EPISODE 1470

[INTRODUCTION]

[00:00:00] JM: The advent of the cloud introduced a new form of technical debt in which organizations can lose track of what infrastructure they have and how it relates to the business. While the cloud's native APIs offer some transparency into your infrastructure, these offerings are best described as necessary, but not sufficient. When companies have a non-trivial question about their environments, getting answers can be a big cognitive overhead.

In this episode, I interview Tyson Kunovsky, Co-founder and CEO at AutoCloud, creators of the open source project CloudGraph.

CloudGraph is a free and open source GraphQL API for AWS, Azure, GCP and Kubernetes. CloudGraph provides a much more intuitive GraphQL-based interface to ask questions about your cloud infrastructure in context of the relationships between components.

[INTERVIEW]

[00:00:52] KP: Tyson, welcome to Software Engineering Daily.

[00:00:55] TK: It's a pleasure to be here, Kyle. Thanks so much for having me.

[00:00:58] KP: Where does software engineering begin for you?

[00:01:01] TK: Yeah. For sure. I actually had an interesting path into software engineering. And I guess to start at the beginning, which is usually a good place to start, let's go back to when I was in high school. And at the time, my parents actually ran a relationship research company called The Gottman Institute, whose work is based on the research of relationship researcher John Gottman.

If you've read it, you might recognize that name from the first chapter of the book *Blink* by Malcolm Gladwell, which talks about our snap perceptions and judgments and how those influence and colored the world around us.

Any ways, throughout his research of the course of almost 40 years in thousands of couples, John found out something really interesting, which was that he could predict constantly with upwards of 90% accuracy whether a couple in a committed relationship would stay together or get divorced just by watching them fight for five minutes, which is terrifying to think about, but also super cool.

Over the years, John and his amazing wife and fellow psychologist and researcher, Julie Gottman, developed a therapy methodology that leveraged their research in what to do and what not to do in relationships to help couples and therapists with the skills and tools necessary to strengthen and repair their relationships. And it was essentially my parents' job to help the Gottman's take this content that they made and then make it easily accessible to the people that needed it. And at the time, they were doing this mostly through physical products. Think about books, in-person workshops, tutorials, even board games.

And while my parents did a really good job to create the company foundations necessary to bring this research to the world, a lot of parents, though, they weren't the most tech-savvy individuals. And I distinctly remember sitting at dinner with my parents one night and they were talking as usual about work, and strategy, and expanding their audience, and how to make that happen. They kept going back and forth about ideas for marketing, growing their business and their physical product line. And while that was definitely an important aspect of their business, I remember thinking distinctly that they were missing something pretty obvious. Because to me, as a product of the Internet age, it seemed clear that it could be a good idea to digitize this wonderful physical content that they created and use that to really amplify their message.

At dinner that night I told them that I had an idea for making an online platform that would take materials they created and then make those materials accessible online via an LMS, a learning management system, and then allow anybody to sign up and use it. And they were super intrigued by the idea.

And I remember I got a budget of $5,000 dollars to build a first version, which, by the way, back then seemed like a whole lot of money for an entire software project. And I decided to help them build it out. And it turns out the platform became pretty popular. And I spent a couple of years after college helping my parents create more digital content. This was back in the good old days of jQuery, Backbone Marionette, back when DHH was doing this thing and Rails was becoming popular. And that's how I initially got into software engineering to help my parents.

And as a side note, it turns out that working at a relationship company run by your parents would be pretty relevant years down the road. I had no idea. But I've actually started AutoCloud and CloudGraph with my wife and our Chief Product Officer, Evelyn LaTour. And let me tell you, there's nothing like starting a company in the middle of a pandemic being stuck inside and working out of an 800-square-foot apartment together. That's a test of a relationship. And I'm just really thankful for the tools and skills that I learned at The Gottman Institute as they gave me what I think is a really unique perspective that I wouldn't have otherwise.

And then quickly in terms of how I got to where I'm at today, when I was tired of having my dad as my boss, which took a surprising amount of time, by the way, he let me do a lot of cool stuff that I wouldn't have been able to do anywhere else. I moved on, worked as an engineer for a handful of startups at Slalom Consulting. And eventually started my first company, which helped enterprises migrate workloads to the cloud. That company ended up being acquired by a management consulting firm here in Chicago where I became their Head of Technology, and eventually left to start AutoCloud and CloudGraph, which is where I'm at today.

[00:05:16] KP: Well, let's get into the CloudGraph story. What exactly does it do for your clients?

[00:05:21] TK: Yeah, great question. The one-liner is that CloudGraph is a free and open source GraphQL API for AWS, Azure, GCP and Kubernetes. And I guess the longer answer is that, with AutoCloud and CloudGraph, we're essentially attempting to map the entire cloud and SaaS ecosystem and are building out a new kind of API. One that makes it easy to access all of your cloud and SaaS data from a single source regardless of how much data you have or where it's from.

And with this new kind of API, folks who work on the cloud, or just generally in dev, sec, or ops can quickly query data about anything in any level of detail so they can solve a host of operational challenges pertaining to compliance, security, multi-environment billing, asset inventories, you name it.

And I say a new kind of API, because unlike using the AWS, GCP or Azure APIs, or even the new AWS cloud control API, we're building CloudGraph to add three important capabilities that are just not quite there yet today with the cloud service providers themselves, but we think are really important to have.

And the first capability is that data should include built-in relationships between services. For example, say you're on AWS and you query an EC2 instance. When you do that, you should be able to understand all the information about its EBS volumes, the auto scaling group to which it belongs, if it belongs to one, target groups, it's subnet, it's VPC, etc. The connections between resources often tell a story of how a component is used as part of a larger system, which can be really handy to know depending on your purposes.

The second capability is that data should be co-located with insights. Same example for EC2, say you use and you make an API call to do AWS EC2 describe instances. When you do that, it would be incredibly useful to also see supplemental information as well. When you think about it, the most fundamental unit of organization for any CSP is the service, or any CSP is the resource itself.

For EC2, you think of all the data you'd ever want to know about an EC2 instance, such as cost, compliance, utilization. Those all belong to an EC2 instance. Why not associate the resource with its full insight data? If we treat the resource as the primary owner of the data and then begin to supplement that resource with additional data, we can vastly improve the cloud insight retrieval process.

And I guess the third and final capability is that there should be a single open source API that allows developers to query any kind of cloud or SaaS data no matter what that data is.

If you look at a provider like AWS, this is basically what the cloud control API has attempted to do. But this API should extend across AWS accounts and even across cloud providers like Azure, OCI, DigitalOcean, GCP and the others. An API like this that allows for a single unified source of data and one that is most importantly of all type safe. Meaning that, ahead of time, you'll be able to know if your query is valid from a data and connection perspective and can understand all of those different relationships is kind of what we're striving for. That's a little bit about CloudGraph and what it does.

[00:08:41] KP: Well, I know all that information in some way must come it's home of origin. The ground truth is the AWS API. Is it essentially a time saver that you're just going to make all the requests for me? What if I have rate limits and things I'm worried about?

[00:08:56] TK: Yeah, absolutely. I think that when it comes to the value proposition of something like CloudGraph and why the world needs it, you brought up an interesting point, which is rate limiting. Right now, what our system does is go and reach out to the various cloud providers and pull in that data. It's essentially doing a bunch of described calls under the hood to get this data. We're actually working on a real-time system that leverages the cloud provider event buses like CloudTrail and AWS, for example, to make sure that this data is always up-to-date and live.

And in that way, you can think of it as a near real-time cache where you can hit this API as much as you want. You're not going to get rate limited. That can be a problem for some really large internet scale businesses. And on top of that, you're able to get back data, relationships, insights and all this other information that'd be really hard to get and piece together otherwise.

And I think that when you think of just CloudGraph overall, the cloud service providers have done a wonderful job of building solutions that let engineers create systems to power our increasingly interconnected world. And over the last 15 years or so, products like, say, EC2, S3, RDS, Lambda, these have fundamentally changed how we think about concepts like computing, storage, databasing.

And then in the last half decade or so with the proliferation of Kubernetes and serverless, AWS services, for example, have become increasingly abstract on top of racks of physical servers.

And since everything is just an API abstraction, when AWS wants to introduce a new product, they simply need to expose new API. Of course, I'm simplifying slightly. But you get the idea. And this leads to many, many different APIs on different clouds.

Back to EC2, the EC2 API, for example, has over different methods. You multiply this by hundreds of services for each cloud provider, and that's a lot to stay on top of and understand how to use. And while it's a masterpiece of data center architecture, this choice of hundreds of services and configuration options puts the burden of knowledge on how to properly use these services on us as engineers. And as a result, engineers find themselves having to constantly stay on top of and learn about all the different service offerings, the new changes, and all that takes a significant amount of time and mental energy.

It can also be time-consuming and frustrating, say, to use the AWS CLI or the APIs to make five different calls to describe an ECS cluster, its services, task definitions, tasks container definitions, etc. And while AWS is fantastic at building the actual services that power our businesses, I don't think a lot of headway has been really made into simplifying the day-to-day UX of querying these services in the same manner.

The modularity of the AWS APIs is a great logical organization system. It makes total sense. But from my perspective, it's kind of a burden on the end users in terms of cognitive overhead and learning curve. And this is really where CloudGraph shines, right? We're trying to make it much easier to get your data and reflect the need to query any data about any service in any place without having to hold a massive amount of context in your head or spend hours on docs or stack overflow. And by using those three guiding ideas I mentioned, of connections, data co-location and supplementation, and multi-environment access, engineers can use cloud query to query all of their environments from a single place, which simplifies things tremendously and allows for more productivity overall.

[00:12:22] KP: Many companies today are multi-cloud, and it sounds like your tool can help there. I'm curious if you face any challenges around the parity between clouds. I mean, S3 and Azure Blob Store, and GCP storage option, I'm sure they – Outside of like little performance issues, they're more or less the same from a developer's point of view, unless I get into the long tail of features. I think you said, EC2 has 500 API calls. I can't imagine we have a perfect parity

even across these simple services. Are there challenges you have to deal with in the fact that clouds are not all the same thing?

[00:12:55] TK: Absolutely .And I think developers have a certain expectation of how their API's experience should work. If you're predominantly an Azure developer, you're used to the paradigms that you mentioned for storage accounts and blob storage. If you're working on something like GCP or AWS, you're probably more used to storage buckets or GCS.

I think that from our perspective, what we're trying to do, initially at least, is to aggregate this data altogether in the current forms and paradigms that people are used to. I think that there's a really interesting opportunity in the future to standardize this across clouds. And you think of the example you gave storage, or maybe something like networking, where GCP and Azure and AWS do this a little bit differently.

I think that, in the future, as we think of all the workloads that are moving to the cloud and the fact that it's just really complicated to understand the nuances and differences between cloud providers, there's a big opportunity for us to go and standardize this, where you can query storage and you can get back with that storage query all the storage information related to different clouds that do things a little bit differently.

To start, we try to keep close to the SDKs and close to the paradigms that cloud developers currently understand. We think in the future, there's going to be a really interesting play to abstract that upper level and make it really accessible and easy for anybody to just query all of their data.

To your point, a lot of organizations are multi-cloud. From our research, in fact, upwards of 90% of enterprises are multi-cloud. A solution and a tool that makes the data extraction and aggregation easy and makes it so you don't have to have that domain expertise cross-cloud is something that's really attractive from our point of view.

[00:14:42] KP: How much technical expertise does someone need to use the querying functionality of your system?

[00:14:48] TK: Very little. One of the reasons that we chose to build our system in GraphQL, which might seem like kind of a strange choice to start from a DevOps perspective, is that everything is type safe. What does this mean? This means that we have built-in documentation to our system so you're able to see very quickly, without having to do a ton of research, exactly what you can query. You can also, as you're typing, have your queries be auto-completed if you use one of the tools that CloudGraph ships without the box, like Altair, or GraphQL Playground, or you can use whatever you want, like Insomnia or other GraphQL clients.

All you need to do is be able to hook up to a cloud environment and ingest that data using read-only access. That data is then available to you. And you can write queries relatively easily. We have a lot of built-in examples. The only hurdle I guess is being familiar with GraphQL.

GraphQL is pretty intuitive to use. And developers that are used to working in Java and other type safe languages I think have a good way of getting up to speed and understanding how to query and use the system. I guess from my bias perspective, there is a little bit of a learning curve if you've never seen GraphQL or used it before. But it is relatively simple to get up and running and start getting back data immediately and really get that time to value quickly.

[00:16:05] KP: Are there any common queries you see coming across as users are spinning up on the service?

[00:16:11] TK: Oh, that's a good one. I think when we think of just common use cases for a tool like CloudGraph, one of the really interesting ones is answering a question like what is in a VPC? Let's use AWS as an example. And today, if you were trying to answer that question, there would be dozens, if not hundreds of API calls that you'd need to make manually. This makes asset inventory and just understanding what you have running an incredibly painful and time-consuming process.

Say that you knew the name, but you didn't know the ARN or the idea of a VPC. Well, to start, you probably need to run a command to obtain a list of all the VPCs and then find that one that you're looking for. And now that you know the ID of the VPC or the ARN, you can use that to start querying all the things that could potentially be running in a VPC. You might start to run queries to describe the instances. Then for each instance, to describe volumes, and maybe

security groups, knuckles, route tables and so on. Plus, if you have other additional calls you need to make to understand relationships, you need to make those too.

And at this point you've probably spent 20 minutes Googling, reading documentation, running queries, and you only have a sliver of the information you need. And if you're like me, then chances are you actually don't even know all the things that could be running in a VPC. So you still need to figure out how to figure that out.

And rather than having an asset inventory of what is running in a VPC become this Herculean task, if we start thinking in terms of a graph, and GraphQL things become a whole lot easier. Because it turns out that GraphQL, backed by a graph database, is a beautiful and expressive way to query your data. Because every node in a query can know about N many other nodes. We can write effortless queries to figure out what assets are running in a VPC and automatically join across service relationships.

Users can query a VPC. And in doing so, also query things like load balancers, EC2 instance, subnets, etc. And like I mentioned before, it's important to note, because we're doing this via GraphQL, everything is type safe. If we use the built-in query tools that CloudGraph ships with, you get auto completion, auto generated documentation, and you can understand all the possible things that could be in a VPC. You even know if your query is valid before you triggered the HTTP request to get your data.

It's these kinds of common queries. What is running in my account? How do I quickly get that information out? How do I understand the relationships? The supplemental data that we see our users really doing a lot?

[00:18:40] KP: You'd mention that there's a single open source API. Can you talk about the advantages of being open source in that regard?

[00:18:46] TK: Yeah. You look at companies like HashiCorp and Terraform, they've done an amazing job of building out this provider ecosystem where you're able to go and, for whatever cloud provider you're using, use Terraform. If there isn't one that provider that you're currently

leveraging, or you have some sort of proprietary internal system, you can build a provider that adds support there.

We think that this model only makes sense in open source. Allowing people to have full coverage is something that's really important to us. And while we've built out kind of the big, major public cloud providers and some other supplemental smaller ones, we want to allow anybody to essentially add the data that they need, enhance the system and update it to make it useful for them. And that was our choice to go open source from the start.

Ultimately, if we're able to build a community that allows folks to go off and have that full coverage, add in, say, supplemental compliance checks that they want to do, add in the billing data that they need to see, we become this wonderful graph and web of information that anybody can use to power the core of their cloud businesses. One where they can understand all the different data points they need to understand how to solve their operational challenges on the day-to-day. And we just make it really, really easy to map all of cloud and SaaS data for whatever purpose you might need that for.

[00:20:08] KP: Over time, I've observed a lot of technology groups get a little bit bloated. Sometimes maybe a spring cleaning where you find a bunch of things you've been paying for that can be turned off. Are there any quick wins or low-hanging fruit like that that you can help me discover?

[00:20:22] TK: Oh, absolutely. Think of a common scenario which happens in a lot of companies, which is you spin something up. Maybe it's a massive Redshift cluster. Although you should probably be using Snowflake. Or maybe it's just a bunch of instances that you forgot to turn down, right?

What we've seen from our users is a few folks have been able to save up to 30% on their AWS bills by going queering things and saying, "Hey, what is this thing? Look, it was created about six months ago. Is anybody using this? We had no idea it was there." In really large footprints where you have just mountains of data and it's hard to go and just sift through and understand if you're looking in the console or looking at JSON via your command line when you make queries. It can be really challenging to understand the purposes of these things.

And over time, as we hook into things like CloudTrail and supplement that data more, hooking to things like CloudWatch to plot utilization metrics, which we do have support for today, you can understand the average utilization of, say, an EC2 instance or some virtual machine over the course of the last two weeks. And if you're like, "Hey, this isn't really being used. It was created six months ago." You're able to find these outliers, these orphan resources, and terminate them. You're able to do this completely for free using an open source product, which is something that we're pretty proud of.

[00:21:44] KP: Do I need to do anything special related to things like tagging in order to get the advantages of your solution?

[00:21:50] TK: Tagging definitely helps. One of the really cool capabilities that we have is you can query – Let's just use AWS as an example to keep things simple. But say that you have five different AWS accounts. And in your different AWS accounts you have a prod stage, you have tests, you have dev, you have a ton of resources spread across these different accounts. With CloudGraph, you can query all of your different accounts at once. And if you have tagged things, and you say tag things with environment production, you can actually run a top-level search and say, "Hey, give me back every single thing that has a tag of environment production. And then across this specific VPC, which you filter on, give me back all the EC2 instances in it."

It can be really helpful from a billing perspective and an aggregation perspective to add tags to help you enhance the queries that you can run. And that can be a really big added benefit of using CloudGraph. It's not necessary to do, but it definitely makes things easier from an inside perspective.

[00:22:52] KP: Are there any recent releases or new features on the roadmap you'd like to tease or discuss?

[00:22:57] TK: Oh, for sure. There are a couple things that are coming relatively soon that we're super excited about. The first is we are trying to add as much compliance and cloud security posture management data as we can to the tool. For example, today, right now, you can go and you can run queries against AWS. And you can have that automatically check for things like CIS

1.2 in order to go and understand from an overall cloud security posture management perspective are there any potential security holes or violations? You'd be surprised despite the fact that we see headlines all the time in the news for data breaches and negative things that have happened that not a lot of companies have great security stances and great security tooling despite the explosion and growth of security overall in the last few years.

The cloud security posture management side of things is something that's really important to us. We're adding checks for all the major clouds, for Kubernetes. And we're going to go really deep here. There's a lot of companies out there that are expensive companies that do cloud security posture management. We want to build the world's largest open source library of cloud security posture management checks in order to go and allow anybody to check anything they need. We make it really easy to add your own. We've created a simple way to do that. So you can go in, and if you have proprietary checks, if you're, say, a large bank and you need to make sure that you're only using certain hardened AMIs for EC2 instances, you can write checks that do that for a list of approved AMIs. That's one, the cloud security posture management side of things.

And then a second really interesting thing that we're thinking about leveraging is when you think about the way that we're representing this data, which is in a graphical format, it allows us to understand how resources can be connected to different resources. And from a blast radius perspective, if there's, say, some sort of compromise machine, or you need to understand what can this thing possibly talk to from you know an overall access perspective? There's a lot of really interesting work that we can do related to identity and access management and roles and role chaining to allow you to query using the graph. And then every result that's returned, say, from a VPC, then you query a subnet, then you query an EC2 instance, you'll be able to understand what different things can talk to in order to understand the overall blast radius from a security threat issue perspective if there is some sort of adverse or negative event that happens in your system.

[00:25:30] KP: We've definitely got security professionals. I'm sure someone from the CFO's team would be interested in using this. Do you have any common personas of the types of people that need to ask these types of questions?

[00:25:41] TK: A lot of it is day-to-day cloud engineers. Folks that are in dev, or ops in general. I think that a lot of times, we've covered things like asset inventory, or compliance, or security, or billing. A lot of work on the date day-to-day for engineers, where you have to go and make queries to understand your data, to understand the current state of a cluster, or the current state of your database, or your S3 buckets. These folks that are the hands-on people that are doing the work are often using tools like the CLIs and APIs from the various cloud providers.

We've seen a big uptake from the individual contributors and the individual engineers that need to go and understand this data for their day-to-day purposes. And that's actually a big part of our user base right now. These individual engineers who are able to go and now very effectively and quickly query cross account, query all of their data, get back exactly what they need in a very declarative and beautiful way where they don't have to fetch too much data, spend time looking at documentation, go and really understand how these services work. Because it's all built into the graph. You have your documentation. You have your auto completion. And it really democratizes and makes it easier to access that information. Not everybody is the world's best cloud architect. It takes years to truly understand how to build effectively on AWS, Azure or GCP alone, let alone all three. Plus, with the constant changing nature of these cloud providers and the services that they're adding, the tweaks that they're making, a tool like CloudGraph just makes it really easy and consistent to get your data and do so in an easy fashion.

[00:27:19] KP: At what point in an organization's maturity level do you find that they start having the questions that you can best help answer?

[00:27:27] TK: That's an interesting question. I think let's break this down between small businesses, mid-sized businesses, and enterprises. I think for a lot of enterprises, if you look at the typical go-to-cloud journey, probably five, ten years ago, folks started saying, "Hey, we could probably" for the various reasons that AWS is a good idea, which I won't get into, "go to the cloud and save money and do all these wonderful things and really have the flexibility and extensibility to go off and make our digital transformations more successful in this ever-increasing competitive world. And for those organizations, they typically have a lot of really complex legacy systems.

Terraform and infrastructure as code has, in the enterprise of the last five, ten years, become very popular. But it wasn't always that way. You have a lot of stuff that's spin up via ClickOps, spin up manually. And in those large organizations, it can be really useful to have a tool that you can use to audit your cloud and do so in a really easy fashion.

I think that at the enterprise level, it's those bigger, a little bit more complicated legacy companies that weren't necessarily born on the cloud, but that need a good way to manage and understand their assets.

For smaller companies that might be a little bit more mid-sized, those folks typically have systems on the cloud. Maybe they're not as big. But they still do need a good way to go off and understand how those work. Maybe some work was done via infrastructure as code. Some work was done via ClickOps. But they just need a good way to be able to pull out and answer that information.

I think that when they're looking to do a V2 of a system or go off and add new components, that's a great time that we've seen that they've been leveraging CloudGgraph to go off and get this data and really pull it in.

And then for startups, it can be really interesting, simply because you look at modern technology stacks, and the fact that almost all companies now are cloud native and bored on the cloud, it can be really interesting to have a tool from day one that allows you to consistently, audit, secure and understand your resources.

A free tool that allows you to go off at your leisure and pull in that data so you're able, to from a reporting perspective, or a compliance perspective, just understand where you're at and make sure that everything is configured in a best practice way.

There's a lot that we want to do to bake into the knowledge graph that are sensible defaults that can tell you things like, "Hey, are you paying too much for this particular component?" Or, "Hey, from not just a compliance, but a security perspective, what would you need to change this instance over here?" And over time, I think that as we add more of these insights that are co-located with the resources themselves, you'll become more valuable to early stage companies

who are trying to move fast and break things, as they say, in order to make sure that they're operating in a good fashion as well.

[00:30:19] KP: The clouds are established. But in some ways, there are also works in progress. There are new things coming out all the time. Maybe not so much going away yet at this point. But certainly, questions, services, customers of yours will want to be learning about that you'll have to adapt to. What's the process like for responding to the upgrades and enhancements that the various cloud providers put out?

[00:30:42] TK: Oh, that's an absolutely fascinating question. And I think there's both a naive approach and a sophisticated approach to answering that question. The naive approach is, "Hey, let's just go add this new service that AWS spun up." Say, they added some new quantum compute service and you want access to that.

Well, for us to add new service, if we go and do it manually, isn't a ton of work. We need to basically go get the data, pull it in, figure out if there's any connections we can, add bada-ding, bada-boom, we're good to go. That's kind of that naive approach and a way that you can solve that problem.

The more interesting approach and way to solve that problem is, when there's a new API that's released, if you're able to go and make a request against that API and return data, maybe you spin up some test data, there's a lot of stuff that you can do from a dynamic code generation perspective where you can essentially go, and based on what you find, create an entire SDK component dynamically that will go in, format itself, create the calls that it needs to get the data, put the data in the right place, add the connections. And that's what we're really focused on there. It's an uphill battle just because there's so many services for different cloud providers to go and do everything one by one.

And over the course of the next few months, we're transitioning wholly away from that system to dynamic code generation system where you can go and you take an endpoint or a service for a various cloud provider. We pull in that data and we dynamically generate that part of the SDK for you so you can go and use it as your leisure.

And then I think the big picture idea here is, one day, regardless of the source of the data, to allow you to go, enter, say, the authentication credentials as part of the query string that's going to be used in the HTTP request to get the data, and then have an entire provider be created from the data that we find. And we view this tool and this capability as a way to get feature parity. But also to your point, to maintain and ensure that we have up-to-date coverage for little tweaks that are added to different services or entirely new services as they're released.

[00:32:45] KP: When you think long-term, do you have any concern that one of the large cloud providers could take a predatorial interest here and say, "We're just going to kind of clone what you guys have done and build it in."

[00:32:57] TK: Well, there's one I won't mention in particular that seems to do that quite a lot. But I think, from our perspective, as we look to the target users and the folks that have been using CloudGraph a lot, the enterprise companies, the more small and medium-sized businesses, typically there are multi-cloud resources in these organizations, right? You have BigQuery. You have Azure Active Directory. You have AWS, EC2 instances, or whatever those are.

If AWS, or Azure, or GCP, or Oracle, or whoever were to go and build this, there's not a ton of incentive for them to go and make it really easy to use GCP, for example, or Oracle, or AWS, or Azure, whatever cloud that they're competing with. There's fierce battles from a pricing perspective to try and lock people into clouds. It's almost impossible to move from one cloud to another cloud without spending, depending on your size, hundreds of thousands, if not millions of dollars to re-architect the system, test it, get up and running.

While one of the cloud providers might build out a system that's similar to this, in all likelihood, they're probably not going to want to build a multi-cloud system. And that's where we really shine. Allowing to pull in data from any insight or place. And not just specific to cloud data, but SaaS data as well. One day, if you have Salesforce information that's related to information running on, say, EC2, or you have information related to some sort of other CMS or thing that you have, we want to allow you to pull that data in and have it be part of the overall knowledge graph. In this way, we're looking to build the mind and the overall organization system for all organizations that are running on the cloud. While there is a possibility that one of the cloud

providers will build out the functionality that we have, we think that we're okay. We'll hopefully survive.

[00:34:44] KP: Well, it seems like at least once a year, maybe more often, maybe a little less often, but we hear it generally makes the news, there'll be a big cloud outage. One of the major providers is down for some reason. What's my experience like as an AutoCloud user making queries in a time when the actual service provider is unavailable?

[00:35:03] TK: Excellent question. Right now, today, that is basically you're going to have an outage experience and you're going to get a response that says, "Hey, we can't find this data. Or this data is unable to be returned." The good news is that because our system takes historical snapshots. We're able to go and allow you to at least query some of the data that you were able to pull in recently. While this won't be helpful in the immediate term if there's an outage, you're at least able to access things that don't change very much.

You think of a VPC, in any sort of production environment, you're probably not spinning up – Well, you might be. But you're probably not making a ton of changes to the VPC itself. Maybe the components in it, auto scaling groups, EC2 instances, they're at least able to pull some of that information out.

Even if the cloud providers go down, the cache of information that we have available in CloudGraph is pretty up to date depending on how often you trigger it to run. It can be used as a backup in those situations where the various cloud providers go down. And you still need to get that operational information out for whatever you happen to be doing.

[00:36:08] KP: Well, you'd mention I can get on AutoCloud with a free account, which is definitely interesting to me as an indie developer. But if I'm in the chair of a enterprise technologist, I have a few more requirements, what sort of service offerings do you make available for larger companies?

[00:36:24] TK: For larger companies we have – For AutoCloud specifically. And I should mention just the distinction there. AutoCloud is the SaaS platform version of CloudGraph. We decided to make the distinction because we want CloudGraph to be forever free open source

software. We really believe in giving back to the community building out a valuable provider ecosystem and giving almost all the functionality that we can in CloudGraph available to users.

And for AutoCloud, for folks that are looking for a more managed version of the software, one that has things like built-in 3D visualizations and dynamically generated architecture diagrams for your entire system, we've kind of drawn that line and puts some features and functionality into the enterprise platform.

And from an enterprise perspective and for those larger clients, we offer a self-hosted version. We offer a managed version of CloudGraph on AutoCloud that is incredibly scalable and you can hammer pretty well and not get rejected. And we also have some interesting things coming down the pipeline for AutoCloud, like real-time streaming when it comes to being able to have your data available.

When you think about that, that's going to be a really interesting and powerful thing. Because a lot of large organizations that have to go and make calls, thousands of calls, hundreds of thousands of calls to go and get their data in a timely fashion, can't really do that currently using the cloud provider APIs.

If we can create a temporary cache of your data that's near real-time and fed in by all the relevant details and information that we can pull in from the cloud provider event buses, that kind of gets around that problem. Because we're using GraphQL, you can go and you can make, instead of a hundred different API requests, a single API request to get the data that you need to do your job.

Our system is incredibly performant. It's backed by a very powerful graph database. And it makes it seamless and easy to go and get all the data you need on your scale, not on the cloud provider scales themselves.

[00:38:19] KP: For engineers that work with the cloud who want to learn a little bit more, where can they go?

[00:38:24] TK: They can find us at either autocloud.dev. If you're looking to take a look at the AutoCloud platform, like you mentioned, it's free for developers for up to 2500 assets. You'd be surprised. That's enough for several medium to large size production environments. You can check out AutoCloud there.

And if you're looking for the free open source version that you can run locally as a command line tool or deploy to something like an EC2 instance or an Azure VM, you can find that at cloudgraph.dev. Or you can simply Google CloudGraph and check out our GitHub repository.

[00:38:57] KP: Well, Tyson, thank you so much for coming on Software Engineering Daily.

[00:39:01] TK: It's been my pleasure, Kyle. Thanks so much for having me.

[END]