**EPISODE 909**

[INTRODUCTION]

**[00:00:00] JM**: A new employee at a software company needs access to a variety of tools. In order to get started working, the employee might need Slack, email, Google Docs and Amazon Web Services, and all of these require an account with a username and password. Setting up all of these accounts can be time consuming because the company needs to go into their admin portals and create the accounts, and then the accounts needs to have the right security policies and configuration settings. And when the employee leaves the company, all of these accounts needs to be shut down.

Okta is a company that builds identity and access management software, such as a single sign on tool that allows users to log in to all of these different types of accounts using only an Okta login. Okta was started in 2009 and has grown steadily since then, going public in 2017.

Hector Aguilar is the president of technology at Okta, and he joins the show to talk about the software stack of Okta and how the company has evolved overtime as it has become a core infrastructure provider and hired a large engineering team, and scaled, and built additional products. It's an interesting story about a company that has grown and built new products and scaled its engineering workforce. I hope you enjoy it.

[SPONSOR MESSAGE]

**[00:01:27] JM**: LogDNA allows you to collects logs from your entire Kubernetes cluster in a minute with two kubectl commands. Whether you're running 100 or 100,000 containers, you can effortlessly aggregate, and parse, and search, and your monitor your logs across all nodes and pods in a centralized log management tool. Each log is tagged with a pod name, and a container name, and a container ID, and a namespace, and a node.

LogDNA is logging that helps with your Kubernetes clusters. There are dozens of other integrations with major language libraries, and AWS, and Heroku, and FluentD, and more.

Logging on Kubernetes can be difficult, but LogDNA simplifies the logging process of Kubernetes clusters.

Give it a try today with a 14-day trial. There's no commitment. There's no credit card required. You can go to softwareengineeringdaily.com/logdna to give it a shot and get a free t-shirt. That's softwareengineeringdaily.com/logdna.

Thank you to LogDNA for being a sponsor of Software Engineering Daily.

[INTERVIEW]

**[00:02:48] JM**: Hector Aguilar, welcome to Software Engineering Daily.

**[00:02:51] HA**: Thank you. Thanks for having me.

**[00:02:53] JM**: It's great to have you. You joined Okta in 2012, and we're going to talk about your journey and the Okta product and how Okta has evolved. Describe what the product did when you join. What did Okta do?

**[00:03:08] HA**: Well, I think initially Okta was focused on basically single sign on. The idea that you would need to have credentials to multiple cloud services that you need to get your work done, and Okta started as a product that would allow you to have a single credential for everything that you would use to get your work done.

So if we go back 8 years ago, there were a lot of cloud services, certainly less than now, but there were still a lot of cloud services already. You would have your Salesforce, your Box, your Dropbox and anything that you would use as a cloud service, and the idea would be to have a single sign on for everything that you had. That was a longtime ago.

**[00:03:54] JM**: Why is that kind of identity management useful? Why do we need a product for that?

**[00:04:01] HA**: Well, I think there are several issues with identity. The first one is really security, right? What was happening back then is that organizations were provisioning all of these accounts to these cloud services, and then you would have to set up a password for each of those cloud services. All of those cloud services would have different password restrictions, different security settings. Then at the end of the day, the user would have to remember all of those passwords. So what ends up happening is that you end up writing all of your passwords in like post-it note, which is not great. In addition to that, if one of those services gets compromised, then you're using the same password for all of those services. Obviously, all of your services get compromised as well.

So the idea of having kind of single sign on protocols with strong authentication so that you can sign in once, get kind of strong authentication with multi-factor, and then you basically get access to everything that you need, well, I think continuous to be very appealing to our customers. So that's kind of one thing from the security perspective.

The second aspect is the provisioning of all of these accounts to all of those services used to be manual as well. You would have to go to Box and provision a user and then go to Salesforce and provision a user. And Okta also allows you to actually provision and de-provision those accounts. That means that when an employee joins one of Okta customers, they automatically get provision to all of the services all over the services that they're going to need to do their work kind of day one. That's basically one of the nice things about Okta that your day one experience becomes really great.

In addition to that, when somebody leaves the company or when you have to de-provision an employee, we also on to provision the account or suspend the accounts on all of those services, which means that because that's done automatically as well, you don't have the concern of users having access to your company data after they leave the company, which was also again one of the problems that we had as an industry a long time ago.

**[00:06:04] JM**: So there's a few types of products that I'd like to contrast with Okta just to help listeners who are unfamiliar with this kind of product. So I run a small company, for example. We're based around Google Apps, and Google Apps is my notion of identity, and it works for us pretty well, because we only have a couple of employees. We don't have a large scale employee base. We don't have a bunch of new accounts that need to be provisioned on a regular basis. How does Okta differ from that kind of experience, the experience of the small business owner who just uses Google Apps for their identity?

**[00:06:45] HA**: Well, I mean, everything that I have described so far is the perk that we have 8 years ago. The perk has evolved quite a bit. So the problems that I was talking about is basically the way the company started. Really looking at the exact problem that you're talking about. When you have a small kind of business that has, let's say, Google as their email and maybe their document repository. That's basically how they start using Okta. But then I'm sure you're using GitHub and I'm sure you're probably going to start using Jira for your backtracking.

**[00:07:22] JM**: Definitely GitHub.

**[00:07:23] HA**: Right. And maybe you'll use Confluence, or maybe you'll use some sort of wiki pages, or maybe you'll use Amazon Web Services to provision your infrastructure, and maybe you'll have some monitoring systems like maybe Splunk, maybe Zumalogic, or maybe you'll use Wavefront, or Zabbix.

So at the end of the day, it's not only what you use for collaboration. It's also anything that you use to really run your product. I'm only talking about engineering, but eventually you're going to be using QuickBooks. You're going to be using NetSuite. You're going to be using Salesforce as well.

So all of these services is exactly what Okta can do for small business. Again, that's how we started. So now you have this concept of an identity for your company and any employee can get provision to all of those services, and if they are in engineering, they will get basically access to GitHub, and maybe Confluence, and maybe Jira, and maybe whatever you use for backtracking. Whatever you use your roadmap planning. Then anybody that joins finance will also, in addition to their Google Account, they'll get their Salesforce account, or maybe the NetSuite account. So at the end of the day, you really create a central view for the identity in your company and how that identity is used for all of the services that you use.

That's basically when companies are small, right? As the companies continue to grow, you're going to continue to need access to most services, and that's basically how Okta continued to evolve into really becoming a platform of identity for everything that you use. Because it's not only the cloud services that you use. Maybe you'll have some internal systems. Maybe you'll develop your own provisioning system for accounts in your own services, for example. Well, Okta can connect to it.

Or in the more kind of newer scenarios, maybe if you're a small company that doesn't even want to build that identity for your own product, then you use the Okta platform product, which is basically the second part of our business that we also have, where we can actually manage the identity for your product as well.

So there are customers that have gone beyond what we call the workforce, which is people that are using cloud services to actually embedding Okta into their service. Because of course we know our entities. So we have APIs. So you can actually directly call our APIs to actually power your identity for whatever product or service you're developing. So we have the developer side and the IT side if you will.

So you have everything that I have been just talking about is kind of what the CIO would be interested in, or the director of IT would be interested in. But we also have kind of what the CTO would be interested in, which is really using Okta APIs to use as a platform to power identity for service. Then they don't have to worry about not only using passwords, but also they don't have to worry about multifactor authentication, supporting kind of multiple standards, face ID, or thumb authentication. All of that can be done via just Okta APIs.

**[00:10:19] JM**: So the first engineering challenge that stands out for a product like Okta is that you essentially have the security properties of a password manager, and that means that you could – In a worst case scenario, you could have this honeypot of password information that if it was exposed, it would really create a significant security problem for a customer. So security is something that's been paramount to the product from day one. Can you describe how security management factors into the engineering process at Okta?

**[00:11:02] HA**: Yup. Absolutely. So securities is embedded in our DNA, in our culture, in our engineering organization of course, and I've been in security of a number of years to know that security is not a digital thing. It's not you're secured or not. It's really a continuum of things that you need to do. I think part of it is just embedding it really in the culture and constantly evolving it.

So from the way we train our engineers in kind of common security gadgets in code or concept, the way we have our security team constantly pentesting and constantly doing source code reviews and constantly hiring even other firms to constantly look at the security of Okta now. Again, it's a continuum of things that spans across several aspects of security.

Now, if you look at the protection of private material, if you will, that also has evolved overtime. Now, we use KMS services. We actually use AWS KMS service to all of the private material, and we also have kind of segmentation of keys across our customers for obvious reasons. So there are a number of things that we can do.

One thing that I think is important to note is you talked about kind of usernames and passwords. Okta is much, much, much more than that, because we also do multifactor authentication. So it's not only kind of your username and password. It's also all the private material that you're going to be using to authenticate to Okta.

For example, when you log in to Okta, our customers can also configure multifactor authentication, which means that, for example, you could get a push notification to your watch or to your phone after you type in your credentials to verify who you are. Or you can use WebAuthn, for example, and if you're in a Microsoft surface, you can authenticate with your face.

So there are a number of things that go into strongly authenticated user. And all of that private material, we have to protect, and there are kind of different ways that we do that. In addition to that, we have a ton of data that we can use to do some analytics just because of kind of the top or service that we are and kind of the scale that we have gotten to. We have a lot of data and we can also identity signals to protect – Basically, we look at what people are doing. And because of that, we can identity patterns and we can identity signals that can help us also strongly authenticate our users.

**[00:13:19] JM**: Okta was started in 2009. This was the earlier days of Amazon Web Services gaining traction. Was Okta built in the cloud from day one?

**[00:13:33] HA**: Yeah. I mean, our founders are from Salesforce. Our CEO used to run engineering for Salesforce in the very early days. So from the first line of the code, Okta was designed to be a cloud service. We'd never had an on-premise server till this day. Everything is cloud. We decided to go on AWS, but also you need to think about the fact that this was AWS 10 years ago. So a lot of the infrastructure and services that AWS has, they didn't exist back then. But they had enough that we could really build our infrastructure in AWS. So everything is cloud from the beginning, multitenant from the beginning, scalable in a cloud way, and from the beginning basically designed to be a service.

**[00:14:20] JM**: Describe the software architecture as it stands today. Obviously, the infrastructure options on AWS have improved a lot. You've also got a whole variety of other cloud providers you could be using. You've got a variety of virtualization and containerization technologies you could be using. What is the software stack look like today?

**[00:14:42] HA**: Well, our software stack also has evolved. If you think about 10 years ago, I'm not even sure that containers existed. Certainly, Docker was not what it is right now. So our architecture has evolved. But if we look at it, it's a traditional enterprise stack, where we have basically multiple tiers of stateless servers backed by a database and a bunch of caching engines and backend jobs.

So we're built in Java. I would say 95% of the product is built in Java, and we're backed by MySQL databases, many of them. Then we use Redis for our caching and also RPC. So in a high-level, as we started learning about new services of AWS, for example, KMS, we're really looking to what other services that we could leverage. But we're also very careful about not just chasing the latest shiny thing, because a big thing for us is reliability.

From the very early, we  understood that we were building an infrastructure service, and a lot of our customers kind of think about us as power as internet. If you don't have the Okta, the business cannot continue, cannot work, especially the customers that are using us to back their service. I mean, identity and reliabilities is incredibly paramount for them.

So because of that, we cannot put our architecture at risk. So what we do is that anytime there's a new service that we want to use, we typically create an abstraction before even start to trying it out so that we could always kind of have – Ensure that there's no single point of failure in the architecture. So I would say that the way the architecture has evolved has been mostly around reliability and scalability in the sense that we need to understand the failure modes for all of our components at different levels of stack and make sure that the system is able to work around those failure modes and react very, very quickly and hopefully at computer speed, and not human speed.

As you know, I mean these cloud services and infrastructure services that are out there, they don't provide redundancy by themselves, or if they provide redundancy, we cannot even rely on that redundancy. It's basically building an architecture where we can build redundancy on top of redundancy and we can make sure that at any point in the architecture, there's no single point of failure. That has been the real evolution of the Okta architecture, and everything that we have been adding is built that way.

For example, if we need to send an SMS. If we want to send an SMS as part of your factor, so that you type in your username, your password, and then we also send you an SMS message, right? Well, we cannot rely on the SMS provider, because the SMS providers could also be having a problem. So we have two of them. We automatically are doing dynamic routing, which means that if one of the providers is having problems or our customers have. We have basically some heuristics that we have built into the system, and the system will automatically switch to other providers. We have to that at any point of the architecture.  So it's not only the services that we use, but also how we use them to ensure no single points of failure

[SPONSOR MESSAGE]

**[00:17:51] JM**: Email has been around for longer than I've been alive, but there's been surprisingly little innovation in inbox management. SaneBox is a new way of looking at your inbox that puts features like snoozing, and one-click unsubscribe, and follow-up reminders as first-class citizens.

If you are overwhelmed by your inbox and you're almost ready to declare email bankruptcy, try out SaneBox. In the onboarding process, SaneBox analyzes your emails and helps you sort them into categories. You can get a free 14-day trial and a $25 credit by going to sanebox.com/sed. That's S-A-N-E-B-O-X.com/sed.

These days, I spend more time in my inbox than I do in front of my coding environment. Back when I was programming a lot, I would spend hours configuring my coding environment because I wanted to maximize productivity. If you spend as much time managing email as I do, it's crazy not to set yourself up for success with your inbox. So stop the craziness. Get sane with SaneBox. Go to sanebox.com/sed and get a free 14-day trial as well as a $25 credit.

Thank you to SaneBox for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

**[00:19:21] JM**: Let's take containers and Kubernetes as an example. This is a piece of software that has become much more accessible to the average engineering organization. 10 years ago, yeah, I don't know if I think containers were a thing in the sense that Google had internal container technologies. There were C-groups and namespaces in Linux, and those could be turned into things that people called containers. But today, yeah, you have the productization of containers in terms of Docker, or the usability improvements in terms of Docker. Then you have the container cluster management system in terms of Kubernetes.

Now, that doesn't necessarily give you anything that you didn't have with just EC2-based virtual machines on AWS, but it does give you some greater flexibility, some better economies of scale in terms of managing your infrastructure. So with that kind of architectural decision, I'm curious as to how quickly you would move in that direction, or if you would even move in that direction, because maybe it doesn't even matter that much. Maybe that's not something you even care about that much, because the thing that's most important is reliability. Maybe this is just such an important server component that you just don't even want to mess with it. So as kind of a case study in Okta infrastructure, how have you approached the newer containerization options?

**[00:20:56] HA**: Yeah. So the reality is that as the Okta architecture has evolved, also our product have evolved as well. We have new offerings. We have kind of new things that we're offering to our customers. So I'll give you one example. A longtime ago we didn't have an LDAP interface. We basically build our own directory. So we can be the master for users. But our APIs were not LDAP compatible.

So we started basically having customers that actually wanted to connect systems that were using LDAP APIs directly to Okta, and we didn't have an interface for that. So we had to build a new interface, a new translation layer, if you will, that would take the LDAP APIs and translate them into something that Okta could understand.

At that point there were two options. One option could be put in the existing system just add a little more code to that translation layer and then you're done. The other option was basically to say, "Hey, let's just evaluate the technology that are out there right now and how we can leverage them and see if we can create a service that kind of looks more modern and gives us all of the kind of new things that we could get from containers from anything that AWS uses."

So we decide to actually go with a microservice. So the LDAP interface that customers can use, it's actually backed by a microservice, and then that microservice is a stateless service that basically those have translation. That's build with containers and built kind of all of the kind of new things that are out there.

So it's a matter of like anything that is new, we try to leverage kind of the new technologies and the new stuff that's out there that is also changing very, very quickly. Then for the stuff that is kind of more legacy, we try to find when is the right time to basically extract maybe a portion of it and make it and try to use the newer technologies, but a stuff that is going to continue to work and is going to be very reliable. So we don't really want to touch it.

When we touch it, we want to make sure that we convert it in a way that we always have a fallback mechanism. That's why any change like that where we actually touch a module that has been working for like – I don't know, now 5, 10 years. It really takes months, if not years for us to make that transition. So we're very, very careful about it.

Of course, we'll not try to leverage the latest things that are out there. So we do have initiatives that are kind of taking some of those modules and evaluating when we need to use kind of newer technologies. But everything new ends up being kind of in whatever kind of new architectures are out there, so being based in those.

**[00:23:32] JM**: Something at the heart of Okta's engineering is integrations. So as you said in your initial description at the product, if I get an Okta account when I sign up for a company, if I join the company as an engineer, my Okta account, when it gets spun up, is going to spin up a user account for the company's Box, and the company's GitHub, and maybe the company's Stripe account, or Twilio and all these different SaaS tools that I might need a login for. Okta becomes the shim over that login experience.

That means that you have to build integrations with all of these different SaaS providers. Integrations can be a really overwhelming challenge for a company that does have to build a lot of integrations, because as I see it, the number of SaaS tools that you have to integrate with is it's growing and growing and growing and you have to maintain those integrations overtime. So how do you systematize the testing and the expansion and the demands that come from a growing surface area of different products that you have to integrate with?

**[00:24:54] HA**: Yup, it's stuff. The way you described it, it's a good way to think about the problem. So the reality is that when we initially developed Okta, we probably needed to focus on maybe like 20, maybe 50 products. You have the Boxes of the world, the Salesforce of the world, and everything was basically build in a monolithic way, if you will. We would be developing those applications. They ended up being kind of like a fixed hardcore classes in Java that would connect to those services. Eventually, it was very obvious to us that we needed to really take more of like a platform approach.

So over the years we basically created a full-on metadata-based architecture so that we can develop these applications much, much faster as kind of new services come along. In some cases, we don’t even have to write code, because at the end of the day you start looking at kind of things that are common across multiple services and then you just need to configure it. So it becomes a matter of just configuring a template, if you will, or templatizing these configurations.

So over the years we have evolved that piece, but it’s definitely one of the biggest things that we are. I think I would say that Okta is three things. So we are an infrastructure company. We’re a security company and we are, of course, an integration company, and not necessarily in that order. Integrations has been really big for us. As I said, we have a metadata system to do all of that. We have application analysts that can tweak or fix an application very, very quickly.

But also we have a lot of heuristics-based monitoring, because it’s almost trying to live in a world where a lot of it is unknown and also you don’t really control it. Because, for example, tomorrow, one of the thousands and thousands of providers that we support could change their API without telling us, because of course they don’t have to tell us. So we need to be able to detect that and fix it very quickly.

Now of course we have really good partnerships with a lot of ISVs, but there’re also a lot of smaller ISVs that are not even going to know that they’re breaking integration with either the single sign up protocol or the provision protocol. So a lot of what we do, and this is very big for us, is being able to monitor, identity patterns, identity kind of regressions so that we can react very quickly. Try to find as many micro additions as we have and also being able to evolve these applications as new services are added.

In addition to that, we also create frameworks so that customers can build their own applications. So if you have an internal built system, Okta doesn’t have to build that application anymore. There used to be a time when we did. Now, it’s basically more of like a platform approach, where customers can actually build their own integrations. That basically exploded as well in terms of the numbers. At one point I remover several years ago, we had about 3,000 or 4,000 applications in what we call the Okta Integration Network. When I did a quick search to see how many customers had built their own applications, the number was about 13,000.

So it's definitely more than order of magnitude, because once you let customers build on your platform, and they may have 5 or 10 internally-built applications that they want to integrate with Okta, then that number becomes really exponential. Just fast-forward to 3, 4, 5 years, and then that continuous to grow.

Last but certainly not least, which is also a very important integration, is that we also extended our reach, because before we only used to do cloud applications. Now we can do on-premise and now we can even do applications behind the firewall. So applications that even our system or our developers do not even have access to, you can still build them with on-premise provisioning or on-premise single sign on that we can support as well. So the number of applications has grown dramatically over the years.

Because of that, we also have a lot of data and we produce a business that work report every years, where we actually show all of the trends in applications and what people are doing with the Okta application network. Yeah, big, big thing for us.

**[00:28:58] JM**: One of the trends in the last 10 years has been the growth of API companies. So obviously Box has an API. But Box is also a rich platform where you're managing files. The more recent things that people think of as API products are things like Twilio, or like Stripe. How does Okta integrate with these different kinds of APIs? Is there anything notably different about Okta's integration with the API companies relative to the more classic SaaS product types?

**[00:29:40] HA**: We also are an API company in the sense that there is a lot of customer that are using Okta for their product and service. So the traditional workforce is one of our areas. The other side is basically using Okta APIs to power your systems. So there are some even consumer websites that are using Okta APIs that you would not even be able to stell that Okta is behind-the-scenes and authenticating the user. But customers can actually use our APIs. When we started kind of having that kind of line of business, we basically started getting really good at really authenticating APIs.

So, for example, we can be an open ID token provider, for example, where people can actually secure those APIs and do API access management with Okta. So that's basically more on the CTO side that I was talking about, where we actually become really authentication for those services and we can give you kind of tokens where you could even kind of authenticate your own services. In a world of microservices, for example, where you even want to authenticate your own APIs, you could also use Okta as open ID provider and have client IDs and create kind of authenticators all built in Okta.

Now, that is a very, very different thing than what we have been describing as basically the workforce or connecting to cloud services. This is more of like what a developer would do. If you go to, for example, developer.okta.com and you're trying to build authentication for APIs, it's going to be a very, very different experience that you go to to our website and go to our workforce side, which is more of, again, the CIO versus the STO. Yeah, I mean the short answer is it's very, very – It's a different world. But Okta as an identity platform, we also think about not only authenticating people, but also authenticating services and authenticating APIs.

**[00:31:37] JM**: I'd like to walk through a login with Okta in some technical detail. Can you just explain what happens when a user logs in to Okta? Give me some of the breakdown of the security key management and the workflow of different services behind-the-scenes.

**[00:31:59] HA**: Wow! That is one of the things that I think is part of our value. I think the value that we bring to our customers is that we make the complex look very, very simple. So there are a number of things that can be involved in an Okta authentication. I'm going to try to do – I guess one of the common ones, which may or may not be one of the simple ones.

So let's say you want to authenticate to – You click on a link. Let's say you get an email and you get a link from – I don't know, a Salesforce record that you want to see. So the way it works is that you see a Salesforce link on email, then you click on it, and your browser, it's going to take you to that Salesforce page. Salesforce knows that you're trying to access a specific tenant. Then Salesforce is configured to use Okta or Salesforce will know that Okta is your identity provider.

So then Salesforce is basically going to redirect you to Okta with some magic direction and then your browser is actually going to show an Okta page and it's going to be your sign on page that is going to be customized to you. We already know the tenant. We're going to show you whatever page it's customized to meet whatever your tenant name is. Then it's just probably going to be username and password, right? The most simple scenario. So you type in your username. You type in your password. That sends those credentials to Okta.

Then at Okta, there are some policies that are evaluated, and then at that point we evaluate where you're coming from, if you are even allowed to authenticate. If you have any geo-location restrictions, for example. If you have any specific policies around your device, around your IP address, anything that gets evaluated in that policy.

So let's your policy requires some multifactor authentication. So then at that point we're going to return a different page to you so that you can use one of the multifactor that you have configured. Let's say that that factory is a push notification to your watch. So at that point, the page that we send you, it's just going to be like, "Hey, you're going to be receiving a push notification. Just accept it."

Then at that point, the backend is going to send via the APN if you're using – Or whatever the Apple push notification service or the Android notification service. You're going to get it to your watch. In your watch you're going to see the location of the request. You're going to get the IP address, some information. You get into your watch. You say accept.

That accept goes through a different flow obviously. Whatever your watch is using for, Wi-Fi or LTE, and that's going to get to Okta. Okta is going to make that connection. It's going to know that, "Okay. I just had a partial session and the verification from this second factor." That is going to create a strong authentication. Okay, now we know who you are.

Now we know who you are. And then we're going to send that – We're going to basically create a SAML assertion. And then that is going to really like a token, if you will, that we provide for your browser so that your browser can actually present that to Salesforce. So that token gets to your browser in the form of a redirect. Your browser basically redirects to Salesforce.

Salesforce will basically verify that that's signed by Okta and there was obviously previous trust that was created when Salesforce and Okta were initially configured. Then finally Salesforce will say, "Okay. We know who you are. The SAML assertion will have everything that they need to authorize you," and then you're going to see the page that you had access to. That's a simple one. I can give you a little bit more complex one.

**[00:35:38] JM**: Wonderful. Now, this illustrates one facet of Okta, which is that, okay, this is core infrastructure and it needs to be quite fast and there's a lot of integration points. There's a lot of network hops.

So I'm wondering, what kind of instrumentation do you have to be able to understand the trace of all of these different network hops and to be able to diagnose latency issues that might be occurring along this complicated request path, which by the way is just one of I don't know how many integrations you have. But you must have some kind of standard instrumentation infrastructure for being able to detect what's going on in this kind of trace. Can you tell me about that?

**[00:36:36] HA**: Yeah. In this particular scenario, I choose not to use multiple identity providers, sort of delegated authentication. For example, if it's an LDAP server or some active director authentication that was involved in this flow, which happens also for certain authentication paths. So you're absolutely right.

So we're manacle about monitoring, and I guess the simplest answer is there is a request ID that is basically passed left and right, but there is also a lot of systems that are involved in this. So we also are redundant in the way that we monitor our service. So we have several products for APMs, several products for log management. So we can actually track all of these and all of the attempts of everything that is happening and we can tie all of these requests together.

But we cannot monitor that at human speed. That would be impossible at the scale that we are. So we also have created a lot of alerts, and dashboards, and heuristics, and analytics around all of these requests that are happening in our system. One thing that is – I don't know how unique it is, but it was definitely unique 10 years ago. Maybe, I don't know, if it's still unique now. But our test team, what's traditional called like a QA team, our team. It's basically engineers that are not only kind of like doing the traditional blackbox testing. But they're basically writing tests and writing synthetic transactions that are constantly happening at Okta just to monitor what's going on.

But in addition to that, they also create dashboards. They also create automated alerts. They also create these heuristics and analytics to identity when trends are changing. And that's basically how we can monitor and react very, very quickly hopefully before customers notice that there is a problem. Then once kind of our automated systems alert, we have basically all the traceability of everything that is going on. And we have thresholds across everything that I talked about.

For example, in delegated authentication, when we have to delegate to another system. That is constantly being monitored. When we see a trend where even like a few customers are seeing slight delays, that basically triggers what we call tribal alert. And those tribal alerts to a lot of architects and engineers that are basically looking to see if there's a problem. If we identity that there is something that could potentially cause problems, then the engineer calls what we call a flair.

Then when you call a flair, it's basically PagerDuty calling a number of, again, architects, engineers and managers just looking to see if there could be any customer impact. After that, we escalate to other phases or yellow and red just to make sure that we can try to detect things that are not doing great in the service even before customers start noticing. Again, this is something that we have evolved over 10 years, and obviously I'm oversimplifying it. Yeah, I mean you hit the nail on the head. I mean, we have to have traceability for everything that is happening, because the complexity is incredible. Also, this has to happen within seconds, if not milliseconds.

[SPONSOR MESSAGE]

**[00:39:46] JM**: Looking for a job is painful, and if you were in software and you have the skillset needed to get a job in technology, it can sometimes seem very strange that it takes so long to find a job that is a good fit for you.

Vettery is an online hiring marketplace to connects highly-qualified workers with top companies. Vettery keeps the quality of workers and companies on the platform high, because Vettery vets both workers and companies. Access is exclusive, and you can apply to find a job through Vttery by going to vettery.com/sedaily. That's V-E-T-T-E-R-Y.com/sedaily.

Once you're accepted to Vettery, you have access to a modern hiring process. You can set preferences for location, experience level, salary requirements and other parameters so that you only get job opportunities that appeal to you. No more of those recruiters sending you blind messages that say they are looking for a Java rock star with 35 years of experience who's willing to relocate to Antarctica. We all know that there is a better way to find a job.

So check out vettery.com/sedaily and get a $300 sign-up bonus if you accept a job through Vettery. Vettery is changing the way people get hired and the way that people hire. So check out vettery.com/sedaily and get a $300 sign-up bonus if you accept a job through Vettery. That's V-E-T-T-E-R-Y.com/sedaily. Thank you to Vettery for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

**[00:41:35] JM**: You've been with Okta for 7 years, and the org has grown a lot in that time. Can you describe how the engineering org structure has evolved since you joined the company and how it looks today?

**[00:41:49] HA**: Yeah, absolutely. I mean, when I joined, we're very, very small. I think we probably had like maybe 10 people in engineering. So it was very, very small. We have hundreds of people now in engineering. But I think the core thing was that I think the organizational structure has evolved much as the company and the business has evolved. We're very lucky, and knock on wood if I find one here, but we do have a lot of people that have stayed in the company like for many, many years as well. We have people that have been here 3, 4, 5, 6, 7 years. So we have a lot of the same people that were in the company when the company was a startup. Continued to be in the company, because again they're excited about the mission. They believe the work that they do is impactful. They basically have evolved with the company as well. So they have created basically teams around them.

So we're constantly kind of obviously improving the talent and the talent pool as well, but also the company has evolved to the point that I think one big thing that I think I could give you as an example of a structure that has changed is that, before, we used to have a team that would just develop, let's say, the product. Any features. The same team would basically look at scalability, performance, reliability. This was basically one team.

At one point in the evolution of Okta, we decided that it was such a critical thing to make sure that we thought like an infrastructure service that we actually divided the team into the team that only deals with infrastructure, scalability, reliability, technical operations and a team that deals with the product features. That was a huge transition for us, because at that time, the product management team didn't appreciate that we were basically really like carving out a big piece of the engineering organization with really strong architects that were only going to be looking at really bits and bytes and not any features.

Then that's also a big transition for the company, but it's proven to be something that was very, very powerful, a very powerful message for the organization until this point. I mean, we have a ratio of people that are only focused on that. These are not just the traditional kind of system operators.

I mean, these are engineers, architects that have been in the value for many, many years that are developing product that are only focused on API performance, on reliability, on scalability on self-healing, and they're not basically – They cannot be distracted, for example, by a customer feature that we need to release very quickly. That they are only focused on the infrastructure part of the service on the core kind of bones and bits and bytes of the service. That's basically one example. I mean, the transition has been quite big if you think of kind of how the company has grown over the past few years.

**[00:44:38] JM**: What's the hardest engineering problem that you're solving today at Okta?

**[00:44:42] HA**: I think the hardest problem – There are a lot of problems and part of my job is always being worried about kind of the next thing and always being paranoid. But I think the hardest problem is just to make sure that as engineering organization, we can get ahead of the business growth. We've been kind of very fortunate that the business continuous to grow where customers are getting bigger and the types of challenges that we solve are getting bigger and more complex.

I do remember that a long time ago we would be scared when there was a customer that was going to go live with – I don't know, like 5,000 users and we were like, "Oh my God! I hope the system can handle that." Now we have basically go live literally millions of users. So it's a very, very completely different game. I think our big challenge as an engineering organization is to make sure that we can anticipate the business growth and we can build technology that is going to be ahead of the growth. For us it's all about business growth. For us, it's about business growth.

So if we were kind of – If the business was at the stage where we were, I don't know, onboarding thousands of users, we need to make sure that the system can do 10,000 users. We basically need to be always one order of magnitude ahead of the business growth. That's a huge challenge for a company like us, where, literally, in the span of this interview, hundreds of thousands of people authenticated Okta, right? So we always need to make sure that the technology can scale much, much faster than the business.

**[00:46:12] JM**: So along Okta's growth, you have expanded into this customer identity segment, and from what I can tell, this is the largest product adjacency that you have moved into. The largest brand-new product adjacency. I mean, obviously, it's related to the core workforce identity product. But it's something different. Customer identity, if I'm trying to manage the identity of my customers, of my ecommerce company, or my media company, that's a very different product than managing the identity of people who are working for me. So what have you learned in the process of building out that product adjacency?

**[00:47:01] HA**: Yeah, absolutely. I think one of the things that I just mentioned as well was in the workforce side of the house, there isn't a lot of companies that have millions of employees. But on the customer side of the house, there is a lot of companies that have like millions of customers. So it's definitely a different game as you mentioned. I think the biggest thing that we have found is that the actual kind of infrastructure and the API monitoring becomes much more complex when you're talking about people that are going to be authenticating, literally, millions of customers that are going to have people that are going to be authenticating, really, millions of times a month versus an employee that will authenticate maybe once a week and maybe once, because then they're going to use whatever they use for their business.

Also, the spikes. There are some very interesting spikes that we see when people go live with new products or services. But I think the biggest learning has been, for us, the way we have to tune the monitoring to understand the use cases of our customers and what's important for them. So I think at Okta we have been getting very good at understanding what our customers are doing with our product, and there are kind of different ways to think about it depending on what they're doing and what their business is.

For example, if you are a grocery store that has potentially thousands of – Or even millions of customers that want to get coupons every specific day. Then we know kind of how the usage patterns are. So we have to build and tune our monitoring so that we know that everything is find for that particular use case. Whereas we have all our customers that have kind of probably go lives and things that are selling that happen all the time that we also need to get ready for that.

Also the type of customers that are more like a set schedule of things that happen every month. Basically, what I'm trying to say is that we have to have tuning and monitoring that tailors very specific use cases. And in the consumer space, it's much, much more than the workforce identity, and the variety of those cases is exponential as well.

**[00:49:15] JM**: Okay. Last couple of questions. Tell me something that you believe about modern software engineering that I would be unlikely to hear from anyone else.

**[00:49:24] HA**: I'll give you two, and these are two things that are really, really embedded in the culture of Okta. The first one is that continuous integration loops are not enough. I listen to that specifically when we talk to small companies that have never really seen large scale and they'll tell you, "Oh, yeah. Just make sure that your test cases as well done, and then you'll never have problems in production."

In a product like Okta, that would be very hard, because we're always looking and designing for their known. We don't know how our customers are going to be kind of integrating with our products and services. We don't know kind of the integrations that are customers are building. So our test cases have to be really good, of course, and they have to prevent everything that we know. But we also need to get ready for their known.

So the pretesting, if you will, becomes – Or the post-testing if you will becomes as important as a pretesting. So monitoring synthetic transactions and really understanding how the product is being used becomes sometimes even more important than what you do pretesting. So that's kind of one thing. Really, the fellas that continuous integration, continuous deployment loop is always enough for quality. I think that's just the beginning. There are much more things that you have to do to ensure a service with high quality.

The second one is we have a hashtag that we use at Okta called #nomysteries, and that has been something that has been very evident at Okta, where because we're dealing with so much complexity, sometimes engineers will kind of default to saying, "Oh, you know what? There was a network blip, or maybe a customer is doing something that we were not expecting, but they've stopped and we don't have to look into it."

So because of that, we have this #nomysteries, because for every single thing that we identity in the product, we need to find the root cause for it. Sometimes we spend, literally, months looking into exactly what the root cause was for specific thing even if it didn't have exact customer impact. And because we're manacle about that, we have identified a lot of very interesting things in our software, in the products that we use, or even in our providers that have helped us build a much more robust and high-quality service.

So for us, the no mysteries and the post-testing, if you will, is a fundamental part of our culture that I don't think is needed in all organizations. I think it would help all organizations. But I don't think it's as critical as it is for Okta to have those two things really nailed.

**[00:52:01] JM**: Okay. Hector, that sounds like a great place to close off. Thank you for coming on Software Engineering Daily. It's been great talking to you.

**[00:52:06] HA**: Thank you.

[END OF INTERVIEW]

**[00:52:16] JM**: Monday.com is a team management platform that brings all of your work, external tools and communications into one place making cross-team collaboration easy. You can try Monday.com and get a 14-day trial by going to Monday.com/sedaily. If you decide to become a customer, you will get 10% off by coupon code SEDAILY.

What I love most about Monday.com is how fast it is. Many project management tools are hard to use because they take so long to respond, and when you're engaging with project management and communication software, you need it to be fast. You need it to be responsive and you need the UI to be intuitive.

Monday.com has a modern interface that's beautiful to look at. There are lots of ways to use Monday, but it doesn't feel overly opinionated. It's flexible. It can adapt to whatever application you need, dashboards, communication, Kanban boards, issue tracking.

If you're ready to change the way that you work online, give Monday.com a try by going to Monday.com/sedaily an get a free 14-day trial, and you will also get 10% off if you use the discount code SEDAILY.

Monday.com received a Webby award for productivity app of the year, and that's because many teams have used Monday.com to become productive. Companies like WeWork, and Philips and Wix.com. Try out Monday.com today by going to Monday.com/sedaily.

Thank you to Monday.com for being a sponsor of Software Engineering Daily.

[END]