EPISODE 1184

[INTRODUCTION]

[00:00:00] JM: Cost management is growing in importance for companies that want to manage their significant cloud bills. Kubernetes plays an increasing role in modern infrastructure. So managing the cost of Kubernetes clusters becomes important as well. Kubecost is a company focused on giving visibility into Kubernetes resources and reducing spend. Webb Brown is a founder of Kubecost and joins the show to talk about Kubernetes cost optimization and what he's building with Kubecost. Kubecost could be an interesting infrastructure solution to people who are spending too much on Kubernetes.

[INTERVIEW]

[00:00:41] JM: Webb, welcome to the show.

[00:00:42] WB: Thanks for having me. I'm really excited to be here.

[00:00:46] JM: Okay. Many people are managing Kubernetes clusters. What are some difficulties in managing a Kubernetes cluster?

[00:00:52] WB: Well, there's, yeah, a lot of complexity that comes especially with scale. One of the areas that we really focus on is helping teams get cost visibility and specifically cost management. But we think there's still a fair amount of complexity to be managed, whether it's performance management, reliability management, access control. We still think that there's a good bit of complexity in the kind of day two part of the journey that a lot of teams are really focused on solving as more and more teams get to real production scale with Kubernetes.

[00:01:31] JM: When you do go to production and you're managing a Kubernetes cluster, how much manual work is needed to scale up and scale down that cluster?

[00:01:39] WB: Yeah, it's a good question, and it really varies I think about by the team and their kind of maturity level on Kubernetes. We do see more and more teams using tools like cluster auto scaling or pod autoscaler. But even with those there's often times some like manual oversight or manual configuring those actual tools. So there's still a good bit of manual work, whether it be, again, kind of access control, security related, or, again, actual workload or infrastructure scaling. I think we're seeing a lot of, again, new tools and improvement in tools to take out some of that manual element. But today, there's still a good bit of manual work to really scale infrastructure to massive, massive clusters with hundreds or even thousands of nodes.

[00:02:32] JM: You work on Kubecost. What is that?

[00:02:34] WB: Yeah. So Kubecost is a platform for giving teams that run Kubernetes cost visibility and cost management solutions. So we're built on open source. We come in and let teams see cost allocation by, say, namespace, label, cluster, even all the way down to the individual pod or container level. And then with that visibility we give them insights and a small bit of optional automation to actually help them manage that on an ongoing basis. That's included with additional governance oversights, things like budget alerts within Kubernetes clusters, like efficiency, threshold alerts, that sort of thing. So really, again, a lot of our visibility is within Kubernetes, but we're really focused on teams that run Kubernetes and all of the related services they're using that may be outside of the cluster as well.

[00:03:31] JM: There are a lot of different cost monitoring platforms out there. Could you explain what you focused on?

[00:03:38] WB: Yeah. It's a great question.I think it speaks to a little bit of like our background. My co-founder and I before starting the company, were at Google working on infrastructure monitoring given the nature of Google's infrastructure. A lot of that was focused on containerized workloads. And what we saw was that as teams transition to containerization and specifically orchestration platforms like Kubernetes, there's a lot of new complexity that's introduced, technical complexity. There's also a lot more likelihood that infrastructure is going

to be really dynamic. So like we were talking about infrastructure and workloads are going to be dynamically scaling across region, across nodes, maybe even across accounts or across providers. And then third is that technical or engineering decisions are getting made differently is what we commonly see and oftentimes they're getting made in a more decentralized way where each individual team may be controlling more and more in their kind of deployment configuration decisions. So as a result of this, we just feel like fundamentally there's a platform shift when teams move to Kubernetes, and it just necessitated new tools specifically for cost visibility and cost management.

[00:04:54] JM: And how does Kubecost compare to standard cloud cost monitoring products?

[00:04:59] WB: Yeah. I think there're kind of three main differences that we see. The first is that we are truly Kubernetes first. So we truly build all of our products with teams that have Kubernetes at the heart of their infrastructure. And when we first launched the product what we were seeing is that most teams that were in this position were kind of building something their own, whether it be kind of estimates on Grafana dashboards or kind of building their own data pipelines. And then, secondly, our product is based entirely on open source. So we have our own open source repos. We also integrate really tightly with other open source projects in the Kubernetes ecosystem like Prometheus and Grafana. And then third is that with our product, by default, and it's something that's really important to us is that users get to own and control all of their own data. So they don't have to egress any information to us any of our remote servers or anything like that. They truly have total control over their data.

[00:06:07] JM: So what should I expect from Kubecost after I install it into my Kubernetes cluster?

[00:06:12] WB: Yeah. Great question. I think, first and foremost, most teams can install it in less than five minutes. It's truly just like a Helm install or deploy a YAML file to your cluster. From there, we take read-only privileges by default. So you can see in our product, after deployment, you can see breakdown across really any meaningful concept that you want to see. So you can see cost broken down by namespace, by team, product, environment, project,

etc., where all those concepts kind of map to your internal organization, whether that'd be by Kubernetes label, annotation, etc. You can see that data by microservice. You can see it by controller. And, again, you can even see individual pod or workload.

So tons of different visibility on the allocation side. You can also see just an aggregate view of where you are spending resources. So things like you're spending money on nodes and load balancers and disk and even out of cluster assets like S3 buckets or Cloud SQL or RDS instances. So we would give you that kind of full visibility and, again, kind of break down in cost. So from there, we would also give you insights into how you can reduce spin and specifically reduce spin while balancing performance and reliability constraints. So we would give you insights to say you're at risk of being CPU throttled or hitting an out of memory eviction while also balancing that with helping you, again, reduce the amount of resources that are provisioned or finding the lowest cost class of resources available to you. So it's really both of those things kind of available out of the box in that less than five minute install visibility as well as those insights around managing or optimizing spin.

[00:08:16] JM: How does Kubecost compare to monitoring products like Grafana and Prometheus?

[00:08:22] WB: Yeah. So our product is actually built on top of Prometheus and tightly integrated. And so what we do is we generate a bunch of new metrics available for Prometheus. Specifically, we would integrate with your billing APIs across all the major cloud providers and we would provide that data available directly in Prometheus for teams to query against. That in turn can also be viewed directly in Grafana.

So we work really tightly with those open source technologies. And, again, expose kind of new data, new metrics that are available for new visualizations. And a lot of those are, again, around cost visibility or cost allocation as well as cost optimization or cost management.

[00:09:12] JM: What kinds of cost optimizations do people make? Like what kinds of savings do people get out of using a cost optimization tool?

[00:09:20] WB: You can think about this as like being relevant to different parts of the stack. So if you look at your actual infrastructure, you can say what like class of nodes are you using or what size of nodes or VMs are you using? You can also say you're using on demand versus spot versus, say, savings plans or reserved instances. So a number of insights that are available at the infrastructure level. And then on top of that if you go kind of one layer up, if you look at the orchestration layer, there's a number of insights available. So how are you using things like cluster auto scaling? How are they configured? Are there configuration options that are preventing that from working efficiently? Are you replicated across multiple zones in your actual cluster deployment? So a number of things there.

And then a third is actually at the workload level or like resource, like workload resource provisioning or configuration. So things like tuning the size of pods, so requests and limits. If you have like abandoned or orphaned resources. So things that have been provisioned but are not actually in use anymore. So really we see insights at all three levels. And depending on your situation, we've seen each one of those layers have a 30% plus percentage spin reduction impact. But it really just kind of depends on kind of how you've managed your infrastructure to date. And oftentimes the complexity in your organization as well as kind of applications that you're building on top of Kubernetes.

[00:11:07] JM: So say in more detail, what are the different areas of my infrastructure that I want cost optimization on related to Kubernetes?

[00:11:17] WB: Yeah. So if you look just at the infrastructure layer, I think you can break this down into three different components. The first being the amount of resources that are provisioned. So here, one of the most common exercises is like a right sizing exercise. So you could say given that I have these workloads that demand this amount of CPU, this amount of GPU, this amount of memory, etc., we can run essentially like a bin packing algorithm in our product that helps you determine the right size of nodes or each one of those resources that can provision. And that we help you do in the context of your specific environment. So if it's a dev cluster versus a prod cluster, you'd want to provision different resources. So that's kind of

part one, the amount of resources. So part two of that would be the amount of time that those resources are provisioned. This is where things like cluster auto scaling come into play. This is where things like dynamically turning down, say, staging or testing environments or scaling your infrastructure in an intelligent way based on some either reactive or predictive like demand model. So that's kind of part two, which is just around timing. And then part three is really focused on the cost of those individual resources.

Again, if you know that you need 10 CPUs, you may have different options in terms of usage type. So spot versus on demand for, etc. You may have opportunities to make an investment in, say, a reserved instance or a savings plan. You may also have options in terms of which region or even provider you run that in. So it's really around those three different areas where there's a number of key decisions that can have a really big impact on infrastructure cost and reliability as well.

[00:13:20] JM: As you mentioned, you worked at Google. Can you say more about what Google did for cost optimization or just server optimization?

[00:13:28] WB: Yeah. Google is such a massive like organization. There was a lot done. You can look at it in in several different layers. One is a lot of individual engineering teams did a lot of like optimization for their application using either metrics that they generate or would be available from like centralized monitoring teams. And then there's also centralized teams that are looking at saying things like health performance and cost monitoring. And, again, I think you can also further split that between internal like engineering teams and then, say, engineering teams focus on external things like Google, Google Cloud products. So there's a ton of investment from applying like new machine learning models and also just kind of more manually going in there and saying removing orphan workloads, for example. So I think you can say that there's a bunch of teams really focus on it and there's also engineers throughout the organization that think about it on kind of a recurring basis.

[00:14:36] JM: Now, of course, within Google, they're not really looking at it as a cost optimization as much as like a financial cost optimization as much as a resource optimization

problem. If you look at optimization, cloud cost optimization, from a financial perspective versus just a raw resource utilization perspective, how do things change?

[00:14:58] WB: I think you're exactly right. Within Google, it was more common to look at this from like a resource quota or resource or just like overall capacity constraints and less as a focus on kind of ROI. I think that when you look at it from a dollar standpoint, you introduce a lot of new like strategic and business questions. And that's kind of like we're spending X-dollars on this micro service. What is it actually doing for our company or for our overall software application that we're offering to say internal or external users?

And I think that brings in new decisions, again, around, say, shifting from on demand, to spot, etc., that could still meet the same capacity constraints but would be like a more financial decision. And then I think it also kind of introduces a lot of these strategic decisions where you see teams like fin-ups getting involved in helping make some of these kind of engineering but also business strategy decisions.

[00:16:13] JM: Tell me about the engineering behind Kubecost.

[00:16:16] WB: So the interesting thing here is that the Kubecost open source project was started by my co-founder, Ajay Tripathi, who was an infrastructure engineer at Google and then Yelp before this. It was started before we actually started this as like an official company or startup. And so we truly started the project in mind first and foremost to just be helpful to teams and to build the like experience that we wanted. And by part of that, we were again big believers in Prometheus and Grafana as initial solutions.

Sso we built the back end in Go first. And what that did was, again, it essentially created a Prometheus metric exporter so that we generated a bunch of these like new metrics for Prometheus to use and initially we just exposed them on Grafana dashboards. And then we came behind and built our like UI, which is predominantly React today. But you can find that original like Github repo still out there under the like Kubecost cost organization name. And it is

our go back in that is used today for not only our open source, but also for our like commercial tiers of our product as well.

[00:17:38] JM: What's the biggest engineering problem you had to solve in building Kubecost?

[00:17:44] WB: There are tons of technical complexity here. When you think about how dynamic Kubernetes deployments or clusters typically are, whether it's jobs being constantly introduced, pods coming up and going down, nodes coming up and going down, etc. Overall, just managing that complexity and managing at scale is where we've spent the majority of our time over the past year and a half. And so we now have teams with thousands and thousands of nodes running our product and we do that at all major cloud providers across all different like asset types.

So, again, thinking that you're running a very specific type of GPU on spot and, say, any region in the world on AWS. We're going to handle that. And we're going to handle it if that like GPU is only up and running for, say, 10 minutes and then it goes away. So being able to do that, manage that complexity at scale and give users real-time data, was like a really challenging problem to solve. And, again, it's what we've mostly been focused on the past year and a half.

[00:19:00] JM: Can you share more about how people have used Kubecost particularly in ways that might surprise you?

[00:19:08] WB: Yeah. We've seen a ton of like really interesting use cases. I think, first and foremost, is around the like real-time nature of this data. Because we are writing this data back to Prometheus, a lot of teams that are already using things like alert manager did a bunch of really interesting things here. So one example would be – One that I've seen a couple times recently is to say if all of my workloads in this, say, name space across this like spin threshold and there's this amount of waste or specifically we're below this efficiency threshold, let's alert the owner, which we know, based on these labels that are set, we know that the owner hangs out in this slack workspace.

So, really, the ability to just be super like targeted with when and where that alert is delivered around cost visibility to us is a super interesting use case and is just really exciting and very much like fits with our mission of just empowering developers to have access to this information just so there's more and more visibility. So it's not finance coming at the end of the month saying, “Why in the world are we spending this amount of money?” The engineering team just has some level of understanding and awareness of just absolute spend as well as kind of overall spend efficiency. So I think that use case and just kind of the class of use cases around using data in real time to like proactively detect what looks to be an issue and addressing it then and there as opposed to waiting until the end of the month or the end of the quarter and seeing it be a much bigger problem.

[00:21:09] JM: Do you see Kubecost as competitive with the larger cost optimization platforms or do you see it as just kind of like a smaller niche cost optimization business?

[00:21:23] WB: We very much want to be the absolute best for teams running Kubernetes. We want to give the best you know insights , the best like user experience. Like that is our focus today. We have a number of teams that run our product side by side with kind of more legacy cost monitoring solutions. So we definitely see an opportunity for us to kind of be even collaborative or partners there, because there's a lot of spend and things outside of kind of the immediate cloud infrastructure touched by Kubernetes. So we do not view ourselves as kind of competitive with those solutions today by any means.

[00:22:10] JM: In thinking about the cost optimization market, I sometimes compare it to like the monitoring market where you have all these different players and then you have some really big players like a Datadog who are just taking a much larger market share, but it doesn't mean that the other ones, other companies aren't successful. Do you think the cost monitoring platform – Because there's no network effects really. Do you think it's kind of a similar domain?

[00:22:38] WB: Yeah. I don't think this is like a winner take all market by any means. I think it's a massive, massive market I think that when you look closely a lot of teams have different needs, right? So there's not a ton of like homogeneous needs when you look at like broad

groups. And part of that is still I think how infrastructure teams or SRE or DevOps teams are organized. Yes, I think there are lots of similarities for sure.

[00:23:13] JM: Are there any application domains that are particularly hard to optimize like machine learning workloads?

[00:23:20] WB: Here, the specifics of the situation in my opinion oftentimes trump kind of high-level class of application. I will say that things that are like highly time sensitive can be a little more challenging to optimize. Things that are a little bit more like batch oriented and less time sensitive generally have like even more opportunities for optimizations. But I would say, in general, there's no real class of like workloads that I'm familiar with that don't typically have opportunities for gains, whether that be, again, optimizing for cost performance or reliability.

[00:24:13] JM: Can you just give me more information about the runtime of Kubecost? Like once I've installed it, what is it doing on across my infrastructure as time goes on?

[00:24:24] WB: Yeah, great question. So what we do is we're kind of doing two things. We're talking to the Kubernetes API and we're also talking to your billing APIs to ingest new data or create new metrics. So that would do things like a new node join your fleet or a new deployment was created with five replicas. We would be determining the cost of all of those new assets. And that same thing would apply with, again, a staple set was like scaled up or scaled down, for example.

So we're generating those new metrics. And then, in parallel, we're actually reading both the metrics we just created as well as other metrics to build this model of the state of the world across all of the clusters that we're monitoring. And so we build this model in memory and then we actually write it to disk if you deploy our product with a persistent volume or durable storage backing Prometheus in the form of like Thanos or Cortex or something else.

And what that model allows us to do is we can really quickly say what is the cost of, say, like this namespace of the last 180 days? Or what is the cost of this set of labels over the last 45

days? And we're able to do that without ever having to query the backing database, which is oftentimes from Prometheus. So we can just like, say, Google Analytics or other observability tools, we can answer questions that are really hard to answer by just querying a database directly and we can answer those oftentimes in milliseconds or hundreds of milliseconds.

[00:26:13] JM: Is it tough to make a monitoring service run like this without degrading the overall latency of the system?

[00:26:23] WB: No. I think we've like just given the kind of different isolation features available in Kubernetes. And given that we've purpose built this application for our exact use case and our commercial tool, we've been able to not only like reduce load on an individual machine, but also do things like effectively throttle or manage your load on, say, your underlying time series database like Prometheus.

So I think we've made some big investments there. But as a result, it's really hasn't been a problem. I think noisy neighbor problems in general on Kubernetes are still kind of an open problem space. I think there's been like great inroads here and there are some good like guard rails available. I expect we'll see these be hardened over time both like in the Kubernetes framework itself as well as tools both observability and probably more automation help teams manage this on top of Kubernetes as well.

[00:27:39] JM: Another question revolving around your time at Google, do the cost optimization tools that are required for a company as big as Google, how do those compare to the cost optimization tools that might be useful for a significantly smaller company?

[00:27:57] WB: Yeah. Yeah. Great question. We kind of think of this in like three tiers, right? If you're a really small company, you're probably okay with just kind of having estimates or maybe just like winging it. And that could be you're less than 10 engineers or less than 20 engineers. And then on the really big end of the scale, the Googles of the world, the like Netflix of the world, they oftentimes have proprietary systems in-house where there can be real value

to either build something from the ground up or just use, say, our APIs to where they deeply integrate that into their environment.

So oftentimes there's custom systems that they both get a lot of value from and also have the resources to maintain and integrate existing solutions with. So I think there's a little bit more bias towards, again, having an API only solution or just custom building something. But then there's a ton of space in the middle where kind of an off-the-shelf solution like a Kubecost can be super valuable. Because for most teams, like trying to recreate the wheel here isn't like part of their core value proposition. And instead we can just get them great visibility right out of the box that they can, again, have insights that they didn't have before and actually manage spin much more effectively.

[00:29:32] JM: Give any insights on how costs of running a Kubernetes cluster on Google Cloud compares to running one on Amazon or on Azure.

[00:29:45] WB: I would say we have the underlying APIs available to like have insights there. I think that there's enough complexity and enough like that is dynamic that we haven't come out and like had any official studies or analysis on it. I would just say that when you're kind of making the decision across provider, there's a lot of moving parts that like can influence that, right? And that can be in the cluster and out of cluster costs as well as just features, whether they'd be kind of like cloud services directly or more meta features like identity management. Oftentimes those can drive the decision even more so than cost.

We actually think that's – And this is kind of off the record maybe, but like I think it's really interesting. We have not done any like really in-depth analysis to say that cloud provider A is less expensive than cloud provider B. And our perspective is – And, again, just kind of off the record, is that this is like a nuanced enough decision making problem that like you want to go really deep when you do this so that you can like give insights that are actually meaningful.

[00:31:11] JM: While we're on the subject of cloud providers, why is anybody even running Kubernetes clusters? Why not just use these standalone container instances or like Cloud Fun or Fargate? Why even run a cluster?

[00:31:27] WB: Yeah, it's a great question, and you've seen like insanely fast adoption of Kubernetes. And I think you have to say, like at its core, it's because there's like real valuable propositions for teams. I think you can break this down into like several different decisions, which is if you're coming from like a VM world, why make the move to containerization and microservices, et cetera? And then you can also make the decision, "Okay, if you're all in on containerization, what is the right platform for delivering that?" Whether it's something more serverless like a K-native or Cloud Run or kind of Kubernetes itself. And even there, there's more decisions of are you going with a more managed like GKE or EKS type solution or you want to roll your own Kubernetes with something like kops.

I think at a very high level, and I definitely think you know situation specifics matter here, but I think at a high level this is a tradeoff between like velocity as well as control. So when we work with medium or large size enterprises, a lot of times the control they need for running all of their workloads, whether it'd be storage requirements, security requirements, networking requirements, etc., can oftentimes make it really hard for a totally serverless solution today to meet all of their needs. They may be running certain workloads on like a Cloud Run, for example. But from a control standpoint, oftentimes we see compelling reasons for them to have more of the like flexibility and control available with running workloads directly on Kubernetes. And I'll just maybe add to that. We do expect to see more and more teams running workloads on serverless solutions that are maybe built directly on top of Kubernetes. But we definitely expect Kubernetes to be at the core of a lot of team strategies going forward just because of the great benefits around, again, velocity in terms of time to release and ability to scale and just the control that you have available for configuring or optimizing infrastructure and the workloads that are running on top of it.

[00:34:02] JM: Do you often see people at an organization who are dedicated to lowering costs of clusters or is that more of the job of just individual service owners?

[00:34:14] WB: I think it's a really interesting question. I think that this is, one, we see a lot of different implementations of, call it, DevOps or FinOps, whatever the term you want to use for kind of the team or role that would kind of oftentimes manage this responsibility. And we also think that this is very much changing or evolving.

I think the most common pattern that we still see is a centralized infrastructure engineering team that has real responsibility for certain aspects of managing infrastructure, but that they have also given a lot of capabilities to individual application engineering teams. And that could contain either spinning up new clusters or just, say, like managing deployments or replica sets, etc., directly themselves.

So I think in practice it's still this like hybrid mode for most teams where there's like a centralized team that has some platform ownership, but they intentionally decentralize a lot of the kind of workload specific or team specific decision making process just so that individual engineering teams can run faster and kind of make more of those application specific decisions themselves.

[00:35:51] JM: Can you share some helpful boilerplate tips how people can make their Kubernetes infrastructure more cost effective?

[00:36:02] WB: Yeah, absolutely. So I think, again, kind of breaking this down by each individual component. If you just look at the amount of resources that are provisioned, I think this is oftentimes like container right sizing or deployment right sizing. There's also just infrastructure right sizing to make sure, again, you're running the like right size nodes given the set of workloads you're looking to deliver and the kind of SLAs or time constraints that you're looking to deliver them with.

I think there's also the second class around the amount of time or the actual scheduling of resources that are provisioned. So, here, looking at things like cluster auto scaling or dynamically turning down entire environments or subsets of environments based on some sort

of business logic that's relevant to the team. And, again, a small example would be we see a lot of teams with death clusters that are a meaningful part of their spin. Just turning those down, say, after midnight and before 8 a.m. or something.

And then a third class of decisions which is just looking at kind of financial opportunities to lower the cost of each individual resource. Common ones there are looking at the opportunity to run spot on nodes depending on how your applications are architected or how you've like structured the actual microservices that are deployed on Kubernetes. That could be really attractive. And then also a second class would be something like reserved instances or committed use discounts or savings plans, which essentially can let you make longer term commitments for a discount on those individual resources that you expect to be provisioning now and going forward. So I think around those three classes, there's oftentimes big opportunities for teams to reduce spend. When we work with teams, most are able to achieve 30 plus percent savings by some combination of those particular optimizations.

[00:38:33] JM: You've written some about how auto scalers can be used to – Cluster auto scalers can be used to improve the economics of a Kubernetes distribution or Kubernetes deployment. Could you say more about that?

[00:38:48] WB: Yeah, absolutely. This is really giving teams the ability to dynamically scale the infrastructure resources that are provisioned based on the actual like usage demand from their end users. So these are tools that allow teams to really scale down their infrastructure effectively when their kind of applications aren't experiencing peak load. And there's a lot of – Or a fair amount of configuration that can be needed to get that right, and that could be things like setting pod disruption budgets or making sure that certain workloads are annotated as safe to evict when they're using things like local storage, etc. But the net result of that is that, yeah, you can have massive savings if you do have variability in kind of resource demand over the course of a day, a week, or month.

So that's kind of at the cluster level. You can also do that for individual workloads where appropriate and resize workloads either from a request standpoint. So they're directly reserving

more or less resources. Or you can also scale them from a replication standpoint where you have just more and more instances of the same application. Both there can be great depending on the nuances of the workload, but can be great at reducing the actual resource consumption over time, again, when you have variable demand for a particular microservice or a particular application.

[00:40:40] JM: Do you have any beliefs around how these kind of cost optimization should be deployed? Like does it get deployed as a side car or is it like a standalone server or an agent? Or can you tell me more about the deployment model?

[00:40:57] WB: Yeah. So Kubecost is deployed as – You can think of it like a single agent. It like optionally comes with you know UI and some other things available. But that agent model, we believe, gives you kind of maximum flexibility in terms of the ability to like collect data from a number of different sources. But then also have like a standalone solution that kind of we as developers of that are responsible for.

Whereas if you're kind of integrating as like a sidecar to an existing solution, it can be a little bit harder to kind of like manage that overall experience. So, yeah, we think that agent model works really well here. And, again, our agent today is very tightly integrated with, by default, a Prometheus deployment, which again we think provides great benefits by allowing us to generate new metrics and have those available to all of the tool set that is built on top of Prometheus as a project.

[00:42:11] JM: Well, we've explored Kubecost in some detail. Do you have anything else to add? Any other areas you want to explore?

[00:42:20] WB: I think it would be like maybe interesting. Let me know if you think this would be like relevant. But like one kind of maybe why we started the project or something, because, again, we truly started the project first and then the like company or the startup behind it afterwards. And then I don't know if you think there'd be interest in the like – We have an open core model. So we have an open source repo that provides all this visibility, but then we have

like commercial products built on top, which are aimed more for kind of larger enterprises that need things like SAML and RBAC and all that sort of stuff. So I think the net result of that is we've got a free tier that works for like even medium-sized companies, but then there are other like enterprise solutions available built on top of those. So I'm not sure if either one of those would be interesting to the –

[00:43:24] JM: Let's go into both.

[00:43:25] WB: Okay. Awesome. So maybe you may hit on just kind of like why, why we started the company first. So I think like there's two parts of kind of why we started the project of Kubecost and then ultimately the company behind Kubecost, which is StackWatch. And I think, first and foremost, it is like the most rewarding things that I've ever worked on are when I worked with a super engaged motivated team to build something from the ground up. And we like definitely saw the opportunity to do that here, and that is like deeply interesting for me, the ability to have a vision and just go and create.

And then part two of this is we personally are just deeply attracted to the problem space here. And at its root we see an opportunity to really empower developers and get access. Give them access to like real-time data when most teams have none here. And one thing that we think that helps avoid, again, is this problem of getting a call from finance or maybe management after a month, a quarter, etc., and saying, “Why did we spend all this you know money?” And having to go and dig and find the answer. We wanted to, again, really empower engineers to at least just have that information at their disposal, right? So they aren't kind of getting surprised by that phone call and just have some level of transparency or awareness as they're kind of making different infrastructure decisions, but also just application level decisions. So I think it's really that opportunity to go and build something from scratch and then just really motivated by the underlying problems that we've seen teams facing in this like transition to Kubernetes and cloud native.

And then maybe part two, just talk a little bit about given that we created the open source Kubecost cost project first. And, again, we want to empower engineering teams or

infrastructure teams of all shapes and sizes. We've made it a real focus to have a comprehensive like valuable free product or community product and that will always be free. And then where the commercial side of this is layering on enterprise features that are valuable to companies of really large sizes. So things like SAML and RBAC where you have many engineering teams and you have like multiple layers of management that gets increasingly valuable. So that's kind of the strategic approach we've taken with the company. And we think that allows us to hopefully have both a sustainable business, but also, again, staying very much true to our roots, which is we want this data to be accessible to teams of all sizes all the way down to a team of one or two engineers that wants this visibility. And we think that should be free and always free for smaller teams.

[00:47:07] JM: Well, that's great. Are there any other subjects you want to explore?

[00:47:11] WB: No. I think we hit on a ton that's super relevant. I mean I'm just super impressed with the research you've done here and the like depth of questions. I'm not sure if there's anything you'd want to like circle back to where you felt like maybe I could provide some more data or anything like that. But, overall, I think is, yeah, just seriously impressed with how deep you go in it. It makes sense after like listening to other podcasts of yours, but it's – I don't know. It's especially like impressive when it's the like nuances of our own space. So thank you for taking the time to do that. And, again, it shows in the other podcast of like just your ability to go deep on many subjects.

[00:48:03] JM: Well, thanks for the compliment, appreciate that.

[00:48:06] WB: Of course.

[00:48:08] JM: Cool. Well, Webb, thanks for coming on the show. I guess we can wrap up now.

[00:48:12] WB: That sounds great.

[END]