EPISODE 1307

[INTRODUCTION]

[00:00:00] JM: In today's containerized world, it's common to encounter similar issues with known solutions across multiple pods. For most people, there are two solutions, go pod by pod finding and fixing the problem, or do that while also spending months trying to automate that process. This is significant time and manual labor. The company, Shoreline, orchestrates real time debugging and automated repair across fleets. Shoreline makes it easy to define metrics, alarms, actions and bots from the CLI to take an action like draining and terminating nodes tagged for retirement or finding TLS certificates that are close to expiring. For unknown issues, Shoreline lets you debug across every pod from a single CLI, rather than SSH-ing pod by pod, like running a command to grep for errors.

In this episode, we talk with Anurag Gupta, founder and CEO of Shoreline. Full disclosure, Shoreline is a sponsor of Software Engineering Daily.

[INTERVIEW]

[00:00:54] JM: Anurag, welcome to the show.

[00:00:56] AG: Thank you, Jeff.

[00:00:57] JM: I was particularly interested in this conversation because of your vast experience at AWS. So you were at AWS for seven years, eight months. Before that, you were at Oracle for three and a half years. I want to ask you for your condensed thesis on what a cloud should be.

[00:01:17] AG: What a cloud should be? So what we talked about at AWS was the notion of utility computing, providing compute, storage, database, 140 other services, the way you buy electricity and gas, or telephone or Internet from your local utility provider. And so what you really want in that is something that is simple, straightforward to adopt, metered by the usage. And, most importantly, that stays up. And you don't really care about features or performance

from your electricity provider. What you care about is whether you have power. And so I think that that's – And cost, of course. And so those are the primary two value points I think for cloud provider for whatever service it's providing, cost, simplicity and uptime.

And, interestingly, the more uptime you provide, the more you will rely on something and the more you need. My mother in law lives in India. She's got a pretty big generator, because the power goes out all the time. I don't have a generator. But the power goes out sometimes, and then I'm just out of luck. And so I think that's also been part of what's happened as people have moved from on-prem to cloud. The availability has gone up of the underlying infrastructure. But therefore, their resilience characteristics have gone down. They just rely on their cloud providers.

[00:02:59] JM: Of course, the utility analogy starts to break down when you think about the fact that utility companies just optimize their electricity systems. They don't move up market into what can you do if you refine that electricity. They don't move down market into what happens when we make our own generators, for example? Whereas AWS has this vertical as well as horizontal product expansion. And since you became a VP at AWS, I'd love to know, what is the product expansion strategy? It's like how does AWS decide what products to not make?

[00:03:40] AG: So it's very much a bottom up, as well as top down process. So once a year, all the VP's gather in the room, and all of the service teams basically present their last 12 months and their projected next 18. And at the end of that, you get your headcount for the next year. And then later on, you also figure out what do you want to do from a revenue and margin perspective, which determines the hardware that is going to need to get acquired. And so, basically, people and the underlying infrastructure costs are the major cost drivers for a cloud company like AWS.

And so that's where you start to do this initial decisioning around, "Hey, what should we go build?" And then beyond that, everything goes through this working backwards process, where you go and huddle in the room with Andy Jassy and a handful of people to describe, "Okay, let's assume we built this thing and it's everything that we want it to be. What would the PR look like? What would the FAQ look like? What would a narrative look like to say like, "Here's the reason to get into this," and so forth?

So it depends on what the service is, what its objective is going to be. There are services where the objective is to make money. There are services where the objective is to reduce friction for other services making money, and so on. So there are a lot of reasons why you might get into a space. But finally, it's an investment decision. One of the things you may have noticed is that AWS very, very rarely turns off services, essentially never. And part of that is this basic thesis that, hey, if it made sense to get into, it makes sense to stay in, because the original thesis probably continues to hold. The only thing that maybe the issue is more likely to be a gap in the execution, rather than a gap in the basic investment philosophy of going after a space, because that's relatively unchanging. Does that make sense?

[00:05:59] JM: It does make sense. Now I know we're getting a little bit deeper into speculation, and I want to come to Shoreline eventually. I just want to prod your brain a little bit about how you think about the cloud business. So when you look at the product category of Outpost, and also like think about the product suite. So you have the product suite Outpost. So Amazon Outpost, Amazon – What is it? Nitro? The chips? Those are the chips, right? The Nitro chips? So you have the Outpost product, the Nitro chip product, and you have the Firecracker product. If you think about that product suite, what are the kinds of things that can be layered into that stack to further cement Amazon's advantage?

[00:06:42] AG: So I think that's an interesting point. So the first point there is Jeff famously said at one point, “Your margin is my opportunity.” And so the notion there is if I'm paying you for money, paying you some money for something, then you're obviously collecting margin on it. If I have the ability to create something myself that does the same thing, let's say a chip, then I can collect the margin equivalent, assuming I have enough volume to be able to run that business effectively. So that's one notion. The other notion is, of course, that I get to get onto an innovation curve where I understand how I'm using something and I can build custom for that, as opposed to what you're doing, which is providing a general thing that's suitable for many customers. So that's one part of it.

The other part of it is the idea of building a moat. And what I mean there is, like Jeff once told us in AWS like, “Hey, you've built this great castle in AWS. Now all these barbarians are coming to storm the castle. You know what your castle needs? It needs a moat.” And so moat in this sense

is really the technology moat. What are the things that you're doing that can basically reduce your costs, improve your value by doing things that are custom that other people can't buy? And the more you build those things, the more you grow, the more things that are available to be built cost effectively.

So for example, obviously, AWS buys a lot of networking cards every year. Eventually, they bought Annapurna and then they started building their own. If you save a few cents per networking card, that actually is a meaningful thing. It's well worth bringing an engineer on. An engineer you're on. And then, over time, that can help you reduce your costs, reduce your prices. And then the basic assumption and the virtuous cycle there is that, as you reduce your prices, you get more volume, which lets you get more scale, which lets you find both more information out as well as take on more things that help you further gain efficiencies. And that's been true in retail. It's been true in AWS. It's one of those sort of cycles. If you're in a big enough business, more things are worth doing as you get to scale. Makes sense?

[00:09:17] JM: Yeah. Alright. One more general – Can I ask you one more general cloud question?

[00:09:21] AG: As many as you want.

[00:09:23] AG: Alright. Okay. So this category is just so interesting to me, and you have a very, very good set of insights into it. So, actually, maybe a few more. Okay. So one is I've always wondered to what extent the Windows platform is a desirable gold standard lens for AWS to sort of think of itself through. Because like on the on one side, like Windows has this beloved developer platform. It has massive adoption. It has people who are never going to go elsewhere. On the other, it's sort of classically regarded as a little bit overpriced perhaps, like perhaps a little too adversarial. Is it a stretch to think of AWS as borrowing some of the desirable features of the Windows business line? Or do you think that's just like kind of far-fetched and not really come comparable?

[00:10:16] AG: I think the wonderful thing about Windows and the overall Microsoft developer ecosystem is that things plugged together very well.

[00:10:30] JM: That seems like the most positive side of the analogy.

[00:10:34] AG: Now the thing about AWS, in some ways, you can think of it as a collection of Lego bricks that let you do build almost anything. But it's not like this – There's not just one brick that fits into one other brick. AWS favored choice, and letting the customer or developer decide more so than like basically saying, "Here's the things that just plug together super well." I mean, there're some things like that. Like everyone must work with their authentication and identity management, right? There's no way around that. You're not going to have to providers and AWS for that. Similarly, things like VPC. But leaving those things aside, you have a fair bit of choice in what you can do. And they pride themselves on choice, innovation, and all of those things.

When you compare to Windows, I mean, Windows is a huge business and important business for AWS. I think they talk about how they have a larger Windows business than Azure does, which is remarkable. The difficulty there is, of course, you don't get that virtuous cycle of reducing costs year over year there because you're paying money for the license to Microsoft to run them. And Microsoft isn't particularly incentive to reduce your costs.

[00:12:00] JM: Okay, I love that analysis. Alright. Okay. So one more question in this direction. And I feel we're building up your credibility in the AWS department. So that gives us a really nice little springboard into the discussion of Shoreline. But this last database related question. So you were part of the AWS database services team, which includes Redshift, and Redshift being the data warehouse within Amazon that is growing extraordinarily fast, even though it's a business. It's been around for a while already. What's the long term strategic positioning there versus Snowflake? I mean, is it sort of like were just like Amazon does an awesome data warehouse experience and like we don't really think about Snowflake? Or do you like look at snowflake and you find the desirable characteristics and you sort of integrate those? What's the strategic positioning there?

[00:12:57] AG: So it depends on who you are. So if you're an executive within AWS, your interest in, let's say, a sales executive, or Andy Jassy before he became CEO of Amazon overall, your interest is in providing customers choice. You don't want to put – you're basically saying that, "Okay, if you prefer Redshift, great. If you prefer Snowflake, great. If you prefer Teradata cloud, great." So you are not trying to give one leg up over the other.

If you're inside the Redshift team, your responsibility is to make Redshift grow as fast as possible. So you look at your various competitors. You look at what they're doing. You think about what innovation you can provide. And most importantly, you think about how to serve your customer, ignoring competitors, and just trying to do that job better than anybody else is doing it. And so that's really the core approach if you're in product, let's say inside the Redshift team. And it's no different than anything else. Data is an important business and for AWS, obviously. And so being able to get data workloads from on-prem into the cloud and grow in the cloud is hugely important. How people do that at some level? Snowflake is selling in AWS. They're still buying compute and storage. If Redshift is selling in AWS, it's basically selling. And presumably some margin advantage over compute and storage.

And so as long as you're getting data workloads in, I think if you're Andy Jassy, you don't terribly care how they're doing it. Obviously, strategically over time, it's reasonable to believe that the various assets inside the data services at AWS will link together better than, let's say, other things and are going to create a little bit – People will be less able to move to Azure or GCP, where of course, Snowflake also runs.

[00:15:14] JM: Okay. We've set the table sufficiently. Obviously, have a deep domain expertise in AWS and cloud more broadly. You've got plenty of background that supports whatever kind of infrastructure business you could want to build. I assume you could have – After more than seven and a half years at AWS, you could have literally stood in front of a whiteboard and just wrote down business ideas till the end of time. I mean, if you spend that much time at AWS, you must have that many business ideas. What is the thesis for Shoreline? And why did you pick that business line over all others?

[00:15:51] AG: It was really based on the pain that I saw at AWS, and more importantly the pain that I saw AWS customers facing. So you go into a company like AWS, and you're experienced building products that you launch and push and other people install and so forth. What I learned at AWS is like how to run services, and the importance of availability in services. So when I started at AWS, they gave me eight people and said, “Go disrupt data warehousing and transaction processing.” So that turned into Redshift and Aurora. And eventually, I also ran things like glue, and EMR, and RDS and a bunch of other stuff. But those are the first two and

those were the fastest growing services after they launched for AWS. So those are good businesses to get into.

But what I learned there is that it wasn't so much about building the engine technology. It was that, of course. But about half my overall teams were doing the operations and the control planes and the work necessary to make these things simple to use and scale easily, and deal with outages and stuff like that. And so I think people underestimate as they go from being product companies to being SaaS companies how much they're going to need to do there.

And so what Shoreline is really focused on is production ops, day two ops, sort of incident automation. So there have been some great, great companies that have been created around observability, or incident management. And you need those things. You certainly need to know what's going on. You need to know how to route somebody to work on it. But I got to say, I never got that excited about one more dashboard to look at, or slightly better process management to shrink slightly the time before you got the right person looking at something. What got me excited was extinguishing tickets once and for all and through automation. So we're really an incident automation company. We think of that as the third big area in production ops. And what we're trying to do is say that, “Hey, for the things that happen again, and again, and again, why is the human being doing it?” How can I make it a matter of minutes to construct an automation to extinguish it once and for all? Because your fleets grow a lot faster than your ops team, and your ops personnel do move around, and they carry their expertise with them. And so being able to turn it into software makes a big difference, because the software will be there, the software will scale in a way that humans won't. And the software will also take a remediating action far faster than a human will. And it can be improved. There are bugs in software, always. But they can be extinguished too in a way that humans just make mistakes. I mean, you wake me up the three in the morning and ask me to go fix a box. There's a decent chance, at least a 1% chance I'm going to screw it up.

[00:19:13] JM: I don't know about that. I am pretty compelled by your vision. And I see essentially a torrent of opportunities for better cloud experience platforms in the near future. We are so in the dial-up days of cloud is my opinion totally.

[00:19:29] AG: I totally agree.

[00:19:31] JM: Like why do you agree with that? Why are we in the dial-up days of cloud?

[00:19:35] AG: I mean, right now cloud is still at the phase where it's about moving workloads from on-prem, sort of lift and shift. There's some amount of re architecture with things like Lamda and so forth, which are super interesting. But what isn't really there is improvement on operability. And operability is the big problem, and availability is the big value you're providing your customers, then that's something that needs to be done better.

30 years ago, if you were an operator, you'd basically get a page and you'd go and crack open your compact laptop or something and then you'd go and try to VPN into some box and fix it. How is that meaningfully different than what you're doing today? It's not. That's nuts.

[00:20:37] JM: So if I think about – Let's say I think about Shoreline as a universal layer of tools and technologies that I'm going to add to whatever cloud experience I have. Let's say I define your product that way. Maybe you don't define it that way. But that's what kind of what I see. What are the fundamental cloud products that you can offer in a differentiated fashion?

[00:20:59] AG: So what we're really trying to do is just improve availability uptime, reduce labor, reduce errors. And basically the assumption that is, is that incidents will happen. You've got software you rely on. You've got hardware you rely on. All things break. The question is how do you get back to normal as quickly as possible? So it's really a question of how you build in resiliency and how you reduce the time to detect and the time to repair through software. So that's our focus. That may feel somewhat narrow, but we think it's serving a very important part of the landscape for cloud operations, which is really everyone who's involved in production on-call. And we actually think that's a pretty underserved group. There are tools for them. But there isn't anything that actually meaningfully reduces labor.

[00:21:58] JM: Can you go deeper on that?

[00:22:00] AG: What I mean by that is – So there are a bunch of observability products. There are a bunch of different incident management products that will page you. There are tools you can use for Slack ops that put a team together. There are tools that will help you with

postmortems, things like that. How do I actually reduce my workload? I reduce my workload by having the repetitive mundane work done for me. I mean, that's what computers are for. And what a human being should be doing is dealing with the things that are happening the first time or the second time, or in some unusual way that requires some judgment and some intelligence. And it can't just be click through this workflow in a mechanical way.

And even for those, what we do is an interesting thing where we provide real time fleet wide debugging and repair. So in one way you can think about Shoreline as it's kind of like Splunk, except it doesn't have any lag. And it lets you change stuff, not just look at it. And so on the diagnosis and repair side of things. On the automation side of things, you take what you built on that first thing, and then you convert it into an action in a bot so that you can generate an alarm and just automatically repair.

[00:23:25] JM: Splunk, but with no latency?

[00:23:27] AG: That's the real time stuff. So if you take something like Splunk, you have to index your logs before you can do it. They have to go off-box. Go into big latent. They get indexed. They get searched. And now it's searchable. In Sshoreline, if you want to go and let's say grep an access log for warnings, you just go and say, “For these resources with these tags, where this metric is awry, run a grep command,” and it'll just fan-out in a distributed way in parallel. Run the command and bring you the results back. And so you're basically managing your fleet as though it were a single box. And the assumption is, is that you know how to manage an individual box. But what you don't want to do is play whack-a-mole SSH-ing into box after box to run commands.

[00:24:15] JM: Alright, I need you to go a little bit deeper on that. So you said manage your fleet like it's a single box, right? Is that we said?

[00:24:22] AG: Yeah.

[00:24:23] JM: So like what does that actually mean?

[00:24:25] AG: So there's basically two parts. So what we're talking about here is the real time diagnosis and repair of a new issue. So something happens. Let's say latency is high. I don't know why. I don't have the right metrics. It's not something that's happened before. Or I've tried a bunch of automated steps and they haven't fired. So it eventually went to a human. So what do I do as a human? I want to go and look at a subset of my fleet that's related to this portion of the thing. I want to run some – I want to look at the metrics on there in real time up to the second so there isn't any latency. I want to do that against the resources again in real time because my containers are coming up and down. And what I want to run on those, if the metrics aren't enough, is Linux commands, to look at things like disk usage, or the output of a top command, or whatever. And then once I diagnose what's going on, I want to run the command across that subset of the fleet to fix it.

And so in the case where there's something already built, I didn't have to do that. But if there isn't anything built, the only choices is something like Shoreline, or SSH-ing to fix things in a box after box. So that's what I mean. Basically, in terms of managing your fleet like it were a single box. We give you a console that lets you run across your boxes. Filter the resources that you want to operate on. And it's basically a simple connection between resources, metrics, and Linux commands, which can just be pipe delimited together.

[00:26:06] JM: Okay. This is a very powerful idea. We're going to need to talk about the high-level API, or the like way into this platform, whether it's API or like user interface or something like that. We're going to talk about that first. Then we're going to talk about engineering. So like let's give people the full stack understanding what's going on here. So at the most external level, I'm the developer configuring this thing. First of all, why am I doing this? And what am I doing?

[00:26:35] AG: So you're doing this to start with, because you have an incident and you need a way to figure out what's going on and then to fix it. So we have a web UI, we have a CLI, command line interface. We also have a Terraform provider so you can deploy things over time. But anything you can do in the web UI or the CLI, you can do in the other. So it's just a taste question. Obviously, with CLI's, you can integrate them into scripts and so forth, which some people like me just prefer CLI's.

So inside the COI, we have a domain specific language called Op. Op is basically a fluent integration of resources, metrics and Linux commands. So you can do something like grab all my containers and then say pipe, where this tag, let's say application namespace equals bookstore, if you want to say something like just look at my bookstore containers. And then I might say pipe CPU usage. Just grab the CPU usage. Or I might say CPU usage greater than 80% to grab the resources where the CPU usage is high. Or just run the succession of pipe operators. That's why I compare it to Splunk. And you can also pipe in Linux commands. So like where those things have happened to date? I want to run the top command. Run that top command and give me the output. Okay, now for these particular subset of those, run these other things. So it's basically the same thing you'd be doing against boxes. One box, but it's doing it across your boxes, and using dynamic filtering to decide which boxes you want to run on either based on the tags, the resource tags, which we automatically get out of Kubernetes, or EC2, let's say, in the case of AWS. And then you can basically just cascade those things together. So that's what you do on the surface of the product. Does it make sense?

[00:28:40] JM: It does.

[00:28:42] AG: Let me just say one more basic thought. So there are two layers of composability. You can compose together resources, metrics, and Linux commands inside an Op statement. And then the second layer of composability is you can compose together metrics with alarms, with actions, using a bot. That's how you can construct your automations. You basically use conditional logic against your metrics or output of shell statements to generate an alarm. When the alarm fires, you can associate it with a bot, which can be either a Linux command, or a shell script, or a call into AWS CLI, or anything you type at the Linux command prompt and basically bind it together with if this, then that bot.

[00:29:33] JM: So this is a step forward in conceptualization of what orchestration in the cloud is. Like you're basically saying, “Look, cloud at a fundamental level is three things, metrics –” What is it? Metrics, objects, or something?

[00:29:50] AG: Resources. Yeah.

[00:29:51] JM: Metrics, resources and –

[00:29:54] AG: Linux commands.

[00:29:55] JM: Yeah, metrics, resources, Linux commands, which is like all you need for a cloud-based reactive programming model where you basically say like, “Okay, metrics are your health. Like metrics are like your – What's the Linux command where you like just get like the health check or whatever, or like the like list of things? I don’t know.

[00:30:12] AG: Top, for example.

[00:30:14] JM: Top. Yeah, top. That’s what I was thinking. It’s like top. You want to top at all times. You basically want your infrastructure to react at a fine-grained level to top. But like not just fine-grained, but like the right level of granularity. And the right level of granularity is to perform Linux commands on resources?

[00:30:32] AG: Yeah, that's exactly right.

[00:30:34] JM: Okay, interesting. How'd you get there? How do you think about that? That's pretty sharp. That's like very concise.

[00:30:41] AG: So I spent a lot of time in databases doing SQL. And what SQL is underneath the covers is an abstraction over what are the unique objects and how do I try to treat them in a consistent way? And so that's what we're trying to do here. I mean, the differences is that I don't think operators want to like type the equivalent of SQL statements. That's not the world they live in. The world they live in the shell. And so just like Splunk built a pipe delimited language to say like – And that's how they do their group bys, and filters, and just tell me which data you want, and where do you want it. We're doing the same thing. And so we're doing it totally differently. But underneath the covers, what is a SQL statement? Where do I want to get data from? What data do I want to get? And how do I filter it? So that's sort of part of it. And then of course, I want to take action on it. So that's the other part of it.

[00:31:44] JM: Cool. So okay, it's a great paradigm to be playing in. Of course, the dangerous thing about going for a new paradigm is the go to market strategy. Like how do you convince people to buy into this concept?

[00:32:00] AG: So you're right. Actually, one of the challenges with a product like Shoreline is that you're actually changing your environment, right? It's not just looking at it. It's much easier to sell an observability tool, I think, or a process improvement tool. You're not sitting inside the infrastructure to change it. So I think it will take some time. But 10 years ago, no one would have let Kubernetes do an OOM killer. And yet it's an obvious piece of infrastructure that we do right now.

[00:32:36] JM: I'm sorry. OOM killer? What did you say?

[00:32:38] AG: Yeah. Sorry. An out of memory. Kubernetes, you can set it up so that when it's out of memory it automatically kills the container and brings it back up. So for things that are transient and not stateful, or the issue goes away after you kill it, it works pretty well. There are issues there of course. But what we're trying to do is apply that same concept across all the rest of your infrastructure, VMs, and containers, as well as all of the things that aren't repaired by simply bouncing the box, right?

[00:33:14] JM: Okay, got it. Interesting. Interesting. Interesting. Interesting. I mean, what if you made this like – What if you didn't try to sell to individuals? What if you tried to sell to like burgeoning cloud providers? Like what if you basically said like, “We're the platform to build your next cloud –” Like think about Firebase, right? Like Firebase is a card company to build. It should be an easy company to build. What if you just said, “We're the Firebase built in platform?”

[00:33:39] AG: I get exactly what you're saying. I think where we are – It's hard to pitch into the hyperscale cloud providers, obviously.

[00:33:50] JM: But you're not. You're the indie cloud for clouds.

[00:33:52] AG: Sure. Yeah. So what we're trying to do is sell to people who are SaaS companies who are managing large fleets. Because they happen to be running their infrastructure on a variety of cloud providers, it helps to be able to up-level what you're doing into just a notion of Kubernetes and Linux and deal with all the variety of infrastructure you're using, like, let's say, it's Kafka, or RabbitMQ, or Kubernetes, or whatever disk subsystem. That list just goes on and on and on. And so being able to manage the typical faults in that is useful. So that's who we're sort of pitching to right now.

I thought initially about pitching to some of the smaller kind of cloud providers, but I think software as a service is perhaps where you're going as well. And so that's really is the initial version. I also think managed service providers will find this interesting, because there they're actually going and taking on the operational pain of running somebody else's environment for them. And so something like Shoreline proves their service level objectives and improves their margins because they don't have to apply as much labor.

[00:35:09] JM: Can I take my like cloud for cloud, or hipster cloud for cloud thing a little further and ask you? So have you looked at render.com?

[00:35:19] AG: A little bit?

[00:35:21] JM: So render.com. So I put a small investment to render.com. I'm an investor, full disclosure. But I invested because I believe they're the next cloud. Like I basically believe that they are the next major player in the cloud. Like I love Netlify. I love Guillermo's company, Vercel. But like, ultimately, Render just defeats them all, because it's fundamentally better. So the founder, Anurag Goel, you should talk to him, because – I don't know.

[00:35:48] AG: I'd love to not.

[00:35:50] JM: I think it could be a really interesting conversation. Like I almost think about like – Basically, he wants to build basically like cloud reimagined with the developer experience in mind. It's like – He was like employee number nine at Stripe, I think. So he's got like basically the best mind for developer experience of anyone in cloud. And so his vision is sort of like top down – It seems like the AWS thing, it's like top down and bottom up. Like top down, since the

developer experience should be like 10X better than the second best player. And then bottoms up in the sense that like we're going to configure infrastructure better than anybody else. And we're going to make products all up and down the stack. And I almost think like your paradigm is what he should be using at the lowest level. Like, fundamentally, whether explicitly or implicitly, you've basically invented a design pattern that should be used by any major cloud provider.

[00:36:38] AG: Yeah. I'd love to talk with him. I think whether it's somebody who's creating the new infrastructure cloud platform, or it's somebody who's just saying, "Hey, I've got this game, and I just needed to run reliably across my millions of users." It's kind of what they're actually doing, whether they go all the way up to application or whether they stay in infrastructure. They all need to deal with availability in a fairly consistent way.

[00:37:05] JM: But, I mean, not just availability, right? It's like availability is like the defensive posturing. But like we're getting into the time where we can play with offensive posturing, even as a cloud provider. What are the most ambitious services we can make, right? That's where we want to get to, where we're not even thinking about availability.

[00:37:20] AG: Internally, you have to think about availability, right?

[00:37:24] JM: Don't you get to a certain point where you're more thinking like just at the pure product level? Like when you get to like the Apple level, like we're just thinking about the product. We're not really thinking about like is iCloud going to sink on time?

[00:37:36] AG: It depends who you are, right? Let's imagine you're EC2. Pretty big business. Now, inside EC2, there's still someone who's thinking about like things like how do I reduce my cost of heating, or cooling rather? Or power availability, and all of that stuff? Or how is my rack stuff? So that's the offensive posture, and so forth. But you also just have to think about things like how do I keep my systems up? What is changing in the software? What is changing in the hardware? Etc. And some of that, it's really a question of timescale. So there are multi-year projects like chip redesign. And then there are things that you have to deal with right now, like the consequences of some particular thing that is bouncing, or flopping, or whatever.

[00:38:35] JM: Okay. Well, very interesting. So moving back to go to market. So what's the – I'd love to kind of get a sense of like the timeline, like strategic timeline. Like how are you going to market right now? Are you doing POCs? What's your go to market strategy today and where do you want it to be in a year?

[00:38:54] AG: So, right now we are – We're a few weeks away from GA. And up till now, we've been working with design partners and making sure we have something really that's providing fundamental value to them and dealing with all of their core needs, that there's no reason for them to say no. And the way I decided that something is production ready is my customers tells me, “Hey, I'm tired of design. I just want this thing running. Because I just had this issue, and I actually deployed Shoreline in to just figure out how to deal with this issue. And so I don't care that you're not production. I'm running it in production when I need to.” And so at that point I'm like, “Okay, fine. Pay me money.” And so we're just now starting to get those signals from early design partners. And so for me, that's where you consider yourself to be GA.

And so now we move into a phase where we do pilots with people that define the path to production. And for me, the path to production is, “Hey, every week let's meet up. Let's talk about some core issues you want to extinguish forever. And let me build an Op pack for you that just extinguishes it. And hopefully in four or five weeks, you're at the point where you can build those things yourself and you're ready to run.” So that's still early stage. Hopefully over the course of a year, I get to the point where there may be a hundred Op packs, so that out of the box, you're getting something that deals with all of your core infrastructure problems, or you're using 20 out of the 100. And so you get something that's been sort of battle tested by other companies on those. So that's a good place to get started. And then the other thing is, is that maybe we really do want to work with partners in the observability space, in the incident management space, because the obvious next thing for their customers to do is to automate those things away.

[00:40:55] JM: It does make a lot of sense. So you're thinking about basically selling to like the Splunks of the world and saying like, “We're going to help you manage your Splunk infrastructure better.”

[00:41:02] AG: Both help you manage your Splunk infrastructure better. And hey, you've got customers for whom you are use, let's say, signal effects as part of Splunk. You're providing them observability. Once you generate an alarm, wouldn't it be great if they could then just automatically fix a bunch of them?

[00:41:21] JM: Okay, I just thought of another person who you need to talk to. Do you know Avi Freedman from Kentik?

[00:41:28] AG: I may. I'm terrible at names.

[00:41:31] JM: Kentik is this network observability company. Have you heard of it? Maybe you haven't heard it. So anyway, it's a great network observability company. He's a friend of mine, a high-stakes poker player also. But the hilarious thing about Avi is he's so insistent on running his own infrastructure, because he started one of the first DSLs in – Was it Denver? Or not DSL. ISP. He started one of the first ISPs. So anyway, he's much a run your own infrastructure kind of guy. Very cost effective. And I was just thinking while you're talking, like can I use this if I run my own servers? Or like do I have to be on AWS?

[00:42:08] AG: I'd need to talk to him. There isn't any deep, deep integration into AWS. The deep thing that we need out of some cloud provider or somewhere is the ability to discover the resources. And so, for example, I might get tags out of EC2 or out of Kubernetes. So I'd need someplace to be able to get the tagification that someone's using to define what the infrastructure is. And then of course, if I want to do something like grow a disk, right now I call an EBS API. So they need to define the actions on what they need to change and how to change the environment in their world, right? Because I'm sure they're doing it using software. Not having some guy roller skate over to a rack and the plug in a bigger drive. Those days are gone for all of us.

[00:43:09] JM: Can you tell me anything about POCs? Or like early conversations? Or like level of interest from customer base?

[00:43:15] AG: We're getting considerable interest, which is, I think, a sign that there is a deep and abiding pain. So that's great. I mean, for us. I mean, I feel bad for my customers, but

hopefully I'll do something to help them. The challenge we sometimes see is, is that where there's pain, people are also very busy. And so like they can't really look at how to fix something when they're busy fighting a fire right now. So we do get a lot of rescheduled meetings, because someone's like, "Hey, I've got a live event. I'll talk to you later." But I think there's active interest. And I think it's going well.

What you mean by that is, what are the things where people really resonate with pain? I will mention that, hey, I've got a disc op pack, which auto grows a volume and with some guardrails on it. They're like, "Oh, I deal with that four times a day. I'd love that." Or cert rotation thing. Oh, "Hey, my JVMs start garbage collecting, what I really want to do is collect the heap dump, stack dump, garbage collection, stats, move it into S3 and then bounce the thing. I don't want to do that manually."

What frustrates them more than anything else is all of the things that they do again, and again and again that just feels like a waste of time that a machine should be doing for them. So those are the things they're excited by.

[00:44:57] JM: Gotcha. What do you want your business to look like in two years? Like does it look like you've got like Splunk, and LogRocket, and Datadog, and – I don't know. What's another like customer category you can go after? Like what's an easy customer category to go after?

[00:45:13] AG: I think they're incident management tools.

[00:45:13] JM: SignalFx.. All incident management tools. Yeah, got it. PagerDuty maybe.

[00:45:17] AG: Yeah. So I think we'd want to integrate with all of them, or certainly big ones where we could. We'd want to integrate with cloud providers as well. Like get on to the AWS partner network, and all of that kind of stuff. Most importantly, I want to be serving customers, right? And, basically, I want them to be able to say like, "Hey, my on-call shifts are now uninteresting. And I'm spending my time during them doing my other work, like cost optimization, or security optimization, or work that requires intelligence as opposed to just basically playing ticket whack a mole."

[00:46:01] JM: So the design pattern for selling to Datadog, is it – I'm not quite understanding. So if you're selling the Datadog, are you saying to Datadog, "We're going to help you manage your logging infrastructure better? Like we're going to help you garbage collect –"

[00:46:13] AG: No. That would be a partner, not a customer. So the notion with a Datadog, for example, aspirationally would be, "Hey, you generate a monitor in Datadog. Wouldn't it be great if you could just bring in a side panel that looks just like Datadog, but in terms of UI, but let somebody actually fix it right there inside your environment?" So that would be ideal. That's a different kind of thing than, let's say, a direct customer. Let's say, someone who does online games like EA, or Epic, or something like that. In their case, you're just trying to keep the systems running, because any minutes of downtime costs that money. And so you're just trying to make sure that their environment is serving their gamers well. So that's a different kind of – That's a direct customer in that sense.

[00:47:12] JM: Gotcha. So the arrangement would be like, "Okay, so once again, we revisit the API or the user interface for Shoreline." You look at metrics, or you're able to define metrics, understand metrics across your distributed infrastructure. You have resources across your distributed infrastructure. And then you have Linux commands that you're running on swaths of infrastructure, like a set of different infrastructure boxes, right? That's the API?

[00:47:37] AG: That's correct. And so the API is basically here's this – Run this DSL command, which is a pipe B, pipe C, which might be something like host pipe, CPU greater than 90 pipe top. So that way it just combines it all together. You just send in a string, and you get back a string, which gives you the outcome. So that's the basic model. And then you can use that to build other objects like alarms, actions, bots. And we'll, on every one of the boxes, scrape thousands of metrics, compare them, derive thousands of derived metrics, compare them against hundreds of alarm conditions, and take actions all within a second. So we do per second metrics and evaluation, because we really believe that time matters in these spaces. There's a lot of tech underneath that as well, of course.

[00:48:39] JM: We're nearing the end of our time. We've kind of danced around several different topics. What are the major points that you'd like the listeners to walk away with this conversation with?

[00:48:51] AG: I guess, what I'd say is if you have ticket pain, if you believe it's going to grow, we're a useful company to talk to. And just, too, we would love – We're early stage. So we'd love to get feedback on how to do our stuff better. But we also think that we're starting to serve companies in meaningful ways at reducing their ops pain. Maybe we can do the same for you. So underneath the covers, there's lots of tech, whether it's signal processing, or distributed systems, or database query optimization or numerical methods, that we've adapted into this domain in a way that I don't think anyone's done before, which is why I think we're able to do things that other people don't do. But at the end of the day, that doesn't matter. What matters is the value you provide. And so I think what we're trying to do is reduce tickets for people.

[00:49:59] JM: Got it. Really interesting conversation. Thank you for sponsoring the show. Thank you for coming on the show. I'm very intrigued by the company. I want to help in any way possible because I think you're doing something super innovative. I think cloud is totally just the one dial-up right now and I'm sick of it. So I look forward to you making it a little bit better, at least.

[00:50:21] AG: Thank you so much for having us on, having me on. And this was a great conversation. I really appreciate the time.

[END]