EPISODE 1213

[INTRODUCTION]

[00:00:01] JM: Modern SaaS products are increasingly delivered via the cloud rather than as downloadable executable programs. However, many potential users of these SaaS products may need that software deployed on-prem on a private network. Organizations have a wide variety of reasons for preferring on-prem software such as security integration with private tools and compliance with regulations. The cost of setting up a bespoke on-prem version of a SaaS offering was often prohibitive for both the vendor and potential users. Replicated leverages the portability of containers to help SaaS vendors ship an on-prem or multi-prim version of their software. Replicated gives SaaS vendors a suite of ready-made components to help install and manage an instance of their software on-prem. Replicated has seen a lot of growth in its six-year lifespan. Today, 50 of the fortune 100 companies manage apps with Replicated.

Grant Miller is the founder and CEO of Replicated. Replicated recently closed a series B fundraising round to help scale their work in the cloud native space including the launch of their Kubernetes off-the-shelf platform. Grant returns to the show today to talk about the next generation of tools on the Replicated platform, how Kubernetes is changing enterprise IT and why the on-prem software market is not going away anytime soon.

[INTERVIEW]

[00:01:23] GM: Grant, welcome back to the show.

[00:01:24] GM: Thanks so much for having me, Jeff. I really appreciate it.

[00:01:27] JM: We've talked before a bit about Replicated and multi-prem environments, but let's just revisit the problem statement that you're tackling, which is packaging and deploying applications to multiple environments to the cloud or to on-prem. Why is that difficult?

[00:01:49] GM: Yeah, that's a good question. Ultimately what Replicated is really focused on is sort of an even more nuanced version of that, which is funny. We're focused on enabling software vendors. These are ISVs. So folks that have been packaging applications as JARs and OVAs for the last 5, 10, 20 years, but also modern SaaS companies to take their applications, package them up and deliver them into a whole variety of different end customer environments. So these are complex enterprise environments where an IT organization at some big bank will run one of these applications privately.

Oftentimes this has been called on-prem software, but we think about it today more like multi-prem because it's not just data centers and server closets where these applications are running, but instead many of these applications are being installed into an enterprise who's an end customer into their own cloud environment that might be in AWS but it's a VPC. So it's a virtual private cloud that's secured within their own AWS account but it's out of the control of the vendor.

And so this idea of sort of third-party software is just complex and it's more complex than first-party software development. Most of the sort of software development lifecycle, the SDLS, this sort of way that your audience will think about writing, building, testing, deploying, monitoring and managing code, most of the tooling around that is focused on first-party application development and delivery. And so where Replicated really takes a unique and more complicated angle is saying, “Well, this is for third-party application delivery and management.” And doing that with lots of different vendors lots of different enterprises gives you a very complex matrices of applications that need to run in different environments with different requirements. And supporting that – And the challenging part is as an ISV or as a SaaS company that's approaching doing an on-prem distribution, it's a pretty daunting challenge, right? For most ISVs, what's happening today is that they're moving away from these more traditional packaging methods of delivering a binary in 127 page install guide and now they're delivering more robust. fully orchestrated Kubernetes applications. And so they're sort of starting at square one in terms of how do I deliver a Kubernetes application, which is in

itself a complex application into a variety of different environments. And so that's really the problem that Replicated is solving. And we've built a bunch of open source tools around that.

Since we last spoke, I think, Mark, my co-founder and I were on your podcast about a year ago and lots happened in that time, right? The company has – We've raised 25 million dollars as part of our series B. We've added incredible customers like Puppet and UiPath and Tripwire and many others that aren't public yet. And these companies are all facing the same challenge, which is they have a modern Kubernetes application that they need to deliver into a customer environment and they know that they can build their own tooling to solve this, but they can't build everything. And so they come to Replicated looking for the best practices and sort of the tools to help them get there faster.

[00:05:39] JM: As the number of customers that you work with has increased, the number of different software vendors that need to have an on-prem option of their software, what have you had to change about the Replicated software suite? What have you had to change? What else have you had to build out to allow these companies to have an on-prem deployment option?

[00:06:02] GM: Yeah. We've been around for six years. And so throughout that time this ecosystem has really evolved. And you've talked a lot about the container wars in the very beginning, but ultimately what's happened in the last three years is Kubernetes has become the de facto solution for how most modern organizations are going to develop and deliver a cloud native version of their application. And so you know we've had to really embrace Kubernetes, which we've done. We've had to really focus on making sure that these installations are easy to run for a whole variety of end customers, right?

If you're a software vendor and you're delivering to 100 different enterprises, you basically have 100 different end customers or operations teams who are trying to deploy your application and they have a whole variety of different experiences with Kubernetes. Some are really Kubernetes savvy and some are much less so. And so Replicated provides sort of the foundation to deploy to customers who are not super Kubernetes friendly. They just want to

deploy you know what we call multi-node software appliance. They don't even really have to know which Kubernetes under the hood. So they can install that on a VM or on a bare metal. They could scale it out to multiple nodes. It sort of makes it easy to do that. Or we can help you deliver into the most advanced customer who has a fully GitOps pipeline for deploying existing Kubernetes applications and they want to add one more application, this new vendors application into that deployment pipeline. Replicated can facilitate the delivery of that.

So there's really been a lot of technology behind the scenes that we've been building to streamline it for the least advanced, but also allow that most advanced user and everyone in between to sort of chart their own path and find the automation that they might need or the integrations they're looking for to make this sort of – This transaction right between a vendor and a customer to make that transaction successful.

[00:08:16] JM: When you have a large ISV that starts using Replicated, you have a lot of demands on Replicated. Like you have to be sure you're regulated properly, you're compliant. The proper controls are in place and there're no data loss or data security problems. So how do you make sure the platform is properly secured? Do you have any testing procedures or have you gotten like third-party security review? How do you ensure that things work as securely as intended?

[00:08:57] GM: Yeah, it's a really great question. Particularly in light of the SolarWinds hack, which was really a hack around the delivery of their software, right? So someone basically got into their software delivery pipeline and then was able to bring malicious code into hundreds or thousands of customer environments. And so we take security very seriously like most enterprise software companies. But yeah, I mean we do third-party audits. We have a constant focus on security within the team. We're always working to make sure that you know our testing suite is robust. We do a bug bounty program. We follow the ISO 27000 security practices. We're constantly looking for ways to sort of evaluate different threats to do threat modeling and just make sure that our pipelines are secure. And a lot of that's just done through as much automation as possible, right? We don't do anything by hand and we don't let any single person do anything. So everything goes through dual control. Everything is very focused

on how do we make sure that the software that we're delivering is authentically ours? The bugs are fixed quickly. And we work with our vendors.

No one's perfect. There are sometimes – It's funny. Some of the security issues that we've had we're more just like design decisions that we didn't expect people to use the software the way they were using it. And when people brought it up to us, this is how they were using it, we needed to change things. So we have addressed some vulnerabilities in this past year. Nothing that I would say is like super severe, but we worked with our customers and we it took, I'd say, like a pragmatic approach to getting patches out and to working with customers to make sure that they didn't have vulnerable versions of our software out in the wild any longer as well.

[00:10:59] JM: The process of updating a third-party application, like if I'm operating – Like let's say I run an enterprise company and I'm purchasing software from an ISV and that ISV is deployed using Replicated into my data center, of course updates need to happen to that ISV application. How has the process of automated updates and other maintenance things, reliability, uptime? How have the processes around that changed over time, the management of the third-party ISV on the enterprises infrastructure?

[00:11:40] GM: So for many years the way that vendors would deliver their applications was a very bespoke and somewhat custom process everybody would sort of create their own way to deliver their software. And it might – Like I think it's funny. When you look back probably seven or eight years ago, everyone was writing their own type of orchestration. Sometimes it was just a more advanced version of Bash scripts, but everyone was kind of orchestrating some of their services together and figuring out different ways to do service discovery. And so we didn't have such a consistent pattern as we have with Kubernetes.

So for an IT admin 10, 15 years ago, the first thing was read through these run books and these instruction manuals that were provided with every piece of on-prem software and figure out what the recommended sort of next steps were based on what things you were seeing. Was it slow? Were you having intermittent connection issues? And then you would try different configuration tweaks to address those things, right? So this was a very manual way to operate

software. And we spent a lot of time as an industry running software this way where sort of the knowledge was sort of within these operations folks who would sit in a knock and sort of monitor all the applications that we're running and have a 24 by 7 follow the sun type model where people are always on staff looking at the logs or looking at metrics and trying to figure out what they needed to do next. And that was the sort of only way you could run these applications.

Updates, to your point, again you would refer to a new manual and you would look at what processes need to be shut down? When do you need to start the next one up? Like you basically were doing orchestration somewhat manually. And depending on your configuration and your implementation, once again, you were focused on like how do I bring this up correctly? And for many organizations, like if you think about running an exchange server or CRM, anyone that's been in the industry for 20 plus years sort of knows that IT often had downtime issues, right? They couldn't keep these internal services up and running. They didn't have that many, but oftentimes your email would be down or some other service would be down that your company provided. And that was a somewhat persistent problem within the ecosystem.

And it sort of mirrors – I kind of refer to this as application-based operations, right? And it mirrors the same issues that many of the web scale companies were facing in terms of running their own applications on the first party side, right? And this is why Google and Twitter and Uber and Facebook and everyone built their own orchestration systems, things like Borg and Peloton and Tupperware and they built these tools in order to – They couldn't throw more humans at this problem, but they needed more automation to actually manage the applications and manage the responses and they wanted developers to be able to write sort of manifest that describe how these applications should run. And I refer to that as platform-based operations, right? So on the first party software development side you had this shift from app-based operations to platform-based operations, which is very largely kind of categorized just in containerization and ultimately in the widespread adoption of Kubernetes. But the same thing is now happening on the third-party application side where as Kubernetes has become

fairly ubiquitous within the enterprise as well as ubiquitous within the ISV and SaaS company sort of architectures, now it's become this canonical way to deliver third-party applications.

And so what we have is a shift from manual operations of third-party applications like we were talking about before to a much more automated operation of third-party applications done through platform-based ops. So instead of managing all these different applications independently, an IT admin can focus on managing a Kubernetes cluster and then the cluster can automatically operate all of the applications that are deployed to it. And so an update process doesn't need to be run this Bash script or this command or shell into this server. Instead it can be as easy as commit this new manifest in diversion control. Create a pull request. Allow someone else to review your request. Merge that in, right? This is a very production grade pipeline. And then once it's merged in it could be automatically deployed either through whatever CICD process that that organization already has set up, or if they're using some type of GitOps pipeline, maybe Flux or Argo could deploy it out.

And so you have – That entire commit pull request workflow can be automated through a product like the Replicated KOTS project. And our belief is that the more automation you can put around the sort of management of third-party applications as private instances, then the more demand there will be for applications to be deployed this way, because ultimately the sort of consumption of third-party applications as private instances is a much more secure and potentially like much better way to run applications. This is you're giving control to the organization who wants to run the application. And so they can decide you know how to actually deploy it. What to expose the Internet? They don't have to have every single application on the public Internet. This could all be behind you know private IPs and deployed and secured with a BeyondCorp model. So there's a really a lot you can do if you deliver software this way. And we're seeing the industry move very strongly in this direction.

[00:18:37] JM: Can you say more about that? The add-on benefits of this model and tie it back to the BeyondCorp model?

[00:18:46] GM: Yeah, sure. So it's funny. When BeyondCorp was first launched, everyone thought that like there's like, “Oh! Google is putting every service that they have on the internet.” And the reality was that Google was putting every service they have behind what's called an access control, right? So like an access proxy. And that serves for coarse grained access control to determine if someone maybe is within an organization should be able to even access the sort of webserver for an application. So everything had a front-facing kind of like can you log into the generic service?

Then within BeyondCorp, it was doing things around device attestation. You can basically start to validate that this device is secure. It has a secure key on it and it's updated in terms of its browser or its software and you can do all these different attestations. But ultimately the services that are running are actually not running on the public internet. Just the proxy is. And then all the services are running in a private network that Google would host or that any company who's modeling a BeyondCorp would run those on private IPs within a VPC. And so what we're talking about is how do you get those applications? Those third-party applications into that VPC and like what applications are they? And ultimately those applications, they can either be open source all internally written by the company that's deploying them or they can be third-party applications that are delivered by vendors as private instances.

And so obviously we believe that the market for delivering these applications as third-party private instances is enormous because the software market's enormous, right? We don't need to all be reinventing the wheel. Instead we can take and we can all build our own part and then sort of buy from each other and move much faster. And what we're seeing is if you think about the broader world, right? So the advantages of running these applications as private instances is now you have taken this data surface area, which would have potentially been delivering your data or copies of your data to hundreds, if not thousands of different vendors. And those vendors have their own security posture, right? And they have different numbers of employees. Maybe they have 200 employees. Those various employees have different access to different amounts of data and they all have their own security posture. And so what happens is when you're using sort of the SaaS-based model for everything you end up with hundreds of thousands of people and thousands of companies that have access to some of your data. And

this is sort of a challenge in a world where the relationship that companies have with the data that they touch is changing.

I would say that maybe like five or ten years ago any organization thought that they owned any amount of data that they touched. But now you look at some big bank and they think about customer data not as data that they own but as data that they are stewards of or that they might be custodians of. And so that custodial relationship is different than one of a, “I own this data.” Because when you're just a steward for data, you don't do whatever you want with it. You actually have to be thoughtful about what does the real owner of that data, that person. What should I be doing with their data on their behalf? And generally it isn't give access to another 250,000 people.

And so this model, what it allows you to do is say, “Well, let's reduce that surface area by reducing the number of vendors we send data to.” Instead let's bring the applications to where our data already resides. And so this is what it is. This is multi-prem. You’d bring applications to where data resides. You run these third-party Kubernetes-based apps in a private instance and now you have a whole suite of tools and applications that your company can run. They can access through kind of that secure authentication and you as an organization don't have to be as concerned about like where has the data gone, right? You know because you control it.

Now you can still set this up wrong, right? You could still expose an S3 bucket without authentication. So an organization takes their own security and their own data security into their hands rather than relying on their vendors to actually do this correctly. And so that's really the top idea here is that it's just a different model of enterprise software for a different age where data privacy matters more, where companies are using more software. Data residency is a big deal. And the reality is you can actually do all of this in a cloud environment, right? If I'm an enterprise, I don't need to have my own data center, right? With Replicated, we recognize that it's okay to have some number of vendors, right? Maybe I'm working with Google and Amazon and Microsoft. That's three vendors, and I'm using their infrastructure as a service. So they're racking and stacking machines for me and giving me programmatic access. But it doesn't mean I need to work with three thousand vendors, right? There's a big difference.

And so when you look at that spectrum and you say, “How can I reduce it down? Maybe I'll work with 20 different hosted vendors and then the rest of these applications I'll deploy into my own environment.” Like that is a much more sort of realistic future for any company that really cares a lot about data privacy and data security. Obviously there's going to be some services and some organizations, right? Like a local flower shop is never going to worry that much or have the sort of sophistication to do this, but their providers might need to be thinking about this, right? Their providers might need to think about how do they reduce their surface area so they're not sending that data on to a third-level of vendors.

[00:24:50] JM: When you look at the AWS outpost model, this on-prem AWS option, do you have any thoughts about – And I guess I should say for people who don't know, AWS Outposts are these things where AWS will give you a server to put in your data center, like a physical on-prem server. When you think about that with all your knowledge you know over the last six years of building Replicated, do you have any thoughts about where the future of on-prem and cloud software is going?

[00:25:25] GM: Yeah. I mean, I think that the Outposts example, and like Microsoft has their version of this, and Google has their Anthos. They're all similar concepts. And basically what it tells me is that we're blurring the lines between cloud and on-prem. And so if you think about the lines as, “Well, I drew a circle around my data center. I do a circle around my cloud environments.” It's like now you can run Outpost, which is all these AWS services in your own data center. Or if you want you can run Kubernetes on top of your cloud environment and run all of your on-prem components in your AWS account or lift and shift your VMs. And so you can move things back and forth. Now you're not going to move applications like on demand back and forth, but any sophisticated organization is going to have a variety of these environments. And for some organizations, they do want to have that data governance and data privacy and regionally have the data in the country that they reside in or that their jurisdiction is in. And so an Outpost might be a really good idea.

Now the difference between something like Outposts and Replicated is Outpost is really about delivering sort of AWS core services into an on-prem data center, right? And Replicated is about enabling these third-party applications, so third-party vendors to deliver their application suite into any environment, be it an AWS Outpost environment, a GKE environment or a VPC in AWS. It doesn't really matter. It could be a Red Hat, OpenShift cluster. It could be a Rancher cluster. It doesn't matter to us at all. We're just trying to give the tools that allow ISVs and SaaS companies to operationalize and scale the delivery of their Kubernetes applications into a variety of customer environments.

[00:27:34] JM: I'd love to get into a little bit more about the engineering of what you've built over the last year since we have last spoken. What are some of the hardest engineering problems you've had to solve?

[00:27:45] GM: So I guess the first part is just acknowledging that so many things have happened over the last year both from a global perspective, right? We went through a global pandemic that we're still hopefully coming out of right now. The world was really turned upside down. And it was difficult. It was a difficult year for Replicated in some ways, right? In the beginning of the year we were doing well. We had the new KOTS products that were starting to really gain some traction, but we didn't know what would happen in the market. And our team, we were 18 people I think in mid-March or April. And since then we've really grown the team pretty significantly. So we've added another 20 people to the team. We were very fortunate over the summer to bring on our chief product officer. And so our chief product officer is a man named Mark Pundsack, and Mark was the head of product at GitLab for about three years and really helped sort of create their strategy around DevOps and sort of the holistic DevOps life cycle within GitLab.

And Mark has been an incredible partner for helping us sort of determine what our roadmap is and should be. He's recently been spending a lot of time with our customers. Making sure that they understand where we're going and what we're focused on. But ultimately Mark's vision is that replicated will continue to focus on making it easier for vendors to deliver these applications into complex environments. And at the foundation of that is just making sure that

it always works. And so we've spent a lot of time building out pretty sophisticated test grids to validate our software and our customers software across multiple different environments and versions and making sure that you can deploy it onto a Rel server or onto a GKE cluster or onto an air-gapped environment. I mean, really just automating so much of that testing. And then pretty soon we'll actually be enabling our customers to sort of hook into all that testing as well for the release of their own software. And so there's a very concerted effort on making sure that this works, because it's hard, right? There's a lot going on here. There're a lot of variables. And so controlling as many of those variables and allowing the testing is a big important part of it.

I'd say that we've also accomplished a lot in terms of, again, more around maturity of this product. When we talked last year, the KOTS project was maybe four-months-old. And so now it's matured to a 1.0 version. Well, well beyond that. We've added layers of authentication and different authentication providers. We've done you know significant work around integrating with some of the other tooling in the ecosystem, things for backups like Valero. We've always had some monitoring support with Prometheus. But really finding as many of these other projects that we can integrate with to create a very, very kind of comprehensive, robust and stable platform for the delivery of these third-party applications. And it's an ongoing effort, right?

I think when you look at what the alternative to Replicated is, some of these teams have 20, 30 people who are building all the tooling to enable them to deliver their applications into customer environments that are focused on this kind of delivery. I think Palantir has built a pretty big system around this. I know GitHub has. There're a lot of different sort of solutions out there. You look at the challenges that on-prem delivery has caused for a company like Atlassian. They just announced they're going to end of life their server products, which are the sort of half of their on-prem offering. And for Atlassian that challenge was they're going to maintain their data center product, which is what their biggest customers use to deliver their products on-prem. But Atlassian had basically two different companies to support these two different products, right? Jira Server versus Jira Cloud. These were completely different code bases, completely different support processes releasing. So this can really provide a massive

amount of cost if you don't you know do it sort of correctly. I mean, correctly is the wrong word. It's like I'm not faulting atlassian. They've been around for 15 years building an amazing company and they've had a massive adoption and Kubernetes wasn't available when they were when they were first delivering their on-prem version. But a lot of these companies now are making this transition to use Kubernetes as the foundation for their software delivery into customer environments. And with that, it's either build all this tooling themselves or find a partner like Replicated to help them do it faster.

[00:33:02] JM: Have there been any customers that you've interacted with that have kind of pushed the boundaries of what you have and maybe proven that you needed to change or re-architect certain aspects of how Replicated functions?

[00:33:21] GM: Yeah. I mean, one of the advantages/disadvantages of Replicated is that our customers are and have always been these sort of cutting edge software companies, right? So when we first started it was Travis CI and CircleCI and NPM using our tooling to deliver their products. And we've continued to work with the sort of most advanced engineering teams out there, right? So Hashicorp uses Replicated to deliver every instance of Terraform enterprise. And these customers understand the problem. They understand the technology. And when they see an issue, they're often right. I like to say, when our customers give us feedback, most of the time our response isn't something like, "No," or, "Maybe later." It's always like, "Why didn't we think of that?" because it's just such good feedback and almost always a very, very sort of useful incorrect idea. And so oftentimes we implement those things and we take a lot of our roadmap from customer requests and figuring out what customers want.

I'd say that there's a really great example of exactly what you're talking about with Circle CI. So I think maybe two or three years ago Rob Zuber, the CTO of CircleCI gave a talk at KubeCon, and he talked about using Kubernetes to deliver enterprise software. And at that time they were a customer of Replicated and they were using our sort of like version one of the product that we refer to as like the Replicated native scheduler, right? We wrote our own scheduling before there was Kubernetes. And Rob talked about how at some point CircleCI was going to

stop using replicated and deliver their entire enterprise product as a Kubernetes application. So he said this in this KubeCon talk.

And so obviously one of your bigger, more high-profile customers who's been with you for several years saying publicly that they're going to get off of your tools and not be a customer anymore would generally be a bit disconcerting. But I still remember my co-founder Mark and I watching the talk together and we agreed. We knew that Rob was right. We knew that the answer wasn't our original product, our version one. We knew that as the world shifted to Kubernetes we needed to have a different solution to enable them to actually succeed not just in the packaging, because Kubernetes becomes the packaging, but succeed in the delivery and the operationalizing of this entire workflows and process of delivering this software to all these different customers.

And so we got on a call with Rob and we explained. It was like, “Hey, we watched your video and we agree. You're not wrong. Like this is absolutely right, but we want to make that transition with you.” And they basically pushed us and said, “Okay. Well, here's the stuff that we would need.” And it took us probably a full year to really have the right solution, and that was the KOTS product that we launched five quarters ago. And so now the team at Circle is moving their customers over to KOTS, and that's the same thing that we've seen with most of our vendors, right? We've had to make a very interesting sort of product transition. It's not that we made a pivot because we were always doing the same thing, right? It was always about how do you deliver modern applications into complex customer environments, this modern on-prem concept. But we needed a technology and a solution that really embraced modern architectures around Kubernetes and then brought value beyond what we were doing before, which was the packaging. We really had to focus where our value was. And so that value became in automation, in workflows, in supportability, in pre-flight checks and all these different things that we offered. And ultimately we've successfully transitioned just about every one of our customers over to this new tooling and it's been you widely loved.

I think one of the things, this is sort of like more of a business comment, but when you look at a company like Replicated and this product market fit concept everyone talks about, and

everyone thinks they have product market fit even before you have it, because I thought I had it. But what I realized after we actually had it, once we launched the KOTS project and it started really to take off, I realized a key indicator for what product market fit is. It's actually around what is your close rate from POC? So from proof of concept to proof of value to someone buying and becoming a customer.

And before Replicated KOTS, I think we were probably converting about 30% of POCs to customers, and that's just not high enough. You can't really run a full go-to-market organization off of 30%. And so after we introduce KOTS, we bump that up to about 75 or 80 percent. And it turns out there's some threshold. I think it's around 50% where you actually have product market fit. So if you're closing 50% of the POCs that you're doing, then you have a really scalable go-to-market motion and you have a product that the market wants. But if you're not closing it at that rate, then really you don't have it yet.

And so for us, I think we found that by working in conjunction with our customers, by believing what we thought the future of software would be and just working pretty relentlessly to get there. It was definitely a tough few years when we knew we didn't have it. So I think we look back at the success we've had throughout 2020. And the company more than doubled in revenue. We more than doubled in head count. We added these incredible customers. A lot of it comes back to the fact that we were able to really partner with some of these incredible customers. Get their feedback, iterate, and they trusted us enough to know that we would build the right thing for them. And and so for that we'll always be grateful, right? It's not just Hashicorp and Circle. It's folks like Forward Networks and Tripwire and Puppet. These companies just know came in and they embraced what we were doing. I think oftentimes they knew that they didn't really want to be responsible for these tools and if a vendor could successfully do it, that it would actually free them up in terms of a lot of resources. So I think that's why they were maybe a little bit more willing to work with us because they knew that it was an important enough problem that if we solved it for them they would get a 10x return on whatever revenue we generate for them. They'd be generating 10x that that in terms of cost savings. Yeah, I think customers are a really, really important part of it.

[00:40:47] JM: What do you have planned right now around open source projects or in the Replicated ecosystem?

[00:40:53] GM: Yeah. So we have a few core open source projects. The ones that I've been talking about primarily is KOTS, and that stands for Kubernetes off-the-shelf software. The other ones that we have all within the same vein of core Replicated open source projects, one's called Troubleshoot, and that houses these two kind of related but sort of separate use cases. One for generating support bundles, which basically collect all the different information logs, data et cetera, from a running instance. Put it through Redaction. That way you don't have any sensitive data leaking. Bundle it up and then actually analyze it both in the customer environment and in the vendor environment to make sure that any self-remediation can be done. So that's a support bundle within the Troubleshoot Project.

And then also within the Troubleshoot Project is pre-flight checks, which do a very similar thing, which is like basically look at the running cluster and then make sure that it conforms to the requirements of the application, right? Does it have enough memory, CPU, I/O? Whatever else you might need in order to make the application run successfully. And most of the time these things are running at update or running at install time and they basically just try to keep people away from the foot gun, right? Like the worst thing to do is to try to install software when you have under provisioned the actual underlying servers. So you want to make sure that you have all the resources you need. You want to make sure that you're running the right versions of things, because this is all easy to check. And by putting these kind of checklists in front that run automatically, you kind of prevent those like dumb human errors that often cause hours of troubleshooting. So as much as we can do to automate those, we try to do that.

And then another project we have also kind of core to Replicated it's called KURL, also with a K, and this is really around – It's like we call it kind of your Kubernetes URL. So it's a Distro creator. Think about it as a way to declaratively manage the add-ons that you would use in a custom version of Kubernetes. So if you want to – Obviously we use it to allow software vendors to embed Kubernetes underneath their application when deploying into a customer that doesn't have an existing cluster. So you go into someone. They don't know much about

Kubernetes. You want to deliver your application. It's Kubernetes-based. How do you kind of bridge that gap? Well, you have them run this installer. It installs onto a single node or on a multi-node. It can do multi-master. It's really pretty comprehensive. It's built on top of Vanilla Kubernetes kind of powered by kubeadm. You can select different sort of plug-ins. You could have Rook and Ceph, or you could have OpenDBS, or you could have Calico, you could have Weave. You sort of pick all the different add-ons/components and their versions and then this tool basically builds the installer. Can either build a tar.gz to deliver to an air-gapped environment along with the install script or you can just run it straight from a KURL.sh URL that we host for you. And that project is once again completely open source. It's a fairly deep project in terms of making sure that Kubernetes is easy to get running anywhere, and we open sourced it because we just want more Kubernetes ubiquity, right? We don't think that getting a Kubernetes cluster up and running with the required network and ingress, et cetera, should be a proprietary technology. We think that should just be like table stakes for anyone to be able to do at any point. So there's a lot going on around that in terms of cluster API and other sort of interesting projects and sigs, but we think that KURL is a really good example of a way to kind of implement that.

And then the sort of fun one/kind of related but not core to Replicated business is a project that my co-founder started called Schema Hero. And we recently donated Schema Hero to the CNCF. And Schema Hero is basically a way to declaratively manage schema migrations for databases. So it's a Kubernetes operator and you sort of write your migrations in what we call a Kubernetes custom resource. So it's sort of like defined in Kubernetes language and sort of DSL and then you apply that and it sort of manages the migration of the schema in the background almost like an ORM would. So you can version control these schema migrations. You don't have to worry about like a schema migration needing to run through 30 different sets of migrations before your server is up and ready. And this actually is a project that some of our customers end up using, because if you know much about schema migrations, most of the time they were sort of imperative and they would sort of run on top of one another. You'd have to run them in order. And then there would be challenges around like some version change then causing an error in the migration run. And so this just felt like a really simple project that

would have a lot of value. And I think since donating it to CNCF there've been a handful of new contributors and more folks using it. So that's available at schemahero.io.

[00:46:34] JM: Well, Grant, as always, it's been a pleasure talking. Do you have any closing thoughts on this ecosystem and plans for the future?

[00:46:42] GM: Yeah. Well, thank you for having me. It's been great. Ultimately we continue to be incredibly enthusiastic about the future of the Kubernetes ecosystem. I think it's come very far, but we're still just in the early innings. Just in the last year or two is when a lot of big companies who were kind of waiting for the container wars to be over who they've come off the sidelines and they've started to really use Kubernetes in a more beyond just a labs function, beyond just a test. They're starting to put it into real production use cases. We're seeing tons of folks within the fortune 100 in the global 2000s who were adopting it across all the different projects.

And so the Kubernetes ecosystem is going to continue to be a very, very important part of the future of software. And I think that you know the shift to remote and sort of everyone working completely independently from their homes in terms of the technology world has shown that software is a core, core part of the world, right? This is how we interact with our loved ones now. This is how we do so much more than we used to do. And I think it really accelerated the idea that software is eating the world. It's almost like software now is the world. And so as such, the tools that are built for software developers are going to be increasingly important, right? So you're going to see more and more developer tools, more and more highly technical products out there that allow organizations to build and create value for their customers through software.

So I think that the trend has always been in our favor, but I think that even more recently we've really sort of – It's been proven how true it is that the future is going to be kind of so software-enabled. And so it's a really exciting time to be part of this industry, to be part of this field. I think that the communities that we're developing and the way that we're approaching this is really special and we owe a lot to the people that have sort of learned from the mistakes

of past projects and past foundations and have taken that and made sure that this organization, the CNCF, and Kubernetes, this entire ecosystem can endure and can last. I think a lot of the vendors have realized they have to play somewhat nicely together in order for the ecosystem to move together holistically. So it's a really, really exciting time.

We're always hiring. We always love passionate software engineers who are curious about technology and curious about just learning in general. So if folks are out there and they're looking to join a company and work on kind of really nerdy infrastructure software problems, we'd love to talk to them and they can just come to Replicated, look at the job openings or shoot us an email. I'm always available, just grant@replicated.com. I'd love to hear from listeners of the show. The response we've gotten from previous episodes has been really strong and I think your audience is super – They're very much engaged in the content and they're engaged in the sort of like pursuit of what's next, and we just love those folks. So thank you for having me.

[00:50:08] JM: Well, thanks for coming back on. Until next time.

[00:50:10] GM: Yeah, thank you.

[END]