not a specific challenge. That's like a continuing challenge.

[00:50:20] JM: All right. Foreshadowing for either our KubeCon LA talk, or we can do another remote episode, or if you want to pair me if you got some engineer who's been cracking on that stuff. I'd love to go deep on those networking challenges.

[00:50:32] WM: Yeah. Oliver would be the great person.

[00:50:34] JM: Oh, yeah. Yeah, it's been a while. That's great. You guys have stuck together for six and a half years.

[00:50:40] WM: Best buds.

[00:50:41] JM: Have you guys gone to like a relationship count – Actually, I shouldn't even ask this question. I won't to ask this question on air. I'll just say, if I had done six and a half years with a cofounder, I probably would have gone to – Seriously, I probably would have gone to a therapist with them at some point. I'm sure you get into enough vociferous arguments that –

[00:50:58] WM: I think it's to Oliver's credit that he's managed to survive six years with me. It's definitely –

[00:51:05] JM: But you basically have to be mercurial. If you're a successful founder, you have to be mercurial, right? Who not? What founder is not mercurial?

[00:51:14] WM: Interesting. Never thought about that. I've been trying to be honest and transparent. But I guess I could be mercurial.

[00:51:23] JM: You could be all three. William, pleasure as always, my friend. I hope to see you in LA, seriously. And if not, we'll do another show.

[END]