EPISODE 1223

[INTRODUCTION]

[00:00:00] JM: The shift to microservices architectures and distributed systems has been a challenge for systems using conventional security practices such as IP filtering and network policies. In addition, the increasing intersection of development and operations exemplified by the DevOps methodology has expanded the scope of responsibilities in implementing secure systems.

SPIFFE is a set of open source specifications for issuing identity and services such as heterogeneous distributed environments and microservice cloud native architectures. Systems implementing SPIFFE bypass the need for application level authentication and network level ACL configuration. SPIRE, or the SPIFFE runtime environment, is a system that implements the SPIFFE standards to manage platform and workload attestation providing an API for controlling policies and coordinating certificate issuance and rotation.

Derek Edwards is the head of engineering at anthem.ai, and Ryan Turner is a software engineer at Uber. They joined the show today to talk about the challenges of managing security in a distributed system. How adopting SPIFFE represented a paradigm shift in their authentication workflow and how the SPIFFE and SPIRE projects are evolving to meet the needs of the next generation of cloud native applications.

[INTERVIEW]

[00:01:14] JM: Guys, welcome to the podcast.

[00:01:16] DE: Thank you.

[00:01:17] RT: Thanks for having us.

[00:01:18] JM: I'd like to start off by defining the term identity when it comes to infrastructure. What does identity mean in infrastructure?

[00:01:29] RT: Yeah. So I kind of think about identity in three broad categories across the infrastructure. One is like your host-level identity, and oftentimes this is something like an X509 or an SSH certificate that is used for provisioning on the host. And then you have kind of a set of applications which run across your environment. And some of these are like infrastructure kind of components and platform services. You might think of like schedulers or host agents in this category, things like load balancers, proxies. Things that are not solving a core business problem, but are used as a building block for deployment of applications.

And then like the third level of identities I would consider, like higher-level business domain applications. And these kind of use those infrastructure services. They run on these hosts in your deployment. And so all these three kinds of components sometimes have different ways of representing identity, but those are kind of the three broad categories I generally think about.

[00:02:36] DE: Yeah, I would agree. I think when we think about identity, a lot of our historical concepts of identity, especially as you think of hosts and you think of processes, they kind of came about very much in the day of bare metal, really managed services, servers on-prem. Machines that you actually physically control, and in a day when you control the bits end-to-end. And so I think a lot of these underlying foundational descriptors for identity are still very much true, still very relevant, but you start to run into a little more of a challenge when you try to deal with the abstractions that we have today as you get into VPCs and clusters and you have VMs, micro-VMs, containers on top of that. So it's becoming a little more – The definitions are becoming a little harder to classify very strictly speaking, but some of those foundational layers are still very true.

[00:03:28] JM: Tell me about the SPIFFE project. How did SPIFFE come about?

[00:03:34] RT: Yeah. So SPIFFE was kind of born out of this idea of zero-trust networking. And this idea of zero-trust networking is kind of a security model where you assume that your

network is not safe by default, that there might be some malicious actor on your network. And in the past we traditionally have kind of thought about networking security in terms of perimeter-based security where there's some firewall, which sits on the outside of your network, and that sets different policies or rules about what kind of traffic can come in and out of your network. But then once you're inside the network, that's seen as a trusted zone and everything can kind of freely communicate there.

Unfortunately, we've kind of experienced time and time again that there have been a lot of drawbacks to this security model. There are so many more interactions and devices and processes, which now run on networks and communicate with each other over networks that it's very difficult or impossible to anticipate all the different kinds of interactions which might happen on your network. And that just broadens the attack surface for your network.

So this idea of zero-trust networking kind of evolved into this workload identity concept where we wanted to be able to provision identities to specific processes that run across an infrastructure and use that identity as a way to securely authenticate all remote procedure calls across an environment. And so this kind of flips the networking security model on its head and says like, “We don't trust anything on the network by default.” And everything that goes across the network needs to prove that it's something that's trusted on that network. And so that was kind of where SPIFFE came about is through this zero-trust networking idea.

[00:05:32] JM: Can you give an example of SPIFFE in an application?

[00:05:39] DE: Sure. So I think a good example of this would be, say, that you actually have two different two different services that need to communicate with each other over a channel. SPIFFE really tries to provide this normalized identity so that when you think about – When you think about a particular service that's running in a container running on a certain node, you're able to uniquely identify that that instance of that validated workload and that particular host that's gone through attestation. So you know that if you have a database on the one side, say that you're talking to a Postgres database and you have an actual API server sitting on the other side of that channel, that both sides can actually have confidence that you know who it is

that you were talking to on the other side and are able to continuously verify that. And this really spans across or expands outside of the idea of perimeter firewalls as Ryan touched on before, where it doesn't necessarily matter that you may be operating in a separate cluster or a separate cloud or bridging on-prem versus a cloud environment. But you can actually have these identities that are managed through SPIFFE and be able to run those against a policy in a declarative fashion.

[00:06:56] RT: Yeah. I think Kubernetes is a great example of a platform that was kind of born out of this idea of decomposing large monolithic applications and services into microservices is what we call them. And when you break down applications, that makes it a lot easier in many ways to manage those applications and their life cycle. But then it creates a lot of additional challenges around the networking and security models. So more and more microservices architectures have become very common, and especially with cloud native applications that have been built in the last few years, many applications have this problem of how do we ensure that all the remote procedure calls do have some sort of strong authentication material?

[00:07:46] DE: Yeah. And I would actually add to that, that really where you start to see the strengths of having more of a cloud native identity management mechanism like this, it's truly the idea of portability. When you have mechanisms that are more tightly baked into a particular cloud infrastructure or particular vendor, it may be feasible for the short term if you know that you're going to be running strictly within a particular cloud. But as soon as you need to start to explore more hybrid scenarios, if you do need to blend a bit of on-prem and cloud or if you need to span across vendors, and especially in the case of the work that we've done with anthem.ai, we know that partners where we're doing integrations. They very likely have made different choices in their cloud vendors than we have. And as a result it's become all that much more critical that you can have these true cloud native components that you're able to deploy in a portable fashion and you can have a degree of confidence that they'll behave the same regardless of the environment in which they live.

[00:08:51] JM: What is the SPIRE project and how does it relate to SPIFFE?

[00:08:56] RT: Yeah. SPIRE is an open source implementation of the SPIFFE specification and it's a CNCF project, which is now an incubating status. So CNCF, for those who aren't familiar, is the Cloud Native Computing Foundation that sponsors basically the advancement of a lot of cloud native projects in open source and was kind of the driving force behind Kubernetes and many other projects like Envoy which people have probably become familiar with over the years.

So SPIRE is an actively-maintained open source project and it provides a way to deliver SPIFFE identities throughout heterogeneous environments. And so what I mean by that is that it can support workloads running anywhere, and that includes public clouds, private clouds, and it can scale up to large deployments as well where you might have thousands or more of workloads running on thousands or more of hosts. And it's flexible and dynamic enough to be able to securely provide identity in all those environments.

[00:10:06] JM: Can you tell me about a SPIRE deployment and what are the problems that it solved?

[00:10:15] RT: I can speak to one example from work that we're doing within the anthem.ai group is to actually – To establish a mechanism for federated connectivity between platforms that we're building internally in our group and some of our partners that are helping to build additional ML models against our data. They are able to have their own cloud infrastructure in order to actually host their processes, host their workloads that are able to work with our data, be able to generate insights based upon that data. But we are able to leverage SPIRE in this case as a means of establishing these trust domains on both sides and to establish this connectivity between our clusters in a way that we can exchange data back and forth having this this zero trusts networking pattern applied and ensure that we can actually confirm and continuously verify the identity of the workloads on either side.

And so in this case, SPIRE, as we deploy it on one side of that connection or the other, it really establishes the ability to generate these workload identities and to be able to seamlessly

essentially exchange these trust bundles between the two sides so that we can verify identities that are established from one side of the other.

[00:11:40] RT: Yeah. And one thing I'd add on to that as well is one of the great things about SPIRE is that it automatically handles the rotation of credentials. In this case we're talking about identity, and that comes in the form of what's called a SPIFFE verifiable identity document, which can be represented either as an X509 certificate or a GWT. So this has always been a problem with identity and secret access management where credentials are created at one point in time and then they're oftentimes not rotated very frequently. And if those credentials are ever leaked, then the attack surface is very broad for that. So by default, SPIRE will automatically rotate, and using a push-based mechanism will give workloads who have already received an identity their updated identity. And so this is one of the really powerful elements of SPIRE that we've really come to appreciate at Uber as well.

[00:12:39] JM: How does SPIRE solve some of the problems with traditional authentication methods like username and password and RSA keys and so on?

[00:12:49] DE: Well, I'd say that – I don't think there'll be any argument from any anyone in this group that really traditional username-password, multi-factor auth. It's one thing to leverage those for individual end user identity, and arguably I'd say that even that's inadequate these days. But that's a topic for another day. But when you really think about it in terms of workloads and machine-to-machine service kind of communication, it really boils down to, again, the idea that you have a set of credentials, API credentials, that following any kind of good security practice you need to be able to rotate those on a regular basis. And as you look at things at scale, chances are in any kind of high-availability set up you're going to be running fleets of these instances for any kind of workload. And to try to keep those synchronized in a more traditional pattern, and if you have API credentials, you're going to deal with a lot of headaches either to try to synchronize them or something that's very brittle. And so in my mind at least that's really the inherent challenge with relying on more traditional authentication mechanisms especially as you deal with more workload-level identity.

[00:14:02] RT: Yeah. Another thing I would add on to that as well is a lot of times the first thing that a workload needs to do when it comes up is get some credentials, right? It needs to be able to get database credentials or an API key or something that allows it to make requests or fetch some data or perform some operation over the network. And a lot of times we run into this kind of circular problem where we need credentials when the workload comes up. But in order to get the credential, you also need a credential. So like the credentials themselves may be stored in something like a secret store that itself requires some sort of authentication. So you sort of come to this conclusion that like no matter what you do you always have to provision some sort of secret with the application itself. And so that makes it challenging to manage the credentials over time.

One of the great things about SPIRE is that the model it's based on is the workload basically communicates to SPIRE over this workload API and says, “Hey, I’m a workload. Who am I?” You figure out. It's not that workload's responsibility to present any sort of credentials. It's the workload identity platform's responsibility to identify the key runtime characteristics of that process and identify who that process is and then provision the identity to that workload.

And so in this way we kind of solve this like bottom turtle problem, which is something that we talk about in the SPIFFE and SPIRE book. So a cheap plug for that. But it really makes it a lot easier to manage the credentials across your environment when you know you don't need to provision them with the application itself.

[00:15:45] JM: Could you talk about how the operations teams or the owners of individual services need to interact with SPIRE over time? What do they need to do to ensure that their workloads are SPIRE-compliant?

[00:16:03] RT: Yeah. In terms of operation – So I’ll just kind of speak from my perspective, because at Uber uh we sort of don't have a dedicated operations role. Our engineers are also responsible for maintaining and operating services. So I’ll speak more from that perspective. But from a service owner's perspective who needs an identity, really, this is a pretty easy thing for them to get going with. We provide libraries at Uber that basically make it easy for

developers to integrate with SPIRE and get their identity. And it's a purely configuration-driven integration for them. So they need to do some sort of onboarding where they register the identity in the platform. And then once that identity is registered, they just need to add some configuration stanza to their service configuration and then by default they will get an identity when they come up. So that's made it pretty easy from their perspective.

And I would say as far as like seeing whether things are working or not, we emit metrics and logs in our libraries, which allow developers to have more confidence that their integration with SPIRE is working as intended. So that there's some additional development that our team, a team Uber who manages our workload identity system, has done to make it a little bit easier for developers.

[00:17:32] DE: Yeah, I'll add to that, that we've actually developed a set of tooling, a set of configuration in a similar manner. From the sounds of it, I think we're a little earlier in our journey to really polishing out this tooling to completely eliminate all of sort of the onboarding friction that's involved there. But what we've been able to do within our group is to really leverage kind and being able to set up more of these realistic local environments as developers are bringing onboard new services. In the beginning, in the earlier days, as we were working with SPIFFE and SPIRE, it was a little bit of a challenge in the earlier days to initially ensure that we could get a service pod into a working state where it was necessarily happy and you had an identity that we could begin to leverage for applying policy between these connections. And it's taking a little bit of effort to get to a point where now we've applied around layers of tooling our own scripting around the service onboarding process so that as we go about building new services and pulling them into the overall deployment configuration that it's a little more of a push button type of a deployment. A little closer to what Ryan described.

[00:18:55] JM: For companies that are evaluating, implementing SPIRE, what would you say is the cost and how should a company like measure whether or not it is worth the effort of implementing SPIRE and what's the technical debt that they'll incur?

[00:19:17] RT: Yeah. I think that's a really deep and layered question. I think really depends on the organization and the architecture of their systems. I think for any sort of microservices-based architecture, if you're running services across multiple environments or you have plans to run across multiple environments, it really makes sense to try to build on top of identity primitives that are not tied to a specific environment. And this is kind of the problem when you start relying on public cloud identity systems, for example, where everything is great when you're operating in that one public cloud. But then the moment you want to try to deploy into a different cloud, you have a different set of identity primitives and then there's all of a sudden a lot of complexity in trying to have secure authentication across those environments.

And so by building on SPIFFE and SPIRE, there's certainly some investment there to get it going. There needs to be someone who is operating that platform for the organization. And it takes someone with some deeper security expertise who understands some of these concepts and can evangelize and communicate that to the organization. And that's not always a skill set that is common among engineers. So that's something that is definitely a challenge for organizations. But I think the security benefits kind of speak for themselves. Once you have this kind of identity system up and running, it really becomes pretty easy to make your workloads safe by default in terms of securely authenticating in all of their outbound calls. And that really limits the attack surface for anyone who can get onto a network without a secure identity even if those identities are compromised, the SPIRE issued identities. In most cases they're intended to be short-lived. So the attacker only has a very limited time frame to do anything with that identity. So I think the benefits are many fold, but there is certainly some investment and probably technical debt in order to migrate from whatever identity system if an organization is using identity platform already. So we use SPIFFE and SPIRE constructs.

[00:21:39] DE: Absolutely. I would agree with that. I think, really, it's going to depend very much on the specific company situation. Yeah. We were fortunate when we were starting our work on the anthem.ai side that we were largely working through the lens of a startup, very much greenfield work in a lot of ways. And for us, we knew from day one that we needed to bake zero trust concepts into the core of everything that we were doing. The understanding that our goals are to be able to leverage data, to be able to deliver insights, to be able to

collaborate with different partners, to build this more seamless experience. We essentially knew that some of the tradeoffs that we talked about earlier in terms of managing key rotation and things of that vein, that we were not willing to make those kinds of tradeoffs.

And so for us, at least within the AI group, it was a no-brainer. We knew that we wanted to do this, that we were willing to make the investment. I think as you look at more – As you look at a broader organization, a company that has a lot of established infrastructure, you kind of have to know what kind of challenges you're trying to solve and really where you are. It's not to say that these patterns are a silver pole that will solve all problems as it relates to security posture. But I think every company has to evaluate where they are and what they're trying to solve. And I think for us it was very much driven by, in our case, knowing that we wanted to do the utmost to safeguard the data and the privacy of our members.

[00:23:23] JM: I'd like to get into the specific use cases of Uber and Anthem respectively, where you guys both work. So let's start with Uber. Ryan, what were the drivers behind the adoption of SPIFFE and SPIRE at Uber?

[00:23:41] RT: Yeah, for us – So we have a pretty rich microservices architecture that powers our platforms. And for Uber, data security and privacy is a huge core priority for us. Users are trusting us with really sensitive data in many cases, their location, their credentials, like driver's license, their payment information. So this is all really sensitive information that we take really seriously and we want to make sure that anytime data in our environment is accessed or some operation is performed that we're following best security practices and privacy practices with that data. So the need for something like this workload identity system like SPIFFE and SPIRE was really pretty evident for us. And we had tried to solve this problem on our own initially by building our own sort of workload identity platform.

And as part of building that system, I think us as a security organization really learned some pretty hard lessons about the challenges of reliability and building these kinds of platforms securely. There are so many considerations to make. And it's really hard for any one

organization to build this kind of a system in a way that is flexible enough and secure enough for many use cases.

So we saw a great opportunity with SPIFFE and SPIRE to really unify on the identity primitives that we use across our infrastructure and we run a really large complex and heterogeneous deployment at Uber. So interoperability is also a pretty significant concern for us. We wanted to make sure that whatever system we built or adopted was something that would scale up for the needs of our business for years to come. We now run a really global company with multiple lines of business, in rideshare delivery, freight and several others. So we were really focused on scalability, reliability of whatever system that we adopted. And we saw SPIREs distributed system server and agent architecture and thought that it was something that would be able to scale to meet the needs of our business, which was really promising for us.

And the other thing that we really appreciated, I think about SPIRE specifically, is that it has this built-in plug-in architecture where you can extend where you need to. So we run, in many cases, homegrown infrastructure kind of systems. And interoperability is sometimes a concern there, because when you build things on your own, you sometimes have to keep building things on your own to make everything work together. So that was definitely a concern for us, but SPIRE really solves this problem really elegantly with its plugins. And we have definitely leveraged that capability of SPIRE to bring SPIRE to be adaptable to our environment.

[00:26:47] JM: And Derek, how does the impetus for implementing SPIFFE and SPIRE at Anthem compared to what Ryan has discussed so far?

[00:26:58] DE: Yeah, it's interesting. I think actually probably a lot of the same drivers in many ways, with the work that we're doing in anthem.ai. I'm leading a group that's building essentially a next generation platform for running and employing data science and AI at scale. All of this really in the name of trying to solve some big health care challenges. There's no shortage of challenges in this space. And for us, again, for very much the same reasons, it's paramount to really safeguard the security and privacy of our member data as we do go about trying to deliver better data and insights.

And so from the outset we wanted to make these principles into the platform as really a core element that every service that runs within this infrastructure that there is this verifiable identity tied to it, that we can leverage that to establish MTLS connectivity between every component of the stack and to ensure that we can do this not only internally but externally as well. And we knew as we were starting to build this platform that we would have to be able to bridge across different environments. Not just our existing on-prem infrastructure, but in private cloud public, cloud environments and very likely across vendors within the uh the cloud space.

And so in order to really facilitate this, we did want to bake these zero-trust principles in. And I think in a lot of ways what Ryan described, our journey sort of started in a similar way, where we try to sort of go about it ourselves in a very manual fashion to establish these identities. And you quickly learn without really going that far down the road that that is going to be a very painful journey to try to maintain that and to try to scale that. And so it became very painfully obvious very quickly that we needed a means of ensuring that we could establish workload identity, a verifiable workload identity, and be able to deploy that across our stack, do so in an automated fashion and a way that can be really centrally orchestrated. And that led us to start exploring the SPIFFE and SPIRE projects in the beginning. And try to really understand what the entire zero trust community was about, all the activity around CNCF.

And so this has not only really started to manifest itself within our infrastructure on the anthem.ai side internally, but the great thing that has really come out of all of this is that we do have this repeatable pattern that we can deploy now as we do work with new partners, with new integrations so that we can ensure that we're not only applying these policies internally, but we can actually apply these same policies for every partner that ends up coming in touch with our infrastructure. And we can ensure that we are safeguarding our members' data.

[00:30:02] JM: Were there any unexpected snafu's that either of you ran into during the implementation?

[00:30:10] RT: I think from the Uber side, we were kind of core um to the development of SPIRE. In the last couple of years when we started onboarding realized like the platform is very promising, but there were some things that we felt like needed more improvement to be production-ready for us. And I think one of the major things that we tried to tackle was really validating that SPIRE can scale to our environment's needs. So there was quite a bit of investment that we made to do research and actually make changes to the platform itself to improve the scalability of the platform.

And so this was kind of a competing problem that we were trying to solve with actual adoption where the more users that you onboard to the platform, the more the scalability problem becomes an issue. So that was something that we weren't necessarily sure about initially. And anytime we try to build a system that can work in some large environment, it takes time to mature it and to understand it and understand the bottlenecks. And so we have spent a lot of time and effort there and thankfully now have a deployment that has been running very stably for the past year and a half here at Uber. So that would be the main thing I would say has been sort of a challenge or a learning for us deploying SPIRE.

[00:31:42] DE: Yeah. I would say from our side that we – Within anthem.ai. If anything there, there have been fewer technology and more technological challenges. There have been some things that we've learned as we've gone along in terms of the best patterns for trying to integrate SPIFFE and SPIRE into our clusters as we deploy these workloads and getting them incorporated into the actual definitions within our pods. But I think if anything, where we've run into some – Some friction has generally just been around our own understanding and actually really wrapping our minds around the right way to roll some of these patterns into our services and how to build additional tooling around that. So it's not necessarily actually rolling SPIFFE and SPIRE and zero-trust into your engineering on its own, but how can you essentially add additional tooling and automation around that from an operationalization perspective so that you can understand and reason about if you do have something that's broken, because that is related to your SPIRE-generated identity or some kind of connectivity that is failing between services. Just having the right kinds of instrumentation in place, the right kinds of observability and logging in place to be able to catch and understand what's happening.

And so those were some of our learning experiences coming out of that. If anything, probably some of the bigger challenges that we've faced is just trying to, for more of a mindset perspective, being able to help everyone within AI. And as we start to talk more broadly in the organization, just understand how this really works and how to apply this in the context of our existing security patterns. Understandably with any kind of mindset shift there's going to be a fair bit of scrutiny. And so it's a good scrutiny. It's something that we want to now we want to make sure that we are applying the right patterns and using this in the right way. And so I think it's something just as you go into it, as you explore this path, it's not going to be uh – Not something that you'd be surprised about or unexpected, but that's been a bit of our journey going through this. But in the end I'm confident that it will actually support us well along the way as we try to move toward a more digital-first kind of a future.

[00:34:14] DE: Yeah. And I'd actually add one thing to what Derek said there. In terms of adoption of SPIRE across your entire infrastructure, that's been sort of a learning experience and challenge for us. I think when you look at the core problem that SPIFFE and SPIRE was originally designed to solve, it was this concept of workload identity. But in many cases, that really means service identity. And in those kinds of use cases the integration patterns are pretty clear, but there are a diverse set of workloads that we run at Uber, which include like build and test jobs, include like batch workloads and data workloads that have more opinionated stances towards identity and runtime environment.

And so there have been some interesting challenges that we've been trying to work out where we've enabled authentication and authorization on some services and then realize that some of the callers are in some of these environments like a batch job or a build environment where it's running some untrusted code. And so we've been faced with some of these difficult integration scenarios where it's not always clear exactly how to securely deliver identity to those kinds of workloads. And so those are ongoing challenges that we're still working on. But I think the learning for us has been that service-to-service identity is very easy to get up and running with this. And then the broader set of use cases is something that we still need to kind of think through.

[00:35:48] JM: Is there anything else you can add about how you're securing the next generation of applications that you're building at your respective companies with SPIRE?

[00:35:59] RT: Sure. Yeah. I can talk a bit about that at Uber. So at Uber we've been actively working on an internal service mesh infrastructure project, which really extends the reach of where SPIRE issued identities are propagated throughout our remote procedure calls in our environments. So the traditional pattern with service mesh is that you have some sort of proxy, which is co-located with a caller and enforcer service. And that proxy is responsible for doing some of the complicated network routing and service discovery tasks as well as authentication and authorization at times. So we've been trying to build out this system at Uber that makes it really easy for users to do the right thing with authentication by doing it on their behalf. I think most developers don't have a lot of familiarity with some of the concepts of authentication and don't always know exactly how to integrate with a system like this. So with the service mesh we're really hoping that it'll accelerate the adoption of SPIRE at Uber and really bring us closer to that vision of zero-trust networking.

[00:37:13] DE: Yeah. Within anthem.ai, I'd say that for us the primary things that we're trying to solve really always goes back to how we safeguard the underlying data that we're working with. A lot of the infrastructure that we are building within the AI group is how do you run data science? How do you conduct data science? How do you actually train and deploy AI models against data and actually deploy it at scale? And this really goes in support of a lot of bigger picture initiatives. And there are some that you can start to see even within the more public-facing sides of the developer program where we know that we actually are hosting hackathons. We are hosting the ability for third-parties to actually start to explore de-identified and synthetic data and actually how we can start to build models against that.

And so a lot of this infrastructure is built with that in mind and how we can support doing that in a very scalable manner, very secure manner, so that we actually – Again, everything that we're doing is trying to move everything toward this digital-first future for health care and that

this infrastructure is built very much with that in mind. Away from a lot of fragmented systems and into something that can really support this central point of building and experimenting.

[00:38:44] JM: I'd like to revisit this concept of zero-trust networking, because this is a term that people probably hear a lot. Let's just revisit and explain how SPIRE attains zero-trust networking or works towards it.

[00:38:59] RT: Yeah. So SPIRE, so it implements these concepts defined in SPIFFE of a SPIFFE identifier and some sort of representation of that identifier through an identity document called a SPIFFE verifiable identity document. And that identity document is something that is cryptographically verifiable. So it's ultimately signed by some key in the SPIRE environment. And these identities are all sort of grouped under this construct called a trust domain, which kind of defines the logical boundaries of which identities should be able to interact with one another. And so a trust domain itself is essentially represented as cryptographic keys that are used to sign other identities in that domain. And so that's kind of where we developed this system of strong authentication is this cryptographically verifiable identity document called an SVID.

And then furthermore, those identity documents need to be actually delivered to the workloads that they're intended for. And so that is exposed through something called the workload API, which it runs on the SPIRE agent, which is a host agent in the SPIRE control plane that runs on every host where a workload might want an identity, which in many cases is all of your hosts. And so that workload API is what's responsible for delivering an initial identity to the workload when it comes online and requests its identity. And then it pushes the rotated identities over time to that workload over a long lived stream on the host.

And so that's kind of the general flow of how identities are provisioned within a trust domain with SPIRE. And then once those identities are made available to the workload, they can start doing things like mutual TSL to ensure that all the network traffic that they're sending is mutually trusted with the server that they're communicating with and encrypted. And identities can also be sent as tokens like JWTs that are attached as headers to network requests. And

those keys ultimately that are represented in the JWT are ultimately routed back to the trust domains signing key and both sides of that request response, the client and server both know about those trust domains keys because they're delivered through the workload API to those workloads. And so they always have a way to verify that any workload that which presents these credentials is indeed valid and trusted.

[00:41:46] JM: How is SPIRE different than other approaches such as secret managers or other cloud provider tools that might work for identity management?

[00:41:58] DE: I would actually say that in many cases it would work in conjunction with a lot of those secret managers. I think that a lot of these cloud-based secret managers are really abstractions if you have basically an HSM, some kind of some kind of module for actually generating a secure token. And in many ways, SPIRE, SPIFFE can actually leverage those components as uh as a means of generating these workload identities. So it's not to say that they necessarily are mutually exclusive or that one is necessarily a replacement for the other. I think the secret managers, it is a means of holding on to secure keys or generating secure keys on your behalf. And in my view, SPIFFE and SPIRE are generally leveraging these components as a means of actually composing this workload identity.

[00:42:58] RT: Yeah. One thing I might add there as well in terms of cloud provider identity access management tools, a lot of times these systems are designed to work nicely with the platform as a service components that are offered by a public cloud provider. So if you're within their walled garden and you're using their services, it's pretty easy to get going and use those identity constructs typically. Where things get a little more challenging is when you are running workloads in public clouds, for example, that are basically on top of IaaS kind of VMs where the public cloud provider doesn't really know anything about the workloads which are running on those hosts. And so the best that the public cloud provider identity management system can do is give you a host level identity. But as we know, with schedulers like Kubernetes becoming more and more popular, hosts can run many workloads. And so giving a workload a host-level identity is not really a fair way to represent that workload. And workloads can run on many hosts. So it's not exactly a great one-to-one mapping between a workload

identity and the identity that the cloud provider tool is going to give you. And so that's where SPIFFE and SPIRE really come in to save the day. The local SPIRE agent is able to introspect the processes which are requesting the identities in various public cloud VMs and it is able to actually figure out who those processes are and whether they're known or trusted in the network.

[00:44:40] JM: So as we near the end of the conversation, do you guys have any reflections on what's coming up next in the SPIFFE and SPIRE projects and how workload identity management is going to improve in the near future?

[00:44:55] RT: Yeah, I can talk a bit about that. So full disclosure, I'm also a maintainer on the SPIRE project. So I can talk a little bit about some of the proposals that have been made recently and that are kind of in the works. So the main thing I think that we're trying to drive home soon is this mode of running SPIRE with workloads that are in kind of a serverless environment. So they're running on some infrastructure that is not really directly controlled by the operator of that workload. So this kind of breaks the patterns of SPIRE a little bit in the sense that SPIRE kind of depends on an agent to be running on whatever host that the workload is running on. But in these serverless environments, we can't control deployment of an agent in those environments. So we're actively working on some use cases that we would support where we'd be able to provision SPIRE-based identities that are in an environment where an agent is not running. So that's one major feature that's in progress.

Additionally, there's been a lot of research and work going into attesting nodes within an infrastructure using trusted platform modules, which are commonly known as TPMs. And so this is like essentially a way to authenticate a node on the network that is running a SPIRE agent using keys that are provisioned at the time the hardware is manufactured. So that's another feature. We're also investigating, supporting confidential workloads which might run in secure hardware enclaves like Intel SGX. So this is a really interesting area that is still kind of in in research phase, but this would open up the doors to a lot more confidential workloads that may not be able to run in general purpose compute infrastructure.

And there are some other things going on as well like supporting AWS key manager service as a signing service for all the SVIDs in SPIRE and we also are looking into ways to improve forcing the rotation and revocation of SVIDs in the SPIRE ecosystem. So a lot of things going on. Yeah. So there's a lot of exciting features coming up in the next year here hopefully.

[00:47:14] DE: Yeah. I'll just say from our side in anthem.ai. Again, I can't speak as closely to the road map of the overall SPIFFE/SPIRE projects nearly to the same degree as Ryan, but certainly for our part we do see a lot of the work that's going on in the community not just within SPIFFE/SPIRE, but the entire zero-trust community and CNCF being very central to a lot of the work that we were doing ahead and really being able to deliver on our vision for leveraging data and insights for our members. So I'm very excited to see what comes out in this space.

[00:47:48] JM: All right, guys. Well, thank you both for coming on the show. It's been a real pleasure.

[00:47:54] DE: Likewise. Thank you so much for having us.

[00:47:55] RT: Yeah. Thanks for having us, Jeff.

[END]