EPISODE 1152

[INTRODUCTION]

[00:00:00] JM: Cloud resources can get out of control if proper management constraints are not put in place. Cloud Custodian enables users to be well-managed in the cloud. Cloud Custodian is a YAML DSL that allows you to easily define rules to enable a well-managed cloud infrastructure, give you security and cost optimizations.

Kapil Thangavelu works on Cloud Custodian and he joins the show to talk about modern cloud management and what he's building with Cloud Custodian.

[INTERVIEW]

[00:00:33] JM: Kapil, welcome to the show.

[00:00:35] KT: Thank you. It’s great to be here.

[00:00:36] JM: Simple question; what is a cloud resource?

[00:00:40] KT: A cloud resource is any type of thing you can provision through one of the cloud APIs that exist within the cloud control plane. So an S3 bucket, and EVS snapshot, Google Cloud function are all resources that you're creating and provisioning in your respective cloud provider using their control plane APIs.

[00:01:01] JM: Why is it difficult to manage cloud resources?

[00:01:04] KT: Because there are so many ways that people can provision them. They can create them in the console. They can create using Terraform, or Cloud Formation, or CDK, or Pulumi, or Ansible. And so there're so many different ways of provisioning and creating these resources that it creates additional – And there are so many permutations around the configuration of them that it creates a problem of how do you manage them across all the different app teams within an organization.

So individual teams might use different technologies, but at the same time, an organization wants to make sure that regardless of what tools the team used, that their overall footprint of resources is being well-managed.

[00:01:51] JM: Is management of cloud resources about cost or is it about just garbage collection policy? What exactly is it important for?

[00:02:00] KT: So I'd say an organization being well-managed in the cloud really comes at to like three broad pillars. One aspect is security. Just making sure that the resources that are being provisioned are not being shared excessively outside of an organization's boundaries. All these cloud resources are typically URL accessible just by knowing that identifiers. So making sure, like for that traditional distribution of network perimeter security really has shifted out to being the configuration of these resources, since they are effectively globally accessible through a URL.

Second bucket is things around, as you've noted, cost optimization. So finding things that might have been over-provisioned that aren't being utilized anymore and being able to detect those and potentially clean them up, resize them, turn them off at night to affect additional cost savings. And then the third tier is really around sort of operations/compliance. So making sure all of your resources are tagged correctly for cost accounting purposes, making sure that all of your resources are logging to a centralized bucket and other operational things like taking backups, etc. So really, when we look at what we're doing, it's really about helping an organization be well-managed in a cloud across all these different dimensions.

[00:03:17] JM: What is Cloud Custodian?

[00:03:19] KT: Cloud Custodian is an open source project that I started when I was at Capital One in 2015, and it was really about looking at how the organization was handling all different aspects of going to the cloud. And a lot of the things where that mapped to either controls or good governance and operations were done as one-off scripts. And as I looked forward to where the organization was at the time, at the beginning of their cloud turning to where they would be going, that there are going to be thousands of these things. And so Custodian started off as a way of making it simpler to author policies by credit using a YAML DSL, marrying it up to serverless runtimes to different cloud providers for real-time enforcement, and then being able to drive behavior change through notifications and re-mediations of things. It was open sourced in 2016. And so I've been leading in the development and community ever sense. And it's really targeted to be a one-stop shop for organization to find a tool that can address across all these different needs to being well-managed in the cloud.

[00:04:30] JM: Give a few examples of how Cloud Custodian is useful.

[00:04:33] KT: So a couple of use cases for Cloud Custodian. I think making sure that all of your assets have proper encryption at rest across all their storages. Making sure that all of your resources, which potentially have embedded access control policies are not granting access outside of your org. Setting up developed off hours for turning things off at night and development environments across databases, servers, compute, resources. All of the sort of use cases that are around us, three pillars; security, cost optimization and governance operations are really things that people can express for Custodian.

Custodian itself supports hundreds of different resources. A given policy is a set of filters and a set of actions to find the interesting set of cloud resources and then take some action, a set of actions on them. And people have expressed – It's possible to express millions of different policies with Custodian. And so we've seen that across potentially setting up DDoS protection around all of your public endpoints. Making sure all of your publican endpoints were logging during network security. And so verifying that you're using TLS 1.2 or 1.3 encryption across all of your internal endpoints.

The laundry list of things that people do is quite large. And generally speaking, we see that organizations that adopt custodian does continue to have more and more policies to the tens of thousands of policies across different things that they need to do.

[00:06:06] JM: Describe the usage of Cloud Custodian in more detail.

[00:06:09] KT: So Cloud Custodian itself, it's a one line install. It's a CLI tool. We very much try to promote the use of it in a GitOps fashion, where you author policies as code in YAML form. You version control them. You do CI on them during dry runs and schema validations and code reviews, and then you run the tool itself.

So Custodian, your write up a YAML file with policies that are contained within it. Each policy targets particular resource type, say S3 buckets. It comes with an execution mode, which allows you to say either that you want to look at everything that you have out there or potentially want to do an execution mode where you're actually looking at – You're executing a policy in response to a given event.

And Custodian itself supports dozens of different event modes across all three major public cloud providers; AWS, Azure, GCP. So an event mode might be whenever I create a bucket or modify a bucket in this policy, when you run the CLI, the CLI would actually provision the appropriate serverless stack on the different cloud providers such that as those event API calls are happening, that the policy itself is going to be invoked. And so it's provisioning itself into the serverless runtimes with the different providers and hooking up the event streams for you.

[00:07:31] JM: Tell me more about what is coming across these event streams.

[00:07:35] KT: So typically what Custodian is doing is enabling you to look at the API calls that are happening in the cloud control plane. So all the different providers have APIs, and those APIs all have some form of an audit log. So Custodian is basically saying, allowing you to subscribe to those different API calls. And as they're happening, be able to invoke a policy to verify that what's being created or modified is still compliant to what the organization's goals are. And GCP that we use, this is all happening fairly effectively in real0time with a latency of a few seconds in terms of after-the-fact of the API call.

[00:08:15] JM: So what are the advantages of using Cloud Custodian?

[00:08:18] KT: So there's a number of advantages. I think for a lot of organizations adopting the cloud ends up being a longer process, because they also have to deal with the risk and compliance and security and optimization aspects of it. And Custodian gives you a tool that allows you to sort of govern at scale, at speed and doing it in a real-time way. What was first – Building Custodian out, we noticed that different parts of the org would have different tools, and they ended up becoming like a big wall of red. So being to actually do remediation and enforcement and being able to drive the behavior change the developers by letting them know in real-time, like sending them a Slack message if they launch a date based on the Internet, but as well as removing and fixing the problem as well. So teaching the users within the organization how they can do things per organization policy in the future, but also fixing the problem immediately. And it ends up being like the set of things that we, that Custodian addresses end up being sort of a set of universal needs that I've seen across thousands of organizations that are using Custodian today.

[00:09:30] JM: Tell me more about the engineering problems of building Cloud Custodian.

[00:09:33] KT: So one interesting aspect has been doing testing with Custodian and testing the codebase. Custodian supports hundreds of resources, hundreds of filters and hundreds of actions, and you can sort of combine those like Lego bricks. We're looking at how do we make sure that we're actually – And we're testing a lot of the surface area, exercising a lot of the surface API area of the cloud providers. So one of the questions was how do we ensure that we are doing great testing around these things? Because people are using these in mission-critical production environments.

And so we settled on a technique called flight recording where we're actually just writing to disk all the HTTP responses from the cloud providers, and that's been really useful in a couple different context. One is that we're able to – Where the test itself has encoded its context and setup, we're actually to take those unit tests and actually run them as functional tests, where we'll provision noncompliant infrastructure, run the policy to verify its behavior. And we can run those as sort of offline unit tests as part of our CI. And we're up to I think 3000+ unit tests around the different cloud providers. And bringing that same semantic across all the different providers; GCP, Azure, AWS, so that we have good in-depth coverage across the surface space. We looked at doing tools, mocking and stubbing, but unfortunately they weren't high-fidelity enough to all the different corner cases around some of the API areas.

We're actually recently been doing some work here to actually expose some of this capability to end users to be able to author tests for their policies where they can stand up noncompliant infrastructure using a tool like Terraform and then validate their policy actually detected it and took appropriate action successfully.

Another interesting challenge has been Custodian has – I think we're up to close to 300 contributors. So a lot of the codebase has evolved and we've relied heavily on sort of having fairly good test coverage to allow us to re-factor cleanly and freely. So many different contributors with so many different styles. We've relied heavily on [inaudible 00:11:43] and various CI tools to help sort of bring a uniformity to the codebase.

I think some of the tools in Python exists for sort of automatically formatting. Facebook has got a tool called Black. Obviously, in other languages like Go, that's built-in. It would be nice to sort of do. And we've got some additional re-factorings that we wanted to do. Originally, Custodian was just run in AWS. But as we've gotten more providers, we actually want to re-factor out a clean core so that all the providers are installed will independently. Currently, the AWS one is sort of always available by default.

So a lot of it has just been figuring out how to sort of scale the community. A lot of our users are in enterprise, it's actually been interesting looking at how many users create a GitHub account just to interact with Custodian project, which I've always found interesting.

And so some of it is also about educating our user population on sort out what is good etiquette with regards to open source, and then of course trying to encourage contributions.

[00:12:40] JM: How does the DSL in Cloud Custodian work?

[00:12:44] KT: So internally, the Custodian codebase, we actually have sort of a plug-in architecture around resources and the various filters and providers. That class model for a given resource actually is used. We specify in sort of metadata on those classes what the expression capabilities are. How that class element could be used, justified in a policy. And we are doing that through JSON schema fragments. At runtime we'll actually generate a JSON schema that's used to validate the YAML file based on what the actual resource types that are loaded.

Originally, we actually generated out for everything we supported. But we've gone to more of a lazy loading technique where we dynamically generate the schema on the fly based on what's actually in use. That's also been part of our engineering efforts around cold start latency in serverless environments to use lazy loading throughout our runtime.

So the actual YAML policies are effectively an array of policies. Each of the policies specifies a resource type, an execution mode. One of those event modes filters and actions, and those filters and actions, executions, resource types are all sort of being sourced dynamically from these different registries. And then as we got actually execute the file, we'll do an automatic run validation with JSON schema. We can also do a dry run to sort of see what resources the policy were defined without actually taking actions on them.

[00:14:18] JM: What's the deployment model for Cloud Custodian?

[00:14:21] KT: So Custodian came out of a large enterprise, and I've been in the open source community for 20 years. And what I was trying to avoid was sort of a Conway's law of the project itself being more reflective of the organization that originated it versus its intended usage. And so Custodian takes a very un-opinionated view with regards to how people want to deploy it. There are users that use it on their laptop as just an interactive query tool for the cloud. There are users that deploy it in GitHub and GitLab CI, or Jenkins, or Kubernetes. And we're pretty un-opinionated about how a user should deploy it, because if we look at – We recommend to them they treat it as part of their CI process, that they have a policy repo, that they treat policies as code and employ some form of GitOps mentality to deploying policies and executing them.

But CI tools vary widely across organizations and even within an organization. So to avoid being too overly coupled to that, we distribute several tools that can be used in different CI runners that will do automatic analysis of the policy repo to generate up, and then we'll diff policies and be able to do a dry run on that subset and being able to validate across the whole thing.

In some cases we see deployment models that are using centralized accounts to sort of manage across a set of all of the other accounts of an organization. And I'm using account interchangeably for a GCP project and an Azure subscription. In some cases we see them deploy it as sort of leaf node deployed separately within each of the accounts. And we really are un-opinionated about which way an organization chooses to do that, because there are tradeoffs to both. There's blast radius or considerations around centralized. And then there's sort of infrastructure management considerations around going decentralized.

To point it in a different way, like Custodian is deploying sort of its own cloud infrastructure assets, Lambda functions, Google Hub functions, Azure serverless functions to subscribe to those event streams. And those can be managed from a deep provision perspective through a traditional toolset we deploy. In some cases, we found that users want to use a more standardized provisioning tool that they may already be familiar with. And so we have some basics support for sharing out like cloud formation templates from policies. But generally speaking, we find the vast majority of our users use the built-in provisioning capabilities of the Custodian CLI.

[00:16:55] JM: Does Cloud Custodian vary in its coverage based on the cloud provided that you're using?

[00:17:02] KT: Definitely. We've had a lot of great contributions from different organizations including folks from the GCP, AWS and Azure cloud providers have all had contributions into Custodian. But I would say right now we have significant portion of our user population is definitely on AWS. We have a significant number of users on Azure from like – And I think probably at GCP, it probably has the fewest users. Although we have fairly good coverage I think through most of those things.

Part of the challenge with keeping up with the capabilities is also keeping up with all the new features that are coming out in the cloud. So it requires sort of constant iterations and tracking to keep up with all the new features that the providers themselves are producing. From a coverage perspective, I probably say I would be hesitant to try to spitball a number. But I'd say, by far, AWS, I think we have the strongest coverage. I think Azure is probably second. Which used to be third as far as – I'm talking about coverage. I'm talking about addressing every single capability of that provider with a coverage goal of around 90%.

[00:18:06] JM: Are there any elements of cloud management that are particularly hard to implement through Cloud Custodian?

[00:18:13] KT: I'm not really – None of the common ones. I can't really think of too many. The ones that I think end up being – There are some that are more interesting, but they tended to be more data analysis problems, like doing analysis of all of your CloudTrail logs, do automatic IM notification on your roles to create at least privilege things, and/or going through your flow logs to understand the connectivity models. So they tend to be more data-oriented problems of crunching through the datasets to find answers.

With regards to the general cloud management capabilities and protecting resources, I think we have fairly strong coverage. I think there’s been a lot of industry shifting though to moving to more of a shift left to DevSecOps pipeline where a lot of the evaluation of whether or not a resource is compliant or not is actually moving to actually happen within the CI pipeline directly against the infrastructure as code assets that might be in Terraform or cloud formation, or Azure research templates, for example.

So part of what we’re doing now in the open source is actually building out, trying to build out additional capabilities for Custodian policies to be able to evaluate as more of a static analysis of those types of assets directly within the pipelines themselves.

[00:19:34] JM: How does Cloud Custodian help with compliance?

[00:19:38] KT: So a lot of compliance is, is providing evidence that you've done something, that you’ve looked for the thing that you’re trying to be compliant to. And then you’re providing evidence that you’ve been looking for it sort of continuously. And this is what you found or didn't find. And so Custodian has this notion of very rich outputs, metrics outputs, distributed tracing outputs, structured records into a blob store. So, typically an organization will set up a policy that is running. They use the policy repo and Git ends up being an audit log of the policies themselves that are being – And when they came into effect in a given environment.

And then the execution outputs from a policy are typically directed to a blog store, like S3, or Google buckets that GCS storage. And there, those things then form sort of an audit record of evidence for a given account tenant boundary for this policy and this resource type. This is what was found at a given point in time.

[00:20:41] JM: Tell me more about the usage of filters in Cloud Custodian.

[00:20:45] KT: Sure. So our hooking language is pretty rich. So we have a default value filter, which although it's generally designed to do any sort of attribute matches on a resource. And it's using a language called JMESPath, which was originally came from sort of the AWS CLI has this - - query option. And it exists at jmespath.org. It's sort of a loose standard, more de facto implementation. It's been picked up by the Azure CLI and some other tools. Want to give it as a library-based JQ, so that he could now sort of point it nested data structures within the JSON or a JSON documented and use some arbitrary querying to get their keys and values. And so our default value filter is designed to dig in to any cloud resources and its JSON description down any nested level. And then we use that for a whole host of things from finding out when your certificate is going to expire. To making sure that your access keys are being rotated just by passing it different types of values and doing conversions.

In addition to that, we do a lot of sort of related filters. Most of these resources exist sort of within a graph, a compute node that's attached to a network and it has this particular service account or IM access role. And those have access to particular sets of resources. So we use our ability to query the cloud control plane and having this resource coverage to allow you to do related resource filter. So being able to verify that you're using the appropriate encryption key for this particular storage bucket.

And so those sort of one-hop relationships are something that we also directly express as filters. And you can first do, as an example, making sure that the subnet that you're deploying a compute instance to has an appropriate tag that denotes that it's private or that it's for this application. And so being able to do that secondary attribute evaluation on a related resource ends up being a way of expressing a lot of common policy uses.

And then around all these individual types of filters, these value filters, these related filters and lots of ad hoc ones that we produce to purpose our use case is the notion that we can combine these in arbitrary orders and to not – And with nesting. So that you're able to simply express additional policies. We've actually been actually doing work on actually adapting a simple expression language. I've always been sensitive to a policy should be easy to author and easy to read. So trying to avoid sort of, I guess, politely, a YAML vomit. And so how do we make things very concise and succinct to express?

And we've been looking at and used de facto standard from Google called common expression language, and we've had one of our contributors from Capital One has written a self-pipeline implementation, which expresses the self – Expression attribute language directly within – Available in Python, which Custodian itself has written in. And that we are looking at now being able to expose as a new type of filter one where you're effectively during multi-attribute matching within a single expression line.

[00:23:57] JM: You have a business around Cloud Custodian. Is that correct?

[00:24:00] KT: That's correct. So earlier this year – I've been leading the Custodian project and community for the last four years. And earlier this year I decided that one of the best ways to accelerate the growth and adaption of Custodian would be to actually start up a company around it. And so me and my cofounder, Travis, have started up a company called Stacklet, which is really designed to help organizations be well-managed in the cloud and by building on top of Custodian to deliver that out of the box value and scale for organizations that want to adapt Custodian for governance and security and management purposes.

In addition to that, we've also been working with some of our partners in the community, Capital One, Microsoft, Amazon to actually move the Custodian project itself into a foundation. So it is now the Cloud Native Compute Foundation as a sandbox project.

[00:24:55] JM: And what's your vision for the company?

[00:24:57] KT: Our vision for the company is to continue to lead the development of Custodian to continue to add features to it, but also to build out the tooling that organizations need to be able to successfully adapt Custodian at scale. And so, like I said, Custodian itself is very un-opinionated about how to deploy things, and that's partly because different organizations have different needs. But as we go to build out product at Stacklet, what we're trying to do is give users an out-of-the-box experience of managing policies in GitOps and executing across lots of accounts and hierarchies of mapping of those policies to those accounts and having sort of that out of the box experience. Having UI, UX and being able to manage and report on those things as well.

Whereas with Custodian itself, like the philosophy is, is it's going to generate the structured outputs and it will drop them into one of the sec driver metrics or cloud watch metrics. And the best dashboard there in that context is the one that you're already using and you're simply pulling the data from it. Whereas, something like Stacklet platform, what we're trying to do is actually deliver that value to you directly without you having to configure or to aggregate it into your existing foreign capabilities.

[00:26:13] JM: So let's say I have a data science heavy stack and I'm spinning up tons and tons of resources to do intensive machine learning jobs. Would Cloud Custodian help me in that regard?

[00:26:24] KT: So what Custodian's intended to do is make sure that those stacks that you're creating are configured appropriately for the organization's guidelines. So if Custodian itself is not a provisioning tool, it's there to make sure that – If the organization wants to make sure all the data's encryption is at rest, that that's case. It wants to make sure that on that data that you're doing in that data science stack is being made accessible outside of the organization's boundary. It can help ensure that if you leave that stack up accidentally, that it gets shut down. And so it's not about helping an application team provision. It's about helping the organization ensure compliance regardless of those application team choices.

[00:27:06] JM: Describe the onboarding experience for somebody using Cloud Custodian.

[00:27:10] KT: So right now, like it's a one line install. So you would download it. You would look for policies. Custodian doesn't currently adjust the roles engine itself. We have example policies. There're community repos of various policies. So typically your first step is either using one of those as a starting point or authoring one from scratch. You then need to set up a cloud credential against the different policy that's appropriate for your policy. That gives you HTTP access and the right level of permissions. And then you just run the CLI. So you download, you create a policy, a text setter, and then you run the CLI.

[00:27:49] JM: And then once I start running it, how could I explore it? What might I want to do with Cloud Custodian?

[00:27:54] KT: So we have interactive built-in command line help that shows sort of all the different resources and filters and actions along with examples, and that's sort of build-in and also forms the basis for our reference docs. And we've got a few dozen examples for each of the providers on things you can do. I guess part of the questions is really focused on what the use cases an organization has or what the user has for what they're looking to do. I definitely use Custodian in larger environments. It's just sort of ad hoc query of just trying to understand like who is using this particular army or being able to use it as sort of a Swiss Army knife with regards to answering ad hoc questions. But in terms of what you do with the next, a lot of it's based on what your use cases are. Some organizations come and they start off with just basic tag management or on resource. Being able to auto tag things as they're created to know who actually made them. And then they move on to potentially doing some basic security checks, doing some cost optimization. Then where people take things is really open-ended based on where they want to go. And so we have capabilities around lots of different things, but we don't really prescribe to users what they should be doing. They typically already know. Or if they don't know, then they're on the path of learning.

A great example is maybe tag policy, because it's one of the things that every organization has a tech policy, but no two are the same. And so there's a lot of differentiation that happens between individual orgs in terms of what they actually want to do or enforce. And so part of that is knowing what those are to be able to express them.

[00:29:35] JM: And how might I test my policies?

[00:29:38] KT: So right now with testing a policy, you typically stand up some component of the infrastructure, the cloud resources that you're trying to write your policy for, and then run the policy and bound the resource that it took to write action on that resource and everything worked as expected. So that's sort of the basic way that people have written policies to-date.

As I alluded to you earlier, one of the things that we're trying to do right now is actually provide a test harness framework around actually running policy test with a CLI that will actually stand up that noncompliant or that infrastructure you're trying to find using a standardized [inaudible 00:30:17] tools like Terraform.

[00:30:19] JM: Can you tell me that a more esoteric use case for Cloud Custodian?

[00:30:25] KT: Ooh! There are so many. We have a chat channel with about 1,300 users, and I think I see something new or different just about every other day. So as an example, somebody that wants to delete all of their resources, but only during the holiday break, between Christmas and New Year. So basically, nuke-out the dev environments during the holidays. And so Custodian has this capability called conditions, which allow you to express policies that you've committed and merged, but like say the change order for them, but to say they're not going to be in effect for some time in the future. And then we support different capabilities around deletion and modification of resources.

That's a good question. I honestly don't know that I've seen what is particularly odd or weird. I generally find it when talking to users and they provide sample policies. But I still learn everyday people doing new and interesting things to Custodian that I never would've anticipated.

[00:31:27] JM: So once I have attached policies to different resources, what might I do with those policies and resources? Am I querying them? Am I evaluating them? Am I triggering things based off of them? Tell me more about that.

[00:31:45] KT: So when you write a policy for a given resource, depending on how you're running it. So some people use – Our default mode is something called a pull mode, where you're just going to – Effectively, you set up a timer, a Cron job that runs that policy at whatever interval you feel is appropriate, and it's going to go and look at everything in the fleet. So all of the instances of that particular resource type.

When you get to an event-based mode, effectively you're only looking at changes to your fleet or things that are coming up that are new. And so they tend to be useful in different contexts. In many cases, organizations, they're not starting in a green field. They're starting with an existing environment. And so they may say, "Let's give 30 days grace to everyone that's in this existing environment with regards to, say, moving to a new TLS standard version."

And so they will give a grace period for the things that are already existing based on a date filter around their creation. They will then create an event-based policy that will sort do that enforcement for all the net new things that are coming into the environment. And so you can also do this notion of sort of chaining policies together to create simple workflows. So being able to say, as a human expression of a policy, being able to say, "Anything that's not tagged correctly, send the creator an email. And after one-week, go ahead and stop that resource and send another email. And after another week, go ahead and delete the resource."

And so you can chain these policies together using sort of what we call "mark for up" as an action and as a filter, which effectively just uses a tag message on the resource to say that in a week we will take a particular action. And so you can get very rich capabilities and sort of human semantics around policies by sort of chaining these policies together.

[00:33:41] JM: Tell me that some of the more advanced usage of Cloud Custodian.

[00:33:46] KT: So the advanced usages of Custodian tends to be organizations that have sort of thousands of policies that are managing across thousands of accounts and then using some of our advanced sort of output capabilities. So Custodian itself has capabilities around deploying and different execution modes like AWS config, or subscribe to guard duty events hooked up to Google Cloud Security Command Center. But the interesting part of this is being able to get a common set of uploads across all these policies. We'll do deep diving into X-ray integration. So we'll actually do distribute trace outputs around the policy execution itself, all API calls that are happening and sort of mapped-out to the policy hierarchy.

But for most users, the advance usage then typically is about what they're doing in policies. It's not really what we're doing in tools within Custodian itself. It's what they're able to express and how they are able to manage that at their organization's need and scale. And so the most advanced users end up being those with the number of policies, because they continually use Custodian to enterprise-wide scale. It's an open source product with bottoms up growth, but it's different than a lot of projects, because particularly it gets deployed across an organization's entire cloud footprint.

And so those typically have very different needs with regards to those different application teams or lines of business within an organization and which policies they want to have enforced and which environments. But from an advanced perspective, it might be adhering to various standards with regard to like GDPR, PCI and it might going through on advanced event-based policies. And so as a primary maintainer for Custodian, I don't spend a lot of time sort of offering policies for users. I really try to focus on making it possible for them to express those themselves.

So in some cases, I may not be aware of sort of all the things that people are doing today with Custodian. As I said, I'm surprised and find new things people are doing with Custodian every day. So, couldn't per se answer to some of the advanced use cases that people are already doing. Or I can definitely talk to some of our integrations. And from an integration perspective, Custodian takes potentially a different view with regards to what the cloud providers themselves are doing. So we recognize that the cloud providers are naturally going to continue to beyond just their existing breadth of resources and features. They're also going to add capabilities directly related to addressing some of the same problem domains that Custodian does.

And so Custodian takes a philosophy of being the easiest way to use those new capability sets as they come out. So be it, you know, you can take a Custodian policy and deploy it as custom config rule and just – But you get a 10X line of reduction with regards to both provisioning and code. You can use it with AWS Security Hub. You can use it with GCP Cloud Security Command Center. We're actually looking at some capabilities around sort of taking a Custodian policy and being able to transform it into some of the Azure policy language. And so a lot of what we're trying to do is just make it easier for users to take advantage of all these cloud features and to enable organizations to be more productive in the cloud.

[00:37:07] JM: Well, Kapil, is there anything else you'd like to add about Cloud Custodian?

[00:37:11] KT: Yeah. I think I'm really excited about some of our movement into being able to evaluate policies against resources earlier in the pipeline and to continue to work on some of our Kubernetes support as sort of a fourth provider, and was well growing community as we go on our CNCF journey and looking forward to growing the contributor population of our in the community.

[00:37:34] JM: Well, Kapil, thanks for coming on the show. It's been great talking to you.

[00:37:36] KT: Thanks.

[END]