EPISODE 1479

[INTRODUCTION]

[00:00:00] AD: Nearly all new tech companies build in a public cloud and established companies are rapidly migrating applications to the cloud from their on-prem data centers. But this move to the cloud can lead to a visibility problem. Cloud providers offer not only compute instances, but also manage services like databases, blob storage, queues and more. It can be difficult for SRE teams and security departments to understand what is happening across a company's cloud accounts.

Yevgeny Pats is the creator of CloudQuery, an open-source cloud asset inventory powered by SQL. CloudQuery allows you to ingest and structure the resources in your cloud accounts so that you can query them using SQL. This allows SRE teams to understand the source of specific resources, while security teams can ensure compliance with policies. In this episode, we'll discuss CloudQuery, Yevgeny's entrepreneurial background, and raising funding with an open source project.

[INTERVIEW]

[00:00:55] AD: Hey, Yevgeny, welcome to Software Engineering Daily.

[00:00:58] YP: Hey, Alex, thanks for having me. Excited to be here.

[00:01:01] AD: Absolutely. So, you're the founder and creator of CloudQuery. Can you just give us a little bit of background on what CloudQuery is and what it's doing for people?

[00:01:10] YP: Yeah, sure. We started as an open-source project early last year. We started as an open-source cloud assets inventory. So basically, connecting to all your cloud services, cloud API's, then striking all the configuration, transforming it, and loading into a relational database into Postgres. So, maybe you can also think about this as like Terraform, but the other way around, while provisioning anything, but just extracting all the configuration.

So, kind of like an ETL layer for cloud configuration and we started because we thought like, one because I didn't find anything like that and I really needed it. And two, because I thought that it's a

cornerstone to a lot of use cases, in the cloud. There was security, cost visibility, you have like a sprawl of enterprise closed-source vendors that I think their development teams, like 90% of the time are implementing closed source version of CloudQuery, and then everyone has their own small, closed source version, and there is no like, open source. So, we decided to go the open source way, and this also gives the ability to, like users to also contribute back because there are infinite number of API's out there, and there is no one vendor that can support all of that. The only way that we thought, we're able to solve it is to build the open source way and kind of like the similar reason why Terraform is open source, right? Because you want to give people the ability to contribute back something that they need, and the vendor just didn't get into, implement it yet. I can talk a bit later about some of the use cases, specifically in some of our users, and how people like use us and deploy it.

[00:02:56] AD: Awesome. I love that description of a reverse Terraform, and I think the open source nature of it is interesting. I want to get into that later, especially like as a funded company and open source how we do that. But I think you make a great point where if you're building this framework of sort of how do you sort of input resources, get them into this thing to make them queryable, then that open source nature of it just allows so many people to come and do the long tail of things, whether it's a different cloud provider, different type of resource that other people aren't using, they can they can sort of get those in there. So, I love that.

You said, as you're looking to make this, you wanted something like this, and you couldn't find anything like it. Tell us about that. Why did you want this to exist? Getting your resources into a relational database? What were you looking to do with that?

[00:03:37] YP: When the industry around like security, DevOps, cloud, last 12 years, those SaaS apps, I was also on the other way, not only from the vendor side, but also like from the builder side. It's kind of like close to me. But the other thing that the big shift that I saw at my first startup was super random. And that may be like a long answer to a short question. But my first startup was super enterprise, like it was security, top down startup, and I've been there like for two years, but then I understood that I don't want to run the company where there is get a demo button.

So, last five years, I was like focused solely on the developer space, because for me, all the classic vendors didn't make sense anymore. Because I wouldn't use it from the user perspective, right? But a lot of the companies still in the security space, are like top down. You can also go to Palo Alto

Networks, Checkpoint, but also startups, especially like in Israel. So, I said, “Okay, let's see, what's the dev alternative, open source alternative”, and I didn't find any. That’s basically, how I started – there were like a few like open source projects, trying to tackle it, but nothing like that really hit it out of the park with like good architecture and good adoption.

[00:05:00] AD: Cool. You mentioned it’s open source, you have a bunch of different providers and things. Can you give us a sense of like what it supports currently? I imagine you’re being the major cloud providers, but what other sort of providers and things? And then how deep are the resources within some of those providers?

[00:05:16] YP: Yeah, so we definitely started as like focused on the big three cloud providers like AWS, GCP, Azure, like DigitalOcean, and a few smaller providers. But yeah, the focus was really on the big cloud providers. We have also third-party providers, for example, like the Onyx Cloud is maintaining they're on integration and own provider, which is pretty neat. We’re trying to expand that, unhealthy users to do that more easily as we go, for better developer experience, but we already like saw some good third-party integrations to CloudQuery. And regarding the depths of that, yeah, we try to build pretty deep and like support as much as API's as possible, especially like an important one. I think, in AWS config, we support potentially, like even more, and the same thing in GCP, even more resources than like AWS config, or in Google Cloud asset inventory, which is like the enterprise, like the cloud solution, because actually, their team, they have the same problem. They work with the same API's and their solutions is closed source, so they hit the same. So, even though they are in Google, or in AWS, it doesn't help them to cover all their API's.

[00:06:30] AD: I think that's so interesting. It's the same pattern we've seen with Terraform, over the years, where Terraform has better coverage than CloudFormation. I think the CloudFormation team and things are getting better about that. But it's just amazing. When you have that open source nature. Someone in the community is going to want to contribute that so they can use it and it's just pretty amazing.

Okay, so imagine I'm someone that I have a lot of resources in AWS, I want to get a sense of this, I pick up CloudQuery, how do I get started? What's that process look like from getting started to getting it into a relational database? What's happening there?

[00:07:00] YP: Yeah, so there are kind of a few onboarding paths, really like the first step, usually just to Go. So, it's written at Go on single binary and kind of like have similar – actually, really similar architecture to Terraform. We use a lot of the underlying Terraform libraries, like it's called Go plugin. So, to download all the plugins and run them locally, but yeah, it's a single binary. You can download it locally and then you just run like Postgres in the Docker in a local container. So, we can load all the information in your local Postgres. You're connected to your AWS account and that's basically it. We support also AWS specific features like AWS orgs. So, you can connect us to – read all your audit permissions so we can extract all your accounts under your organization, go through them, extract all the configuration across all your accounts, and people around us on like thousands of accounts. I think the biggest one, at least on GCP that we saw from one of our users is like tens of thousands of GCP projects. The nice part is, it's really about like rebuilding those use cases on top of a data platform. So, once you have the data there in the relational like structured way, then you can connect it to whatever you use, like Grafana. Some people connect us to Grafana, to MetaBase, to Apache Superset, whatever you have, it doesn't matter, just the database.

[00:08:33] AD: Yeah, cool. Once I've got all my resources into CloudQuery, what are some of the practical use cases or maybe even teams or different things that people are doing with that data?

[00:08:44] YP: Yeah, so it's mainly split across like two teams that DevOps and SRE teams, kind of like fall under the same bucket and the second one is more on the security side. Sometimes, it's also like the DevOps team also thinks that. But on the DevOps SRE team, it's usually like just around monitoring and visibility. So, you set up alerts, for example, in Grafana, or MetaBase, sending you either like daily reports, or daily – or reports when something changed when you have like, for example, more computer that you want to or some API where you have API's enabled, that you don't want to be enabled at some project. Whatever rules, like you can set up any rules with SQL. Basically, you use SQL as your rule and query engine.

The second use case, yes, around security. So running, putting the guardrails is again, the standard query language, and we also implemented something called quality policies. So actually, just another construction and dump of SQL, so it just like you're on the same SQL, and you can run them like as a batch, you put with all your policies or rules, and notifications and so on. Another thing on the DevOps side that we saw, the use case that we saw quite a bit is not only giving this visibility to DevOps team, but also giving this visibility to the developers. So, let's say you are like thousand people in the

organization and you have thousands of accounts, not everyone has access to all the accounts, and they shouldn't. But also, for example, you're debugging something, or you're seeing something in the logs. Now, you want to understand like, okay, where this in like – in which quality it is, like this IP is located, and then like, in which project, who should they reach out to? Where it's connected. So, just kind of am infrastructure search use case, I would say for developers.

[00:10:43] AD: Nice. I love that. Again, I wanted to get just a sense of scale for queer users like how big are these projects are having? You're saying people with 10,000, GCP projects, how many resources are we talking about there? I mean, are people pushing 100,000 million? What's that sort of look like?

[00:11:00] YP: We sell, 100, 200, 300, and thousands. So, most of our early adopters that we saw, coming to CloudQuery is usually big companies that, yeah, they don't have like a good enough alternative, basically. This is also partially why we built it. But yeah, companies like Tamp, who is vastly another legal organization, using us in production, which is good. And yeah, we're working on bringing more.

[00:11:29] AD: Yeah, absolutely. I imagine just like with 10,000 projects, and trying to figure out which S3 bucket or like you're saying, which IP address and who owns that is just quite the task unless you have something like this. So, with that notion of scale in mind, I want to talk a little bit about the underlying technology. I know you mentioned Postgres, things like that. I guess, were there any particularly hard problems that you had to design for as you're going either in the fetching part, in the storage and querying? What was hard and tricky about this?

[00:11:59] YP: Yeah. So, there were like a few tricky things to solve and I think we are also still solving a lot of them and trying to build like a lot of focus on the developer experience. Because actually, we have two types of users. One is really our users or developers. People who use essentially, the phosphorus or deploys, the DevOps team, the security team, or just the developers. Two, it's actually real developers of our like SDK. So essentially, we build like an SDK where you can build your providers, and we have some users that already did it. And then it's a completely different developer, right? So, you have to make sure your API's and SDK is in a very high production rate, so people can really use it and it can scale.

The other thing is, like the developer experience and the usage that it can support like, high scale accounts. The quality, we put a lot of focus on quality over quantity. Especially, because it's a dev tool, self-serve dev tool, where we don't have any sales team to support it, right? You come to our GitHub, you try it, either it works and then you stay. On the contrary, had it not, then we don't know about it. Like some of the hard problems, yeah, as you say, is around like the fetching. So, we put a lot of work into scheduling and optimization, and how to fetch kind of the most resources and short amount of time. That was one.

The second one was really designing and building CloudQuery SDK from the developer perspective. And then putting a lot of effort also internally and also for external integration, but also internally, how do you scale that in terms of both in development velocity and testing.So, okay, we want to increase our team and we want to support more API's. I see pull requests. It's a lot of API support that we want to add, and we want to be able to add them fast, or to other people to contribute. So, I need to know that okay, it works and it's easy to write tests for that. And like testing infrastructure is quite tricky, so we have Terraforms to test it, and we tried to make really, the contributor developer also quite easy, and we're still working on making it even easier. But that was a lot of the engineering and that's still like going on and we try to put focus on.

[00:14:28] AD: Do you have a lot of integration test against live accounts with real resources? I feel like Terraform had that problem just expense wise of like creating and tearing down resources.

[00:14:38] YP: Yeah, we had. I think it's partially solved. So, the best thing that we found, like, for every resource that we create, we want to test it against live environments. So, we arrived at – the contributor has to write Terraform for that, but we deploy this resource in real environment, and then we test against that, okay, we already to fetch like all the data that we're expecting too. And actually, we are not even tearing this down, because we want people to be able to – we don't want people to wait for this resource to be created, which will be extremely lengthy and we don't want to do nightly task of you committed something in the morning, and then it broke in the night, and getting into all this flakiness, so it's just there. Yeah, apart from maybe caught – things are very expensive that we're skipping, we try to have live and test environment, and maybe asset will grow. We'll have to think about like something smarter. But for now, the cost is not that high. So, we're just trying to keep it simple and test it against real environment.

[00:15:46] AD: Yeah, I wonder if the clouds would help you out with some credits on that, too. I imagine you're helping with adoption in different ways or reducing friction. So, it'll be interesting to see. You mentioned you spend a lot of time on scheduling and efficiently pulling things like that. If I have an organization that has lots of accounts, that has 100,000 resources, how long does that CloudQuery fetch take to go and pull those up? Is that minutes? Is that hours? What's that look like?

[00:16:14] YP: Yes. So, one, it depends on the computer that you have and number of CPUs. That's the ETL workload. I think, like one of the reasons that we also chose those, that's released in terms of distributions, rather than just one binary without dependencies. I think, it was one of the main reasons. But the second one is actually, it's very good in concurrency. So, you can just like create very lightweight routines and we use that. So, I think if you have like strong enough machine, let's say even like eight CPUs, maybe we can go like 16. But you can get it maybe even under an hour. I will say that's the ballpark, if you have like really that extreme big accounts, which should work because you can even fetch it. We saw people fetch it and we also fetch it in every six hours. It's usually good enough for us. Some people do it like every two hours. We didn't see a lot of requests to have real time, completely real time. So, if you have two hours, it usually should be good enough.

[00:17:17] AD: Yup. Are most people hosting CloudQuery somewhere in the cloud, or do you a lot of people just run on their laptops and checking it there? Have you seen that?

[00:17:25] YP: Yeah, exactly. So, the first kind of onboarding guys to run locally, just to play with it, see the defaults that it answers your use cases, whether it's search or visibility or security roles. The second one is deployed on kind of production and to run periodically. You tend to run code a lot of ways. So, we just give some guidelines and pre-made deployments for Kubernetes, just to consistent and not like support a lot of different types of deployments. So, we have popular culture, which you can deploy on your Kubernetes and fool around and the cron job, and you can connect it to your RDS. But we also provide Terraforms for GCP and AWS, to provision the infrastructure is you need – if you need actually, together with the Kubernetes, as well. So, you have like one kind of like, one deployment, so you don't need to jump from Terraform and Helm, but actually, what the TerraForm is doing is deploying the Helm after it provision the EKS cluster, the database, or in GCP. It's the key in the cloud SQL.

[00:18:34] AD: Does CloudQuery support Kubernetes in the sense that I can get a more granular look at like what my Kubernetes cluster is doing? Maybe I have a hundred nodes running, but I've got all these different services on it. Can I get visibility into that?

[00:18:48] YP: Yeah, so we have a Kubernetes integration provider. I would say like data, so it support some of the resources. I'm not sure like what the granularity of the visibility yet. You can plug it in. Something that actually people ask is to connect the Kubernetes provider with the, for example, with the GKE. So, I'll throw you **[inaudible 00:19:12]** like the information from GCP and configuration, while the GKE is to go ahead and use the Kubernetes provider for the Kubernetes clusters on your cloud provider, right? Whether a GKE or AKS to get even more like granular data.

[00:19:27] AD: Yeah, cool. One last thing just on the underlying tech, I feel like – I read a blog post about you're looking at timescale and things like that. Do you have historical looks at the resources I have? Or is it always sort of like a, "Hey, this is what I currently have, and it's going to get overwritten every time when I do a fetch."

[00:19:43] YP: Yes, this one was really tricky to get right as well, and we tried different things on what will work. So, to have a historical view, you basically need to maintain like handwritten migrations for every resource, which is quite impossible from development perspective. But then we talked a little bit more with some of the users. And for them, they actually said that they don't necessarily need it all, like, all in one schema at the same table to always create the time, but it's more like for reactive or investigation purposes. Okay, I want to go back and look at what I had in like this date, like load it, and look at what I have there.

So, some of the things that will probably look into this scenario as that they said, it's actually more of a data warehouse and data lake. So, you want to actually just store it, where it's cheap, and then only query when you need it. This is something that we will look into, in terms of supporting, actually, data lakes, like Snowflake, BigQuery, and putting in there, and then you can also like both use other databases, and also use it for historical reasons. But in the sense, you can also do it outside of CloudQuery, just backup your **[inaudible 00:21:00]** periodically. But we want to give like more native support, right where you can kind of operate to a data lake, and then, okay, it's sort of there. If something happens, I have another point of information that I can go back and use if I need to investigate something.

[00:21:17] AD: Yeah, cool. It could also be cool to have like some post fetch, roll up aggregation queries, maybe just aggregate some of the data into, you're saying like, “Hey, on this date, I have these many easy two instances in each account, just maintaining that.” A smaller number of rows to maintain over time, then you could still get some historical stuff or things like that, if you wanted too.

[00:21:37] YP: Yeah, exactly. For one table, this is something you can do, like, you can maintain migration for one, exactly one table, but not for hundreds of tables. So yeah, this will be one option as well.

[00:21:47] AD: Yup, cool. That's interesting. I want to shift gears a little bit and go into your background. So, you said, you've started a few different companies, this is your third one. Can you just give us a little bit on how you got into entrepreneurship? Maybe some of the companies you had an acquisition in there, some cool stuff?

[00:22:01] YP: So, I guess I could start from the start. I guess I always slashed into – it was my kind of childhood dream starting company. I didn't know why. Maybe ran like a lot of stories about Mark Zuckerberg that it was exactly like 2008. I finished like high school and yeah, I guess, like somewhere there. So, then I joined the Israeli Cybersecurity Intelligence Unit and I've been there for four and a half years. I was into like computers early on and a lot of the – iPhone having then, so I guess it's all accumulated all together. So, it was kind of my passion real from the start. Also started like a few projects like while in the army, kind of like Bootstrap things.

But then, after the army, I joined a cert, like cybersecurity startup in 2013. It was a small server, it was called HyperWise and it was acquired by Checkpoint after eight months. There wasn't even like one customer. And I said, well, like this was my first startup experience. And I said, well, that's easy. You just start a company, you're reworking on eight months, you don't need to have any customers, and then just check **[inaudible 00:23:16]** for a bunch of money. So, I didn't stay in Checkpoint. I had a retention. But I said, like, “If that's that easy, there is no sense to for whatever.” It's money. So, basically, it was a good experience, but from the sense, actually, it was pretty bad. It was learning from completely the wrong example.

[00:23:36] AD: It taught you the wrong lesson.

[00:23:38] YP: Back then, I was sure, that's the way it is. I saw it with my own eyes, and then I thought – the funny thing that I had like two good friends that were working in small startup at the same time, and they had the same experience and just like that, there was just some like micro acquisition spree of just very small companies and we all learn the wrong lessons. So, we raised some money for like startup, initially, we didn't even know what we were doing, but was cybersecurity was hot back then. I guess, it's also hot now. We got some money, we started like, after we got the seed round of like $2 million, we started talking to customers, investors, we understood that actually, our idea is not that good. So, we pivoted pretty quickly **[inaudible 00:24:25]** security, like enterprise level security and started building that for two years and then I realized, alright, no one is coming. No one is buying us. It doesn't work that way. Actually, you have to build a product that people use.

So, actually, it was like what I thought before I wasn't the third startup. Okay, now I get it, but anyway, we are not that great, a lot of product market fee. So anyway, like investors brought new CEO. I stayed a bit like to help with attack, things like company – a solid place. And then I went, I said, “Okay, I want to focus on dev first, PLG. I think, this is the future.” We got it completely wrong here, but looks like they found like a new CEO was like sales lead growth and he landed there. The company is still alive today. But it's always early, you never know. I think it's doing a bit of revenue, which is nice.

So, like a complete disaster. But then yeah, it was totally focused on PLG and for Pike3, about three, four years, was focused on like bootstrap self-funded startups. And my last one was really, it was the CI for fast testing. So, if you know like Google has cluster fuzz, it’s like their open source kind of fuzzing as a service for a lot of open source project. I thought, okay, looks like the market is hot. I want to try it out. I bootstrapped it like with something more user friendly. They have it like – it was kind of a Google internal project. So, I wanted to build something more user-friendly. I built that and we had like, quite a good market sharer, like in terms of open source project using. It was called **[inaudible 00:26:10]**, like 50% we're – faster than 50% was Google, like, of the whole 50% of the project, that we're using fuzzing, right? That’s all of them.

But then I realized that, after the first experience, that we raise money way too early, I was always very cautious to raise money. Because after that, it's the way of no return. Okay, what happens if you realize one week after we raise money that there is no like product market fit at all? Then your stock. Then you have to like, okay, let's make something got – and you're like, “This inconvenient point.” Yeah, I was

very cautious with raising money before, I'm sure. I realized there is not a lot of like product. It's a bit of too niche, not every project needs fuzzing. It's a bit of like C++ specific. If we use the market even more, it's a lot of our projects were really like C++ modular, it's a burning problem with memory corruption vulnerabilities. So, I think people out there said that just not DOD, and C++ plus wants anymore. But if it's like, the margin is getting smaller. Actually, luckily, GitLab said, "Okay, we want this, it's good for us." I said, "Okay, good. Just in time, I'll help you integrate it into blog forum." Yeah, it was a great experience. Also, in GitLab, like looking how they work in remote company and we are a remote company. Now, I learned a lot of lessons there. After that, kind of like went into CloudQuery.

[00:27:46] AD: Did you stick around at GitLab, for a while? Or did you mostly just help integrate? And then say, "Hey, I want to start my own thing again."

[00:27:53] YP: Yeah, I helped integrate. But I was always on the – thinking about the next thing. I didn't know when I will find the next one. It was because I wasn't pressured and I want to be sure also work on the right thing. So, it kind of like grew faster than I expected. Some of my early experiment was the open source. Once I saw it, it started getting like traction. I decided to leave and started like focusing on that full time.

[00:28:21] AD: Cool. That's a good segue, because I want to talk about CloudQuery is a totally open source project, but you've also taken seed funding. So, how do you sort of plan to balance that open versus paid nature? Do you have any, I guess, like business role models in that category? Like certain companies that you want to model after there? What are you thinking about?

[00:28:38] YP: Yeah, so I think I have a few and I think also, it's something like, it's still an unsolved problem, and a lot of companies taking different approaches, depending on their specific use case. So, I think, in some sense, like Terraform is a good role model. But again, I think we – yeah, eventually it will be managed version. A lot of users ask us for manage versions, because infrastructure burden is real and a lot of companies don't have any specific constrain. It's usually a no brainer, as long as you give some competitive and reasonable pricing.

So, this will be our way into monetization to start working on a managed version of CloudQuery. But before doing that, we need to have like big enough adoption, like, let's say, in the thousands of medium

to big organization using us on a daily basis, because let's say we release a managed version, and we can take something 10% conversion rate just for – we don't have any special features, just the same thing, just manage. So, 10% conversion is, I think, something reasonable. If you have thousands to 2,000 of users, it's starting to be – all big users started to be interesting in terms of revenue. Maybe if you have even hundreds, which is not bad, 10% conversion rate, then you wasted a lot of time and money on 10 very expensive users while you could focus on growing the community, maturing the project more, and you can do both, usually, because focus is constraint, and also because money as well, like constraint.

[00:30:22] AD: Yeah. There are a lot of open source companies trying to make some money. I think some of them are going to have a tougher time. I think this one, like, you're saying, people will pay for managed version of this, given that there's a database involved. Do you want some sort of UI, maybe some access control, things like that? There's going to be a lot, I think, a lot of opportunity there to make that work.

[00:30:40] YP: Yeah. I think there will be also like, potentially, and this is something that's just too early to know, just because the things that I'll know, in a year, or things that I don't know now regarding what features we would do, maybe in the managed version, or what features will make sense to build like an extra, or how to do tiers. You have a first tier and then you have a second tier. I literally can't – it will be too hard to predict it. But yeah, eventually you will have to – even if you have like this conversion for just a managed version, you will have to start thinking, how do you increase this conversion? Maybe have you have tiers? What features you introduce to make it profitable? Again, it will depend on financial, what is the team size that you need to maintain? So, you have all those metrics to make some smart decisions.

[00:31:33] AD: Yup, absolutely. It'd be cool to see that go. I have a last section here that I want to talk about, and I came up with some potential features or areas that CloudQuery could go in. I want you to tell me like, no way, we're not going to do that, that doesn't make sense with CloudQuery, or like, this is why that won't work, things like that. So, we'll just sort of riff on these and you can expand on them if you want to.

Number one, to start off. So currently, CloudQuery supports SQL, you can do select statements, select whatever, from my EC2 instances, whatever. What about inserts or updates? Will you ever create or alter resources in CloudQuery? Is it always going to be read only?

[00:32:07] YP: Yeah, it's a great question. I think we kind of discussed it also internally quite a bit. For now, for sure, we'll focus on read only, because we still have a lot of features and a lot of growth to do with just what we have now, and a lot of incremental features, and this one is kind of also like, if you want to – you really like taking the company in another direction, it will also mean a lot of more testing and more development time. So, you want to be sure people really need it in a sense. And that's a good question.

I'm partially sure. I found one use case, which is good, but I'm not sure again how popular it is. For example, in terms of provisioning, people already use Terraform. So, I would say, it kind of feels you don't want to replace that. But on things deleting, kind of deleting resources in the sense, for example, there is a tool, even quite popular called AWS Nuke, which clears AWS environments. You actually give the right configuration and kind of a DSL. If you look at the code, it's also a bit similar to CloudQuery.

So, the next thing is okay, I can use SQL and the framework is the same and then we can just implement the delete thing. But I think you will need – if we were going to make this leave, we'll need some more insights into really how popular that is. Because it can bring a lot of like support, new bugs, support burden and development burden. It's a good question. There is actually another company, I think called like, iSQL, which is, I think, kind of trying to replace Terraform with exactly – insert into, I think it's the UI now, how it's going. But it's an interesting approach. But I'm not sure. I don't know. It's a good question.

[00:34:04] AD: Yeah, I think I agree with you. I'm a big infrastructure as code fan, and I just think like, why would you want to have a drift from your Terraform? You've got the CloudQuery drift in there, and it's just going to add more drift. So, I agree, but it's interesting. The one thing I was wondering is policy remediation stuff. If I will always want to say, on my s3 buckets, always enforced this, it'd be nice as a security team to be able to do that. But you probably just got to go out and talk to the teams responsible and say, "Hey, change your Terraform, so that we're always provisioning this property or whatever." Probably better ways to do that.

[00:34:34] YP: Yeah, exactly. Because then you also didn't get into these like loops. If you just remediated it, and this is like what we heard from the infrastructure team, yah, they are worried about, sometimes about auto remediation, because then it will just go into the loop. The only exclusion that I heard about is, if it's these one or two rules that are really bad, unless you want to repeat it right away, but usually, if it's something that just you want to find exactly like the infrastructure code and fix it there. So, it's really important.

[00:35:07] AD: Cool. Second idea. Have you thought about merging this with metrics in some way? I'm seeing – if I want to say, I have all these S3 buckets, or maybe all these SQS queues are whatever across my infrastructure, find and show me the top 10 SQS queues by the number of messages received in the last day or something like that. You probably have to push this out to like a foreign data wrapper or something and query that outside of CloudQuery, but would you ever sort of mix that in?

[00:35:34] YP: Yeah, I think it's something that we will look into, and I think it's under – it comes under, like, basically, the meta feature is connecting it with other external data. So, if it metrics, but also maybe cost, maybe security, kind of the next one, the most requested one, which also can work also in ETL approach, like metrics and like security issues. So, collect security issues from AWS, security hub, or things like whatever. Get a dependent VOD, or like any other alerts, or Snyk, plug it in, and then you can do smart things and prioritize. Usually, you have like tons of those alerts and then you can ask questions, show me like the vulnerabilities that are just on in this VPC, which is important for me.

So, some of the metrics, yeah, metrics is another place to get information. Probably it's a more tricky one, because it's not a classic ETL, because, as you said, maybe Postgres foreign data or wrapper, because there is a lot of data there. So, yeah, it's a good question. If you want to store it on daily or just query directly and have like Postgres data wrapper is actually good, but actually, a great way to just solve it.

[00:36:52] AD: Cool. What about this third idea. So currently, CloudQuery is sort of like a pull-based system. What about like a more event driven, push based system where if I create an EC2 resource, maybe you guys hook in to my cloud trail and immediately ingest that in. I imagine that takes a lot from the cloud providers to help you out there. But have looked into more event driven and push base updates?

[00:37:15] YP: Yeah. So, it's also on the need longer term, I think to investigate it and try out. But yeah, reading cloud trail or event bridge. For example, in AWS, or in Google, it's a lot other – I don't remember the name, but yeah, the alternative file trail. We're getting into that. And then basically, updating on your on push basis. It's possible, but of course, there will be like, new challenges and new development, new things, if you want to test, just for example, cloud trail also has like 30, sometimes 60 minutes delay. Sometimes it only has like partial information. So, you might want to – sometimes it will be easy just to update in place. The state of this EC2 was changed from running to stopped. So, you just go ahead and change it. But sometimes, you get only part of the information and then you might need to actually do another call to the cloud providers. Okay, "Give me all the information now. I want to update." So, a lot of room for questions and how to do it in scalable ways, and so on.

[00:38:25] AD: Yeah, sure thing. Cool. Alright, last one, and maybe it already supports this, I didn't poke around the hub too much. But what I like non-engineering resources. Maybe if I want to go to my marketing email provider and look up email, or pulling email subscribers, or maybe like replying and pulling employees, or just guests, or whatever. Do you support like things outside of cloud providers? Or is it pretty focused on engineering stuff?

[00:38:49] YP: Yeah. So right now, we don't have those providers, just because we were super late – you can do AWS and GCP and working with those use cases. But it was always on our mind to also potentially support others. The use case there is usually different. It's like more dated team marketing team, and, yes, it's a question because actually, the underlying tech and architecture is exactly that. It's just that our first use case was solid infrastructure. But yeah, we'll be looking into companies Airbyte, for example, or Fivetran, that our focus there, on the long tail of small market providers, but maybe it's something that we'll also look into, potentially.

[00:39:33] AD: Yup, cool. Well, that thanks for playing along with me there on some of those even if those weren't the best feature ideas for you. I appreciate you walking through that.

[00:39:40] YP: Yes. It's been fun, actually.

[00:39:43] AD: Cool. Yevgeny, thanks for coming on Software Engineering Daily. It's been a great discussion. Where can people go to find out more about CloudQuery? About you? Where should they be looking?

[00:39:51] YP: Yeah, so feel free to like to jump on our GitHub page. We also have a discord which is fairly active, so you can go to cloudquery.io/discord on our website on the discord link and you can reach out to me there on my Twitter or on my LinkedIn. You can find me anywhere basically, monitoring all questions. I'm the support team, as well.

[00:40:14] AD: Perfect. Sounds great. Well, Yevgeny Pats, creator and founder of CloudQuery, thanks for coming on Software Engineering Daily.

[00:40:21] YP: Awesome. Thanks Alex. Really appreciate hosting me and I really enjoyed it.

[00:40:25] AD: Sure thing.

[END]