**Voiceover:**

Welcome to What Matters, a podcast from the folks at Mattermost. We'll be discussing ChatOps, open source, DevOps, and everything that matters most to you. Let's see what we're chatting about this episode.

**PJ Hagerty:**

Hey everybody and welcome to What Matters, the podcast from the folks here at Mattermost. I'm PJ, your host as always. For this episode I'd like to introduce you to my friend, Kat Cosgrove of Pulumi. Kat, tell us a little bit about yourself and what you do.

**Kat Cosgrove:**

Hey y'all, I'm a staff developer advocate at Pulumi, which means that I do a lot of teaching people how to use infrastructure as code, how to use Pulumi specifically. But I haven't done this always. I spent a lot of time more broadly in the DevOps space as a developer advocate before this. Before that even I was an embedded Linux engineer, which is a really weird technical journey. I also used to work as a database administrator for an independent video store and as their resident horror expert. So I used to get paid to sit around watching horror movies and babysitting an uncomfortably large access database.

**PJ Hagerty:**

Okay. I was going to ask if it was an access database-

**Kat Cosgrove:**

It was access. Yeah.

**PJ Hagerty:**

Yeah. Video store? Yeah.

**Kat Cosgrove:**

Yeah.

**PJ Hagerty:**

Very specific. Yeah.

**Kat Cosgrove:**

Yeah. This place is still open though. So they're taking long-

**PJ Hagerty:**

Are they're still using access though? Probably.

**Kat Cosgrove:**

Yes.

**PJ Hagerty:**

It's not the oldest open blockbuster in the world, is it?

**Kat Cosgrove:**

No. It's Black Lodge Video in Memphis, Tennessee.

**PJ Hagerty:**

Okay. Okay. So shout out to Black Lodge Video in Memphis, Tennessee. So let me get started. So you mentioned your career has been majorly focused on DevOps, especially the last few years. What drew you into the ideas behind DevOps? What made you say DevOps, this is the way to go?

**Kat Cosgrove:**

How honest do you want me to be?

**PJ Hagerty:**

100% honest.

**Kat Cosgrove:**

Money.

**PJ Hagerty:**

Okay. That's fair.

**Kat Cosgrove:**

Yeah. It's money. So there's a lot going on in the DevOps space, especially once you start overlapping it a lot with cloud-native as a concept. There's a ton of money there and I like not being broke anymore. So that goes a long way but from a not financial standpoint, I'm also one of those engineers who's really, really lazy, extremely lazy. And my laziness manifests in a way where I am willing to spend a ton of extra time and effort at the beginning of a project to make sure that I don't have to do a bunch of boring repetitive shit later on. So automating away things that are personally annoying to me as an industry is really cool. Because this is the way I develop now is not the way I developed six years ago. I do way less work and produce a better product that scales more easily and falls over less easily with considerably less effort. And that's a fucking plus for me.

**PJ Hagerty:**

Do you think that it almost became too easy? Do you think DevOps makes things too easy?

**Kat Cosgrove:**

Sometimes.

**PJ Hagerty:**

Yeah. I feel like there's no universal like, "Oh yeah, 100%. Everything works all the time." It's great.

**Kat Cosgrove:**

Yeah. It's a sometimes thing for me. The first time I used Netlify for instance, I was like, "What the fuck? This can't be this easy. This can't have just abstracted away seven other fucking things that I would have to do myself before into functionally a button click." That's wild. But it is that easy. It does work that well. And that's terrifying because for me, if some part of that stack falls over, I understand what makes up that stack and I can go fix it. But there is a little bit of danger in abstraction there where new developers or people with less experience, people who grew up on this don't understand what's underneath those abstraction layers might have a harder time debugging an issue because they don't know how to write react for instance. So they're just doing everything through the Netlify console.

**Kat Cosgrove:**

I had the same experience the first time I stood up a cluster on EKS with Pulumi, where it was, I don't know, 80 lines of Python for a cluster with as many replicas as I want and deploying my whole ass front and backend website into it and different containers. And it just went immediately the first time I ran my code and it was deployed and online in three minutes. And I didn't trust it.

**PJ Hagerty:**

Yeah. That's the key word I was waiting for. Especially older developers, a lot of times you're just like, "I don't trust this thing. It's too easy. Why? I'm not breaking my brain. Nothing's falling up over too, a bunch of red on my screen. How can I trust this thing?"

**Kat Cosgrove:**

Yeah. Like I typed Pulumi up and then it went, okay, 18 resources created. And I was like, "No."

**PJ Hagerty:**

No, that can't be it.

**Kat Cosgrove:**

No, something's wrong. This is failing.

**PJ Hagerty:**

Clearly something's not working here.

**Kat Cosgrove:**

Something's not working. Something is failing silently. I refused to accept that it was that easy. But no, there fucking was. And that still messes me up sometimes. But yeah, that's the direction DevOps is going in. And I have a love-hate relationship with it because yes, it's great that it's that easy but also when something goes wrong, because something is going to go wrong, figuring out what is wrong underneath seven layers of abstraction can be really difficult if you did not experience the previous iterations of this stack or this tooling, right?

**PJ Hagerty:**

Right. Right. I think this is the source of the argument of, for lack of a non derogatory term, the gray beards are like, well, they'll all learn their lesson when they realize that they've abstracted away their ability to see what's going on.

**Kat Cosgrove:**

Yeah. My dad does that. He is a gray-beard software engineer so he does that to me. And sometimes he's right. He is.

**PJ Hagerty:**

Yeah. We should respect the previous generation. They did give us a lot of great tools that-

**Kat Cosgrove:**

And some terrible ones.

**PJ Hagerty:**

And some terrible ones. Yeah, it a mess and luckily some things rose out of the pile. I've worked at some of those terrible companies. Anyway, because you a couple times, but just to make sure people understand what Pulumi does, let's start answering the question, what is infrastructure as code? I think the term gets thrown around a lot but not everyone really understands what it means.

**Kat Cosgrove:**

So when we say infrastructure as code at Pulumi, it's probably more accurate to say infrastructure as software because when you're using Pulumi to define and configure your infrastructure and throw your app into it, you are writing actual code. You are not writing a DSL, you are writing Python or Go or TypeScript or C# or whatever to handle provisioning an EC2 instance and setting your IM roles and configuring a VPN or whatever. All of that is happening in actual code and it can live either right aside your application, which is what I do with all of my personal project and with all of my demo code for workshops. But it can also live in a dedicated infra repository that handles stuff across projects.

**Kat Cosgrove:**

But yeah, it's handling your infrastructure and bringing it closer to the developers rather than doing what we have been doing with DevOps, which is still keeping dev and ops in separate silos but with just a big window cut between the silos so they can wave at each other. We're making it easier for developers to have some direct insight into what the infrastructure looks like, and ops in general, and making it easier or for ops to have more insight into how this affects the developers and how

they write their code and how all of this works. Because we like to pretend that we have brought dev and ops together as a DevOps community, but we haven't quite got there yet. And this helps.

**PJ Hagerty:**

Yeah. Well, that's one of the things I think that you... I don't know if you've actually seen the talk that I do called DevOps: Philosophy vs practice. And the philosophy is always like, "Oh, the wall has come down." And everyone's happy, and everyone's working together in this completely automated world there. But in reality that's not-

**Kat Cosgrove:**

That's not true.

**PJ Hagerty:**

... totally the case with Dev. There's still issues with Devs being like, "Well, I hand it off and the infrastructure team takes care of it."

**Kat Cosgrove:**

Right. You still just throw it over the wall, right?

**PJ Hagerty:**

It's just how you're throwing it over the wall. You mentioned there's a little window where they can wave each other. Maybe some things are like, okay, it's more like a drive through window where each you can push the button and hand out the fries to the next group.

**Kat Cosgrove:**

Yeah. But they're still looking at big ass YAML config can still be really overwhelming for a developer who has never worked in ops before. I still don't like looking at YAML and I've been working in DevOps for years. And I know that somebody listening to this is thinking, "Ah, it's so easy. It's human readable." And that's true in a box for one project, but there isn't a consistent schema and it can be really easy to screw up. And I wouldn't directly compare it to JSON, which is way more consistent. And if I had to choose between YAML and JSON, I would choose JSON every single day.

**PJ Hagerty:**

I would not disagree with you. Which – I'm sure a lot of people listening, we may have just started a whole political argument repeat. We're going to get comments on that.

**Kat Cosgrove:**

I'll die on this hill. I'll die on this hill.

**PJ Hagerty:**

Okay. Okay. Fair. So we'll mention Kat's Twitter handle at the end of the episode-

**Kat Cosgrove:**

Yeah, one vs. one me on Twitter.

**PJ Hagerty:**

Yeah. Always recommended. Always go up against Kat on Twitter. I'm sure it's going to work out for you. It's going to be so great. When you're done with Kat go after Alexandria Ocasio-Cortez. Very quiet, demeanor girl, doesn't say much. But yeah. Oh, my God, we're setting people up for failure. Yes, this to get us back on track. The main goal is to bring like developers and teams and ops all together. And our idea is modern application deployment is the easier path to use something like Pulumi than can raw Kubernetes or OpenShift. It's easier to have something that takes that step for you, right?
Kat Cosgrove:

Yeah. To be clear, you absolutely still can use Pulumi to roll your own cluster. When I'm giving a live demo I am almost certainly not using EKS or whatever. I'm not using a cloud Kubernetes provider to do that because that's another failure point on stage, right? I am rolling my own cluster and using mini cube locally. So you can use Pulumi to do that, totally valid. But you can also use Pulumi to stand up a cluster on GCP for you and configure everything for you and just let the cloud provider do what it does. You just hand it whatever you want to run in that cluster, set roles and permissions, set access control and then just let it go, all in a programming language that you're probably already familiar with-

**PJ Hagerty:**

Well, hopefully if you're building an application, there's some language you're familiar with.

**Kat Cosgrove:**

Yeah. The Pulumi code doesn't have to be written in the same language as the application code. So you can be building a Rails app and still use Python with Pulumi to containerize it and chuck it on GCP.

**PJ Hagerty:**

Now that makes me want to ask the question, I'm a Python developer and I love it, but I want to learn Go. I know that Pulumi does things in Go. Do I need to be an expert in the language to use Pulumi in language or can I use Pulumi as a launching pad to learn something new compared to what I'm already building in every day?

**Kat Cosgrove:**

So all of our documentation and most of our examples have sample code in all of the languages we support. So if you know Python really well, you can probably just rip off the Go examples and look back and forth between the Go example and the Python example to just figure out what the syntax is. Of course, you're going to miss out on some idiosyncrasies of the language and what is considered the way to write Go versus Python. But yeah, you don't necessarily need to be an expert. Being comfortable with a language does make it easier because for instance, you may not understand whether or not an error when you run Pulumi up is coming from Pulumi or if it's coming from Node.js if you don't know Node.js relatively well.

**PJ Hagerty:**

Right. That makes sense.

**Kat Cosgrove:**

But otherwise, yeah, just code. Yeah. We have examples in all the languages. So if you know one language it's not terribly difficult to fill in the blanks between one language and another with the examples.

**PJ Hagerty:**

Cool. Let me ask this because this is my favorite questions to ask people. Do you think there are better alternatives to Kubernetes?

**Kat Cosgrove:**

Oh-

**PJ Hagerty:**

Knowing full well that you are on the selection committee for KubeCon. I'm going to ask the question anyway.

**Kat Cosgrove:**

Oh yeah. I guess I should have mentioned that in my bio [crosstalk 00:14:38].

**PJ Hagerty:**

I don't even know if you're allowed to say, but how many talks were submitted, they were like X an alternative to Kubernetes? How many of those get submitted to because I feel like that's got to be a heavy toll?

**Kat Cosgrove:**

So I am usually on the program committee for the 101 track. I was on the program committee for the 101 track this year. It's worth noting for the listeners that I am also a CNCF ambassador, and I have never seen a talk like that come up. We don't publish all of the titles or abstracts of talks that don't get accepted. What we do release after KubeCon is the total number of talks submitted and a breakdown by vendor or user or whatever. But we don't actually release the talk titles and abstracts for things that aren't accepted. And while I have that information, I'm also not allowed to talk about it.

**PJ Hagerty:**

Yeah. Okay.

**Kat Cosgrove:**

But I've never seen it come up. As for whether or not I think that there is a viable alternative to Kubernetes, not one that I've used personally. If you have something you would like me to a try that you have an argument for being better than Kubernetes I am more than happy to do it. But it's also worth noting that I think that CNCF ambassadors and people who work as much with Kubernetes as I do are probably not the right person for that question because-

**PJ Hagerty:**

Okay. I can edit this part out.

**Kat Cosgrove:**

Oh no, it's fine. It's fine. I have an interesting point here. The issue is that Kubernetes is easy for me to use because I've been doing it for so long, right? It's easy for me to use and that's not true for everybody else, not even true for most people. For instance, when I was first building the current version of my personal website, I tried to deploy it to AWS using Elastic Beanstalk and a whole bunch of others.

**Kat Cosgrove:**

I was using the code star thing, AWS service where it stands up the whole stalk for you like a Django app on code star. And I could not get that to work reliably. And this is supposed to be easy, right? This is supposed to be the easy way to de deploy website. Couldn't get it to fucking work reliably. So instead what I did was I fucking deployed that shit to Kubernetes on GCP. Putting my personal website that realistically probably eight people will look at a month on GKE is an insane over-engineering of something. But it was easier for me to do that because I understand Kubernetes really well.

Kat Cosgrove:

So what I think is a good replacement for Kubernetes as a Kubernetes expert is not necessarily what the average user thinks is a good replacement for Kubernetes. But if somebody has a suggestion for something that's it is easier for you to use in some way that also orchestrates and scales containers that well, please let me know, I would love to try it. But yeah, it's a thing where once you become an expert in a thing, it becomes difficult to remember what it's like to not be an expert, and that can color your expectations of other products. Similar to me finding Netlify-

**PJ Hagerty:**

[Crosstalk 00:18:05] before. Yeah.

**Kat Cosgrove:**

Yeah, yeah. I find Netlify alarmingly easy to use because that's not how I'm used to building and deploying a website, right? Same thing with "easier" replacement for Kubernetes. But I would love for something to make it more accessible because Lord knows I've had to put out enough fires because of people not quite understanding what a container run time is on Twitter.

**PJ Hagerty:**

Right. Right. Do you think that part of the success of Kubernetes and other similar projects is maybe because it's open source and really people have access to it? It's the plus and the minus all at the same time. You can do whatever you want

with it because it's yours and you can build it that way. Also you can do whatever you want with it because it's yours and you build it that way.

**Kat Cosgrove:**

I think that helps/hurts. It really, really does, which I don't think would've been true 15 years ago because now, open source is mainstream. It just is. It's fully mainstream, big ass multibillion-dollar, multinational corporations contribute to open source "own" open source projects and make a fucking shitload of money off of open source. But that wasn't true 15 years ago. 15 years ago open source software being involved somewhere in your company's business was alarming, right?

**PJ Hagerty:**

Terrifying.

**Kat Cosgrove:**

It was terrifying. We didn't trust it, which maybe sometimes you shouldn't, but we didn't trust it at all back then. And it was considered a bad sign. You don't have your shit together as a business if you're not doing something like closed source and proprietary. But now we've societally swung in a completely different direction. So the success of Kubernetes I think is not just because it's a useful product. Nothing else does it, what it does that well, but also it was a major case of right place, right time. If Kubernetes and containers had been a thing 15 years ago, I think Google would've kept Borg internal. It would've stayed closed source. It would've been a product that they sold not an open source.

**PJ Hagerty:**

Yeah. I've always compared that. I was always a big user of open source languages, but I've always compared the time example like Kubernetes right time, right place. And I always compare it to the programming language of Elixir. Elixir was wrong time, wrong place. It's a great language. It's super powerful, you could do so much with it and it would've been perfect in 2008.

**PJ Hagerty:**

It did really come out 2014 where I was like, "All right, we're already getting over Ruby on Rails," even though 90% of Silicon Valley still builds their stuff in Ruby on Rails. They're just like, "We don't need another language that does something that similar. Thanks, but we're going to pass." And it sucks because it's struggle to get that adoption and it probably never will.

**Kat Cosgrove:**

Yeah.

**PJ Hagerty:**

But Kubernetes didn't have that problem. I think there were some things that were similar to Kubernetes in the early Dockers coming out and becoming a thing days. I think there was some tools that started to look similar-

**Kat Cosgrove:**

Yeah. There was Dockers forum.

**PJ Hagerty:**

Yeah. Dockers forum, it was so resource heavy and cantankerous and it really wasn't... At that time Dokku was still competing with a lot of the platforms like Heroku and Engine Yard, they were trying to figure out, how did you really rack space? How can you deploy easily? And then it just happened that Kubernetes came around, those passes got left in the dust.

**Kat Cosgrove:**

Yep. It happens. And it's always it's weird and sad to look back at, you see this a lot with JavaScript frameworks there. It feels like for a minute there, every six months a new JavaScript web framework came out that was the hot shit. And I don't remember the names of fucking most of them. It was like being on a treadmill of JavaScript web frameworks.

**PJ Hagerty:**

I think you're being very gracious by saying every six months, because I used to do a show with Eamon Leonard who's been on this show before and also we used to work together at Engine Yard back in the day. And we used to do an internal show and we actually had a part called hipster framework of the week. And every week we would pick a new JavaScript framework that had just come out, and this is going to be the shit and everybody's going to use it. And we would point out what it doesn't do.

**Kat Cosgrove:**

I love it.

**PJ Hagerty:**

Or specifically picking one out, it's the most misogynist, sexist thing we've ever seen. Let's make fun of this for five minutes. It was the Wild West of JavaScript frameworks-

**Kat Cosgrove:**

It was weird.

**PJ Hagerty:**

It was super bizarre. They don't even match their names. This is crack and JS. What does crack and JS do? Nobody knows what crack and JS does.

**Kat Cosgrove:**

Is that-

**PJ Hagerty:**

That's so to do with PayPal.

**Kat Cosgrove:**

Is that fake or is it real?

**PJ Hagerty:**

That's real. I didn't make that up.

**Kat Cosgrove:**

Okay.

**PJ Hagerty:**

There's a DOS JS, which would allow you to put an Easter egg on your website so that if you click something DOS would appear and the toaster fly toaster way. There's all kinds of ridiculous things that happen, 90% of them are abandoned of projects and get up.

**Kat Cosgrove:**

That is bizarre. Personally I hard committed to Vue.js because I didn't like Angular at all and didn't super love React. So I just hard committed to Vue.js. I don't care if it's outdated, I'm not a web developer anymore.

**PJ Hagerty:**

Yeah. Well, when the question comes up, the people say... I know I'm saying this on the matter most podcasts. I know we all love Go, but I'm going to say it anyway. People say what language you pick took out on Ruby on Rails every single time.

**Kat Cosgrove:**

I thought every single time [crosstalk 00:24:05].

**PJ Hagerty:**

Not even the Rails, Ruby. I'll just do Ruby.

**Kat Cosgrove:**

It's Ruby.

**PJ Hagerty:**

Yeah. Ruby with Sinatra for little base apps to toss up an example for a talk. Yeah. Ruby is simple. It's made for developers. It makes me happy.

**Kat Cosgrove:**

I don't write Go anymore. Not the biggest fan of it. I loved it at first. But the more I wrote it, the more annoying I found it. Sorry everybody.

**PJ Hagerty:**

All right. So I think we're going to change the title of this episode to Kat and PJ piss off everybody in the Mattermost community, which is fine.

**Kat Cosgrove:**

Which is fine. That's what I'm here for.

**PJ Hagerty:**

Yeah. That's why I make these podcasts. But seriously, Kat, I want to thank you for being on the show. If people want to see you, where are you going to be? Where can they find you online? What are you up to?

**Kat Cosgrove:**

You can find me on Twitter @Dixie3Flatline where you will get a mixed bag of serious tech tweets, shit posts and pictures of my cats. You can also find me on the conference circuit throughout the entire month of May, which is going to be brutal but I will be at J focus, DevOpsDays Birmingham and KubeCon EU. So you can come find me at any of those and argue with me in public about any of my offensive hot takes over the course of this podcast.

**PJ Hagerty:**

Yeah. Don't don't do that.

**Kat Cosgrove:**

I'll give you a pair of cat ears if you try, but I will eat you alive.

**PJ Hagerty:**

Okay. So if you find Kat at Cuban EU, you can get cat ears, but you have to argue publicly. I need to see video of this. I don't know who else is going to be there from Pulumi team but please have an assigned videographer for-

**Kat Cosgrove:**

I'll get David on it. Yeah.

**PJ Hagerty:**

Perfect. Perfect. Just arguments with Kat. Okay. So here we go. I'll just watch that Twitter feed. Make it a Twitch stream. It'll be great. But awesome, Kat, thank you so much for being here. Hopefully we can catch up with you again maybe later this year to see what's been going on, see what's going in the Kubernetes community. For those listening, we look forward to

bringing you many future episodes of this podcast. Keep listening and feel free to get in touch at community@mattermost.com with your questions, comments or episodes and guess ideas. You can always reach out to me, PJ.Hagerty. I am on the Mattermost community server, which you can find @community.mattermost.com. Let us know what you think matters most.

**Voiceover:**

You've been listening to the What Matters podcast, hosted by PJ Hagerty @aspleenic on Twitter. Music is Upbeat Party by Scott Holmes. For more information, contact community@mattermost.com. Let us know what matters to you and we'll talk next time on the What Matters podcast.