TRANSCRIPTS

KERI Mini Conference November 2026

A complete transcript of every recorded talk from the KERI Mini Conference November 2026.

7	5:41:53
TALKS	RUNTIME

November 27, 2026

CONTENTS

Full transcripts of all 7 recorded sessions, in the order they appear in the archive.

01	2026-11-02-kent-bull	59:19
02	2026-11-02-miniconf-promo	00:51
03	2026-11-02-ryan-hansen-evan-asakawa	52:26
04	2026-11-02-ryan-hansen-samuel-m-smith	55:50
05	2026-11-02-samuel-m-smith-1	57:27
06	2026-11-02-samuel-m-smith-2	57:28
07	2026-11-02-samuel-m-smith-ryan-hansen	58:28

TALK I

2026-11-02-kent-bull

KERI Mini Conference November 2026 · talks · 59:19

SUMMARY

AI generated summary.

Kent Bull presents did:webs as a short step past did:web: the identifier is still a domain, but a keri.cesr CESR stream lets verifiers check key history after rotation. The designated alias commits the identifier to that domain, and ConditionalProof2022 describes nested weighted multisig so non-specialists can verify threshold proofs. An implementation exists; the spec still needs versioning and registry details.

Thank you, Karla. And thank you to GLEIF for letting me work on did:webs. We'll get into that a little bit more. We're going to start with the TLDR. But who am I? I'm Kent Bull. I work for GLEIF as Karla mentioned. She was actually the person who hired me at GLEIF two blessed years ago. Was it two? It's going to be… it's going to be two in April. Wow. No, that's going to be three. Oh, that's fast. Wow. I had three kids and two kids in the last couple of years. Things went fast. All right. Did:webs, what's the why the s matters is the s matters most. And of course, we've got did:web. We're trying to say this is very close to did:web. And by the end of this presentation, we'll circle back to this. You'll see what we add in addition to what did:webs provides. And it's just enough so that you can verify with KERI ACDC that you actually control the domains and the keys that you say you control. Let's start with the TL;DR. For those, there are enough people in the room that have seen me before that have seen did:webs that know kind of where things are at. Let me give you a quick update before we dive into the next 23, 24 slides that will go in depth. Essentially did:webs today works with KERI ACDC, I'm going to say version 1. And when I say KERI ACDC version 2, let me be very specific about that. It's actually version 1.1 of the Trust over IP specifications that the KERI stack working group are working on. I'm calling that KERI 2 just for the purpose of this presentation because if you look at the Python reference implementation in GitHub, there's something called the main branch and the main branch is KERI 2. I spend most of my time in keripy, when I say KERI 2, it's really 1.1 of the specs. Just sort of keep that in mind. – And that's confusing. – Okay. How would you like me to say that would be what would be the least confusing way for me to describe it? – No, no, no. Absolutely call it KERI 2. I just… I also spent a bunch of time trying to figure that out.

Like, the thing that you're explaining is super confusing. – It's because KERI as a code base existed for several years before we started the standards track. We had already advanced well into the one point whatever you know and the spec actually incorporated a bunch of lessons learned, so it was already anticipating you know KERI 2.0 but it was one point of the spec. – Maybe you can use the artist formerly known as in spec itself right? – 1.1 formerly known as KERI 2 – Yes, exactly. I'll state that for everybody to hear. Spec version 1.1 formerly known as KERI 2. […]

[…] The TL;DR is that did:webs works great with KERI 1 and I was fortunate with GLEIF to be able to actually fully integrate did:webs with ACDC and a W3C presentation. We know that it works all the way through end to end with KERI version 1. There is a significant amount of work that's non-trivial to do to make it work for KERI 2.

As a part of this presentation, we'll get into later the differences between KERI ACDC version 1 and KERI ACDC version 2. Before we get there, we do have some slides that say what next, where are we going to go with this? Just keep in your mind everything works with KERI 1 right now. And I know it's

version 1, version 2. I just want to delineate between the two. It's not as scary as you might think the upgrade from 1 to 2. We'll get there. You if you understand the fundamentals and the basics, all those transfer, there's just really new cool capabilities that KERI 2 brings us like graduated disclosure, Blindable State TELs that are a step above what KERI 1 brings, but everything from KERI 1 still applies. All right. Now let's get back to, with the TL;DR done, and we're going to dive deep into did:webs. Before we get into that, do we have any questions? Anybody already wants to start us off with? And if not, I'll just dive right into it. Great.

The idea is that we have the web. And if you look at so many implementations out there, remember the DID wars? There were some DID wars. There were the ledger wars. You probably saw Sam's presentation from earlier today. There were definitely DID wars. The battles were smaller, there were less fighters, but there were wars. And usually what would happen is when somebody lost a war, they just used did:web. Or if they didn't know which war to fight, they just used did:web because it was simple. And you had a did.json document you'd put on a web server. That's why I say web hosts, web host delivers the DID doc. And in that DID doc, your DID was your URL. Pretty much very straightforward to transform a URL into a DID. And then in that DID document, you'd say here are the keys they're currently authorized for signing. That's why it worked. It was really it's just a web page, did.json web page. Here's my authorized keys and here's what they can do. It's very similar to how the certificate authority system works in the sense that there's one key when it's done, it's over, it's gone forever. With did:web, what happens when you change your keys? That's when you need did:webs because did:web doesn't support key rotation. Now we actually want to support, okay, I'm going to change my keys and I want to have a relationship between the old key and the new key so that you can know that Kent version 1, 2, 3, and 4 are all actually Kent. You don't just have to take my web server's word for it. If you want to verify, you're the same person across time, that's where did:webs comes in. And when I say KERI establishes authority, the authority to say that I am perpetually one person. I'm not this multiplicity schizophrenic identity across many different keys. I'm a consistent thing over time. That's why the S in did:webs is so special. It's because it allows us to add that characteristic to the world of decentralized identity. All right.

Here's what we're going to go through. Now you sort of get the overall specialness of did:webs. We're going to walk through just an example with did:web and did:webs. Talk about what does it mean to actually resolve a did:webs DID? What are the different capabilities that are in - or need to be in the did:webs resolver? We're going to go into a little bit later how you even though keys change you still maintain control of the identifier. And then I'll get to a current state on where's the specification? Where are the libraries?

Where's the implementations people are using? What needs to be done next? Because there's like I mentioned there's KERI 1, there's KERI 2, there's significant work to be done for KERI 2 and even for the KERI 1 there's some elements of the spec.

We got some review from the KERI stack working group. I should say that I am one of the current editors and co-chairs of the did:webs spec and the did:webs sub working group instead of the KERI stack working group. When we finish a version of the spec, we pass it on up to the KERI stack working group chairs of which Sam is one and I believe Philip Feairheller is another. It might just be those two.

We got our version of the spec ready. We sent it to them and they gave us some review points. There's still review points on KERI 1, which is what we'll get into a little bit later. And then there's a bunch more KERI 2 work to do. And by the end of this presentation, you'll know where the reference implementation is. You can go start to play with this if you'd like. As I mentioned, did:webs provides us a base of DNS, a

predictable location for your did.json A URL is that predictable location. Did:webs doesn't change that. Everything you know about how to use did:web still applies in did:webs. It's just that we take and we add another file and we say here's a keri.cesr. It's a set of bytes that has a set of KERI objects expressed in the encoding called CESR. That's the first time seeing CESR. CESR is a custom wire protocol for how you express KERI data structures, KERI and ACDC data structures. We take this extra file and it's got a lot of data structures in it that allow you to provide verification of, okay, I'm the same Kent over time. I'm the same person. I have all these things that cryptographically verify to say I had these keys today. I rotated to the these keys, the other keys tomorrow and so on and so forth. The keri.cesr stream has all of that data plus a couple of credentials as you'll get into later.

And those credentials I mentioned, they go directly into the DID document and you'll see that a little bit later because all you really need to do is you need to say I have a KERI identifier. That KERI identifier has keys. And I also have credentials to say which domains are authorized. Once you know how to process the appropriate KERI keys and KERI credentials, you can derive a DID document as you'll see in just a couple slides. All right. When your keys change in did:webs, you just replace the published keys. You do keep the same DID because the DID is your URL. But you just change the keys. You throw the old keys away. You've got some new keys. There's nothing in did:web that says these old keys relate to these new keys. You just sort of have that's implied. It's not explicit. It's implicit in the sense because you use the same URL and you just transform the URL to a DID. KERI says, no, we're going to be explicit about that.

We're going to verify the authorized key transitions for the same identifier, which is what AID is. How do you verify that? That's where the key event log (KEL) comes in. And what this means is that we're not just replacing published keys. You're getting a new pair of keys and the old pair of keys we're saying we've rotated away from those. The relationship between the old and the new keys is explicit.

And it would really this top bullet point is a sort of just another way to say the same exact thing. Even though you're going to get the just from a regular web server that did that JSON, you actually when I say controller evidence right here, controller is a term if you see that for the rest of the presentation, we use controller to refer to a person or an entity that controls a KERI key store.

I'm just saying give me your KERI evidence and I'm going to use that to verify what you're telling me in the DID document. A couple of purposes that you might use for did:webs are similar to what you just need identity for security for in general.

Let's take this ACME corporation. They're going to host two things at example .org, they might want to be able to take the same identity as a company. GLEIF solves the problem of organizational identity. We're going to model a little bit of an organizational identity problem here, but with did:webs. What the vLEI brings and this isn't a vLEI presentation, but I'll give you a quick intro is we say, well, we've got this entity identifier that says this is Microsoft or this is Google. And then there's the vLEI that says, Okay, well, this is cryptographically verifiable based on a key ecosystem that we've a trust ecosystem we've established to say this really is Google. It's not an imposter. You could consider that to be kind of like this. You want to host some content. You want people to be able to say this is who you are. You might want to have content in multiple places. And I call it a backup here, but you could really think of it as anywhere else in the world that the same company might want to either release new products, host other content. You need to be able to say you have a consistent identity no matter where your content is hosted. Part of what you'll see through the rest of the presentation is that… We use something called an AID. Is there anybody here that doesn't know what an AID is? Okay. Thank you. An AID is something called an autonomic identifier. It's what the KERI world describes. It's just an identifier. It's there's a

couple of cool properties about it that bring in the name autonomic. But for this example, it's just your organization ID. Your organization, let's say you're running an AI agent, which is sort of where this Ops ID comes in. I said delegated identifier here. That's another thing that's did:webs supports, but it's really based on the KERI support. You might want to take your identity as a company and say, I do all these different activities. Maybe I'm going to delegate this one to a person, this one to an AI agent or so on and so forth. And part of what I've got here on the slide is to say, okay, if you're running an organization, you're going to have lots of different concerns at the organization and you want to have consistent coherent identity across all of them. The foundation for that is KERI ACDC. What we're showing through the rest of the presentation is how we're using did:webs on top of that to provide a standard interface through these decentralized identifiers, which is a the W3C standard to allow you to have a consistent identity across all these concerns and where we're going to the biggest part of the work that remains is actually in this fourth box right here. I see credential verifier checks issuer proof and part of what I wanted to say up front as the TL;DR — a big part of the work is that just because we've got nice things in the KERI world called multisig and nested threshold weighted signature multisig doesn't mean that all verifiers that you consume that. You might have heard from the W3C world, you might have heard of something called the data integrity proof specification. Anybody heard of that? If you know data integrity proofs, well, it helps your verifier know to how to use the right keys to verify a credential signature. Is that correct? We have to do that. I just want to thank you for being here so I could validate that. I was pretty sure about that when I looked into it in April, but thank you. We have to write a similar thing for the did:webs world. And this is a significant piece of work that we don't quite have time to get to.

KERI has this amazing capability for multisig. I said nested weighted threshold multisig. Just because you can sign a document with this really fancy grouping or ordering of signers that can be resilient, fault tolerant and everything you want doesn't mean that the verifier is smart enough to pull all the right keys out of the DID document because you're going to define what's called the verification methods and I'll just let me just skip ahead a little bit. Here's a DID document. Your keys are going to be defined in your verification method area.

And then you're going to say what they can do with the assertion method and you're going to say which ones allow you to authenticate the DID document in your authentication. Just because KERI can produce really in did:webs really fancy DID documents that represent our conceptual domain model doesn't mean the verifiers have that domain model. When I'm talking about that fourth box that I mentioned. Even after we add version 2 support, KERI version 2, there's actually one final piece we need to add beyond that to call it a complete package, which is we need to write a spec and a library so that if somebody just decides to use a fancy multisig group and put all their keys here, well, the verifier needs to be able to detect that it's not just a simple key, but there's a lot of things involved because you don't want verifiers verifying credentials that just have one valid signature you need to be able to interpret what a multisig nested weighted threshold multisig group is. It's pretty straightforward for people that know KERI, but what we need to be explicit and clear, which means we need to write a spec and then write a library that'll do that. I'm seeing at least one nodding head in the audience. Makes me feel like I'm not a total failure. That's great. Anyway, all right. This is an example of a did:webs DID. Let's just walk through it slowly.

Of course, you've got the protocol method at the front did:webs for domain which we'll get into the designated aliases ACDC later and you've got a path. You might have a bunch of different paths here just like a traditional web domain.

And then on the end here is that thing that I said earlier is an AID, autonomic identifier. It's a KERI AID. And one way to think about what did:webs adds on top of KERI is we're using the W3C decentralized identifier standard as a standard way to discover keystate. That's really this… If you had to say KERI in one sentence, did:webs is a standard is a W3C standard way to discover keystate. And when I say keystate, I mean current or prior history for this identifier. Because remember I said key rotation, you could have lots of rotations and lots of history. That's really what we're doing here. And you'll give something that looks like this to what's called the DID resolver and that DID resolver will go and it'll ask for two files as you'll see later and it'll say, okay, tell me what the current keys are and who's going to do this. Let's actually put our minds in the mindset of who's using this. Let's say you're at a gas station and you see this young looking person come into your establishment. They want to buy some tobacco and you're in the state of Utah. You're at a place called Maverick. Maverick says give me your driver's license and you give me your digital driver's license and it's a SEDI one but for whatever reason their verifier can only accept W3C things and so they use the compatibility layer, right? And they've got a SEDI identifier. Somebody has a W3C compatibility layer and they've got a credential that's issued that has a did:webs in it. How is the verifier going to go and find the right key to verify this credential? That's what we're going to learn about through the rest of the presentation. But the idea is that they look at the history for this identifier and say, okay, what's the current key right now today?

I mentioned two files. What I wanted to say with that example is that the who of who's who cares to receive this did:webs is a verifier. When the verifier says, okay, are you authorized to buy what you're trying to buy? They're going to find two things for your identifier. They're going to get your did.json and anybody who's familiar with the DID ecosystem knows this pretty well. And I said did:web form right under it. Because did.json, it's when we when it's actually sitting there on disk or when we retrieve it, it's actually just a regular did:web did.json. But we have another file that's not did:web. That's what this keri.cesr stream is. When they verify or they're trying to make a decision, should […] or not, they go and they resolve both the issuers DID and your DID and they get these two files for the issuer and for you, this is what we're talking about as far as verifying a credential. There's a W3C credential that's been secured with a KERI identifier. And then we go through this process. We put them both together and we essentially say from the keri.cesr stream, can we derive the exact same document that was hosted over here? If we can, we at least know it from that point, it's coherent and it's cryptographically verifiable as far as what you've told me is honest. There's another question that we're not going to get into in this presentation of, well, what if verifiers want to use witnesses and watchers? What if they want to use registrars and observers? That gets beyond what we've decided to solve with the current version of the did:webs spec. And here's why. Does the DID world really care about our special KERI infrastructure? They don't and they can't. And you can make an argument that they shouldn't. What they're limited to here, because here's the thing, as soon as you start to become KERI aware, why use did:webs? As soon as you become aware of the conceptual model of witness, a watcher, a registrar, observer of all of that, and the adjudicator, if you're going to pay that conceptual burden, you can just use straight KERI identifiers. We already have a discovery protocol built into KERI. It's called out-of-band introductions (OOBI). We send everything across the wire that's necessary. If you're already going to pay the architectural and conceptual cost to upgrade all of your infrastructure to be KERI aware, just use KERI. The use case for did:webs is for verifiers that never, that probably have something against KERI, to be quite honest. They say, we don't want to hear you say it. Don't talk about it. We know it's behind the scenes, but don't bring it up. – There are some of those. – There are some of those. – Some big ones, notable ones that have said almost exactly that. – We're like, okay, fine. We'll call it did:webs. We will never use the word KERI. Just don't read the spec. But no, if you take an economic argument, you

could say there are some people that don't want to spend the resources or don't have the resources to spend to get full infrastructure support for the I in KERI. Or they've got a really good reason why, hey, this for the risk profile here, we can accept these risks. Just a plain did:webs, that's fine. What I'm leading to here is that you're doing a check by taking the did.json and the keri.cesr and saying, are they coherent? Can I derive my did.json from my keri.cesr? If so, at least that's not lying to you. And if that's enough for your verifier to say, yes, you can take this tobacco out of the store to the young man he's trying to buy it, that might be enough. You send that all into the DID resolver. You do that for the issuer, do it for the holder and say does it verify? And if so, great. Now you can verify signatures on a credential.

Now I'm going to have this slide because I've had questions come up from members of the community that didn't know about this fourth thing on the slide. I'm going to walk you through it. This is a little bit deeper. If you're new to KERI, don't worry too much about what I'm going to say here or if you need to do did:webs. This is maybe a little bit deeper. Okay, this is the keri.cesr. If you want to go inside that file, that box from the last slide of the … I said you can derive the DID document from the keri.cesr string. And that's actually what the did:webs resolver does behind the scenes. What's in the keri.cesr string? Four primary kinds of content. There's key event logs. Here's my identifier. Here's what keys control my identifier. There's transaction event logs. Remember I said KERI 1 and KERI 2? In KERI 2, we have a new kind of transaction event log. But in KERI 1, we've got what's called a registry TEL and an ACDC TEL. You're going to have one of these for your registry for your credential and you're going to have one of them for the credentials lifespan of […] like issuance and revocation. And you're going to have some ACDCs. Now, why do you have ACDCs in the keri.cesr stream? It's because of a feature called a designated alias. Right now, this is actually something that makes KERI special and distinct. It's the big fake fiberoptic S we had at the beginning of the presentation. This is part of what makes did:webs special is that you make an explicit commitment to say, I'm authorizing this DID on this domain. There's no other DID method in existence that does something like that. That's as portable and decentralized as KERI. You might get a close approximation for DID methods that are on blockchains that have something like this, what I'm hinting at is one of the rules for did:webs is that you're not authorized to… a verifier should reject your DID document if you don't have an explicit commitment to example.com or to whatever your domain is. The reason why you have ACDCs is because it's a 100% requirement that if you want your DID to be hosted on a given URL, you have to commit to that URL with an ACDC. And what that implies, that one requirement implies that every did:webs implementation must have full support for KERI and ACDC. There's no way to get away with it. This is the reason why you have to have it all implemented for a did:webs implementation. Okay, now the fourth part here is a little bit more surprising because you're like, okay, it's natural that we've got key event logs. It's natural that we've got registry or transaction event logs. It's natural that we have ACDCs. What's this rpy? That's actually shorthand for what's called a reply message. And we use them and KERI to describe a couple key records.

There's like a URL or an endpoint role authorization. And what that does is it essentially says this person or this controller, this person that controls these key pairs, this identifier is allowed to act in this capacity on this URL. And we include that in the DID document because you'll see there's the services section of the DID document. They're going to be able to look at okay there's a witness or an agent or a there's one other I was going to say watcher there's actually one I'm thinking of… a delegator! Okay. You might see a couple of different services and you want to verify okay is there actually cryptographic commitment and authorization by the controller of this identifier of this endpoint and in KERI we could do that. In did:webs we can do that and we expressed that in the keri.cesr stream with reply messages.

Now, these are not anchored to a KEL. They are just signed reply messages. And a question came up in the KERI community. They said, well, is everything in the keri.cesr stream, is it essentially just a… This is part of the aspect problem we have to fix that I've got later in the slides is they said, well, should a keri.cesr stream just always be a subset of a key event log, whether the whole thing or just a part?

No. And this is why you have to say, well, if I'm going to have a services section of my DID document and to say where things are at, there's a way we express that in KERI and that's with reply messages. All right. Now that's about as deep as we need to get. Those that are new to KERI, the rest of the presentation shouldn't be as brutal. But we'll see. I don't want to make any promises. All right. We've talked through this. Again, we fetch both the files. We verify that they're coherent and as I mentioned, there's designated aliases and that's where when I say authorized DID names here. We go through the verification process and we say, okay, is the URL that you're hosting this did:webs DID document on, is it authorized? If it's in your designated aliases ACDC, it is. And once we check that all that's coherent, we essentially just derive the DID document from the keri.cesr stream and compare it with the did.json that you've got on your web server.

And if they're the same document, you verify. You're good. We already saw the slide. We won't spend too much time on it. This is just an example of what a DID document looks like if you haven't seen it. I've mentioned there's one thing that I should mention on this slide is I mentioned the concept of a designated alias and I'll just slowly talk through that again. A designated alias is what I was just referring to as this is… no other DID method in the world does this.

KERI makes an explicit commitment to a URL. That URL… I was going to say it goes in the alsoKnownAs that some of them do. Now I might have jumped ahead of myself here. Some of your designated aliases will be placed in the alsoKnownAs section. Let's say you're essentially trying to tell the world: this DID document or the location that I'm storing this did:webs DID document isn't the only location there is. There's a couple of other locations you can get the same DID document, but just under a slightly different URL, but it's the same controlling identifier. You'll put those in the alsoKnownAs and those come from your designated aliases ACDC. Now to the multisig stuff. We do have multisig support and I'm going to say limited multisig support and you'll see why.

There is something that exists in the W3C decentralized identity ecosystem called ConditionalProof2022. And it lets you define a threshold structure for signing something. It was a natural fit for did:webs. Did:webs, based on KERI with KERI multisig. We just needed a way to define, okay, well, there's a… let's say it's two of three. Like this example, we've got three keys that can sign. Two of them are required. ConditionalProof2022 lets us define that sort of a simple threshold structure, but it gets deeper than that.

ConditionalProof2022 also allows you to do nesting. The really fancy cool nested weighted threshold multisig stuff that KERI could do, you can actually express that with ConditionalProof2022, which is actually what we do in the did:webs resolver. But the way that it works is it's how many people here are new to multisig in KERI? Just a couple. All right. I'll walk through that one a little bit slower.

Let me introduce this from why would you actually use this? And I'll mention a GLEIF example. We have defined in the ecosystem governance framework rules for different participants in the ecosystem. And it's actually useful from an operational standpoint because let's say you have a group of three employees who's responsible for a role at your company and that role is to manage a certain identifier. What if one of them is gone on any given day? What if they're sick or they're out or they're unavailable? Do you want to require all three to be at work to be able to sign an event? You might not want to do

that. And there's other lifecycle reasons why you might say, well, let's say somebody gets fired or they die or something. It can be really rough if you say, I've got, okay, I need three of three signatures. You might just say two or you could say four of seven or two of nine, however you want to define your threshold structure, there's actually a lot of business continuity reasons, failover, redundancy, reliability, why you would want to use a threshold structure and the way that this works from the KERI level and then we'll cascade out to the did:webs level is you say, I've got K1, K2, and K3. As long as I have enough signers, okay, this credential is valid or this issuance has happened. That's what multisig is. Now, what did:webs does is it says, okay, for a verifier that doesn't care about KERI at all. If they want to verify that this is a credential, let's say this is a W3C credential that has been projected from maybe something in the KERI world, and I want to verify that this issuer issued it and that this holder currently holds it. But what if those issuers or… that's fine if it's single-sig, but what happens if you've got a multisig issuer or you've got a multisig holder? Now we need to go in there. We need rules to say, okay, well, we need to be able to define the threshold and who is authorized to sign. That's what this ConditionalProof2022 does, is it allows us to define that very clearly within did:webs. With that in mind, let's actually dive into what a verification method might look like for ConditionalProof2022. You're essentially saying and when I say K1, K2, and K3 here, if you're familiar with DID documents, you're familiar probably with the verification method section or the authentication or assertion method section, that's where this shorthand is referring to. There's going to be a verification method that says this particular key is K1, K2, or K3. But what you're looking at here, this object here, this set of text, this is what a ConditionalProof2022 looks like. It says, okay, I need two signers and here's the people who can sign. And I mentioned the authentication and assertion methods of the DID Doc. There's a lot of rules to remember. Especially if you want to as a verifier be able to support the coolest features of did:webs, which is multisig issuers and multisig holders.

And we just this just walks a little bit through a waiting signing threshold thing. We can sort of move on. There's plenty to discuss later. All right. I mentioned remember the four things that were in a keri.cesr stream. And I said you need to… it's not just a key event log. You've got service endpoints. You've got role authorizations. That's what we talked through on this slide. As you might say I have a witness or I have a mailbox or I have an agent and that has a relationship to the identifier that is giving you this DID Doc. And the way we represent that is not with a key event. There's something called the location scheme. The `/loc` and `/scheme` is the name space that we give it in the KERI world. We say there's a couple of data structures we use to represent these things when they're given into a DID Doc. Now, you might ask me, well, why are we including these in the DID document at all? Because I thought we said that we should assume that the verifier doesn't know KERI at all. That's true. And really what we're doing here is we're piggybacking a little bit on the structure of the DID document and say, well, maybe people might be interested in discovering that I have a witness or that I have a mailbox or that I have an agent. This straddles the line. It's a potentially useful construct. It starts to straddle the line. If we're not pulling anything from the witness or from the mailbox or the agent, maybe we don't actually need this, but it seemed invaluable and we kept it in there. But it's debatable that if you know how to talk to a witness, if you know how to talk to a mailbox and an agent, you'd probably just do that directly. Maybe you wouldn't need did:webs, but we've defined it in the spec in case it becomes useful.

Okay, I won't get too far into the delegation because then again delegation is also a KERI-specific concept. This slide is really just a repetition of what I've told you earlier. I said if you want to support the fancy multisig stuff from KERI, we need another spec.

This is when I say credential proof right here. When I say a credential proof has to use an authorized method, where that's valuable is before we let the kid go out of the store with the tobacco and we need to verify the right state of Utah issued this and this kid actually holds it. There's rules you have to use to say, okay, I can verify the signatures on this credential because they can't come from these places in the DID document. And we could walk through this example, but essentially when I talk about the assertion methods, the different proof formats that we have, it's essentially we would walk through, I think that's probably not a good time to walk through the nitty gritty of what that proof spec would be like, but essentially know that we need to write a proof spec and this slide gets a little bit deeper into that. Oh, you want to go? Actually, if you want to go there, we'll go there. – I have a question about the proof purpose. – Perfect.

Okay. – Is there a KERI data model equivalence either in the event log or any of the delegation mechanisms for assigning proof purpose to keys to not the pre-rotation keys but to regular document keys. Can you say this key can only be used for authentication, this key for signing stuff and this key for key agreement?

– Is the question is there something in the KERI data model that would support deriving that from the KERI data model? – Yes. That's the question. – Can I answer that? Signing keys can only be used to sign. Rotation keys can only be used to rotate. And most of the time when you're doing something of high value, you anchor in an event, which means it must be signed with the signing keys or you anchor it in a rotation event which must be signed with the rotation keys, the pre-rotated keys. There aren't any other places that you use keys for any other thing in the whole KERI ecosystem.

Signing is general. You can do anything you want with signing. Right? You could say, well, if I want to sign a particular type of packet, I suppose you could say that, but we don't have another set of keys, right? And then witnesses have their own set of keys. But witness keys are only used for receding. We define keys implicitly for the purpose they solve. We don't have keys that you assign to proof purpose because where the key appears in KERI determines its purpose. Witness key appears in the witness list. Assigning key appears in the signing key list and the rotation keys appear in the next rotated key list. Those key lists determine their proofPurpose. There isn't a reason to have… I never came up with a reason to have some generic semantic thing for keys because in general, I don't believe that keys are very useful. Bare signatures are highly insecure. They're not very useful. You only use them for ephemeral things. And proofPurpose seems like something that's not ephemeral. – What about the ability to designate and rotate key agreement, so encryption keys, which are currently derived from signing keys deterministically in the KERI ecosystem.

Would you be open to having that? – Yes, that's a really good question. KERI doesn't do encryption. KERI is all about authentication. Now the Trust Spanning Protocol does do encryption. There's a need to define how you generate your encryption keys. That's not defined in KERI.

Encryption is a different protocol from KERI. Yes. – Is that a potentially yes? – That's a potential yes. – Good question. Now we've been getting a little bit deeper into the technical depths. I like going there. I know we've got sort of our time is getting shorter. What I'd like to do is look get a little bit more towards the status. We're going to go pretty quickly through the rest of the technical stuff. I will be at IIW. I'll be here for the rest of the week in case anybody wants to go deeper. All right. We talked about a little bit about the DID document. We won't go too deeply into there and historical keys. Let's actually skip ahead a little bit. We already mentioned that hosting can change. The identifier stays the same. If you go from your DID from example.org to backup.example.org, you can have the same identity and you can cryptographically verify that it's the same identity or person who controls that.

All right. We already talked about sort of threshold identifiers and we talked about we don't need to get quite into duplicity detection here or some of the other problems here. We're going to skip over. Let's go right to where the spec is at.

Okay. The spec we're on version 0.10.3 of the did:webs spec. That's where we're at today. We've got a number of open items. We've got about 17 different open items that we need to handle and we thought we had a normative baseline around August 11th. We've actually been working earlier this year with the DIF recommended DID methods to have did:webs become a recommended DID method. We're not quite there yet. And as I mentioned, we've still got the KERI version 2 stuff that we need to add in. And there's some review points as we'll get into later. I'll just add that we need to provide clarity around transferable and non-transferable identifiers. There's a couple of other things, but there's a little bit of the timeline I became aware of the fact that if this is actually going to be used in production for a consumer who wants to verify multisig issuers and multisig holders, we definitely will need to write that credential proof spec and write a library. Right now, everything's usable with simple multisig and with single-sig.

It's pretty straightforward to have a single-sig issuer, single-sig holder. It's because it's just one signature and just one key. And when I say simple multisig, I mean with not nested weighted threshold stuff. You can have a threshold, but if it's just a flat set of multisig members, that's fine. That's easy today. It's just the challenge is that verifiers don't know how to interpret the nesting, is the challenge. And if we're going to have a way for the verifiers to interpret the nesting, that's a decision we have to say, do we want to have the rules for that interpretation of that nesting, the did:webs spec? Probably not.

did:webs should be about keys, right? Not about credentials. There should probably be a separate spec if we say this is going to be a spec for verifiers that want to interpret W3C Verifiable Credentials Data Model. They're going to verify that credential with keys that are defined by did:webs and KERI for a W3C credential that has multisig issuers, multisig holders. That's more of a credential proofing spec. And it's not just a spec. You'd want a library as well. That's easy to use. If somebody decides they want to use this in production for a really advanced multisig, that's some work that needs to be done later. But where we're going though is we want to finish out the diff process. And to get to where we really do want did:webs to be a fully accepted, fully supported DIF recommended DID method. And we were pretty close earlier this year.

– Is the spec being approved at Trust over IP, a prerequisite for DIF to recognize did:webs as an acceptable DID method on their list? I'll repeat the question just so we can all hear. It's a good question. Is ToIP approval of the did:webs spec required in order for the DIF recommended DID methods group to put it on their recommended list. As far as my understanding, it's not a hard constraint. It would help. But the more important thing is that we have two implementations and that the DIF recommended DID methods team believes that it's stable enough. Now, to be responsible, I need to say I don't know for sure. I need to talk to Otto and help with the other people.

Jonathan, if Jonathan were here, he could say. – I'm pretty sure that they don't require it because they're approving it based on DIF. It's a DIF standard. Whether or not it's Trust over IP approved matter doesn't matter. It's just whoever it is that's working under the aegis of DIF has to follow the DIF rules. And that's you and Jonathan. And all you're doing is leveraging the task force at Trust over IP to get it right. – Right. I will say that on the criteria like you mentioned from DIF on the rubric that they give us to

evaluate how ready we are. There isn't a criteria for whether or not the spec is approved. – that's what I say. DIF allows approved specs from just anybody that wants to contribute it under DIF. It has its own approval process. – Got it. It sounds like the answer is no.

ToIP isn't a gating effect on it. – Okay. And the idea, the goal is that we want to be just like did:webvh and did:webplus and there might be a couple of others now. We do want to be a DIF recommended DID method. It just looks good. Once we have the KERI v2 stuff in there and all the open points addressed, I don't see why we wouldn't be. All right. There I mentioned some review from the KERI stack working group chairs. I say Sam's review up here. It was his particular comments. We've got four on this slide, three on this slide. I broke them down into a few different categories. We don't have to go into the individual depths of what these review criteria are because that's my job and I won't force you to go through that. Just be aware there's a significant number of open points that we have to clear to get the spec into the next version and it's generally around… some of them are actually about… We need to show examples. Okay, what is an ordinary did:webs consumer receive? We need to walk you through, okay, what does a W3C credential proof looks like? And we're going to add that at least for the simple multisig and single-sig walk people through examples so that you can by the time you're done reading the did:webs spec, there should be enough examples in there to be able to give you a pretty good illustration of how do I actually use this? We're going to add some examples there. And this is a harder one. The nested thresholds like I've been mentioning. We do have the recursive mapping but we need the worked case. We need to show here's if you're going to do a nested threshold multisig actually a worked case of it done. If we do the most complicated one, the other ones are piece of cake. Okay, we've got… there's other things that need to be… other open points, but the big one is really the keri.cesr ACDC version 2. We need to be very explicit about things like what happens when you have a KEL that begins in version 1 and then we have some version 2 events. How does the stream declare its supported protocol versions? And CESR and KERI 2 already have a way to declare protocol versions, but we need to be explicit about that in the spec. We need to acknowledge that yes, there can be a version 1 stream or a version 2 stream can be partial.

There's a couple of things that we need to go in there and just clean some things up and be more explicit about. And then the registry state is going to be a particularly interesting one in KERI 2 because we have a new kind of registry and it can be blinded. And what about unblinding life cycle of the registry state? We just need to be more clear and explicit about everything that KERI version 2 brings into the picture. Okay, there's an interesting discussion…. Since we have time… maybe let me actually jump forward a little bit. I get a little bit more into some of the other open points, but it's probably would be belaboring the issue here. What I think would be what would be valuable is to show the existing repositories, show you where things are at, so that you can go play with it yourself. Right now the if you look at the GLEIF GitHub, we've got GLEIF IT, we've got the did:webs resolver. That's what this 0.3.7 is right now. That's the current version of the did:webs resolver.

That's the reference implementation. There's one old pull request to it right now, but we'll just go ahead and skip beyond that. I'm going to skip forward a little bit to show you some resources. Okay, we do have the Trust over IP specifications. I mentioned there's the did:webs resolver. There's a couple of other things that are worth mentioning. When you actually build did:webs into a wallet or a verifier, you'll want to use this did:webs-ts library. That library is not a implementation of did:webs and TypeScript. It's just when you are as a verifier, you are verifying credentials. There are libraries, resolver libraries that we use to resolve key state. That's what this is useful for.

It's actually useful for from TypeScript resolving key state. You're not generating it, but you're pulling it in as a part of standard DIF infrastructure. And then as a response to the DIF recommended DID methods work group earlier this year, we presented to them the Python implementation. And we were pretty much getting to the point where they said, okay, well, we will accept did:webs as a DIF recommended DID method. And then they said, does it have a second implementation?

And I'm like, no, it doesn't, but it will soon. I started on a TypeScript implementation. I tried to keep it as minimal as I could for as long as I could, but I did start a TypeScript implementation of KERI, ACDC and did:webs just enough so we can sort of get through that process so that when they ask us that question, we can say yes. Personally, I don't think it's fair. I don't think it's fair to have that as a criteria because there's it doesn't take into the complexity of the different DID methods is completely different. Like if you look at the lines of code that are required for some of the other DID methods, it's in the thousands. Whereas a KERI implementation could be tens of thousands closer to a hundred thousand. There's a lot more complexity going on. Personally, I understand people's desire for a second implementation, but to a truly secure complete implementation of KERI ACDC is non-trivial. It's a big deal. I'm going through the process just enough so where we can get through if that actually becomes a hard blocker. We can be unblocked by that, but that's what this second line item is. And if you want to go to a place to learn a little bit more about did:webs, this hasn't been quite enough where you want to learn more details, you can go to didwebs.info. That's the website actually. This GLEIF IT on the front of it, that's just where the repositories hosted. But there's a informational website you can point your AI agent to at didwebs.info. Some other work. Go ahead. We got a question in the back from Phil.

– One of my primary interests in this is being able to do DIDComm on top of KERI. I'm not actually even sure what question to ask, but I mean, is it the case that I can create a did:webs from any KERI anytime? Do they change as you rotate keys? What's happening under the covers? I was a little late, maybe you already answered this, but… – That's a good question. How do we support? I'll just repeat the question for the benefit of everybody. How do we support DIDComm with did:webs? And then there was a second question there as can we turn any KERI identifier into a did:webs at any time? Yes. And there's one decision you have to make and then one additional decision. What domain are you going to use for the identifier? And the reason is after you make that decision, you take whatever that domain is, put it in what's called a designated aliases ACDC and you self-issue that. That's the only other decision you have to make. But other than that, the answer is yes, any KERI identifier can be turned into a did:webs DID, whether it's transferable or non-transferable.

And as far as support for DIDComm, I'm curious too. I've never quite done that. I haven't looked quite deep enough to be able to know enough about DIDComm to answer your question. I would imagine. If you just think about the inventory of what KERI gives you or what did:webs gives you. You've got a full key event log, you've got keys, your keys can change, you've got consistent identity and did:webs is really just a discoverability protocol in that it's a DID document to say, well, what's my latest key state? Without knowing anything about DIDComm, if DIDComm needs to look at a DID document to get latest key state to do communication, did:webs could fully support that. – I had a conversation at lunch And what a lot of people don't understand about KERI is that KERI normatively can be used in a very simple way. A normative KERI AID can be non-transferable, use the term, means it does not have a KEL. It has at most an inception event, but it's a vacuous inception event, which means a non-transferable KERI identifier is equivalent to did:key, which means you can use it with DIDComm. It's no harder than using

did:key. It is KERI. It's full KERI. It's fully normative KERI. It just doesn't have all the security benefits of a KEL. But it still works. It's still KERI. A KERI stack will recognize it as a KERI identifier and treat it appropriately.

– What's cool about the non-transferable identifier is you can essentially derive not just the inception event, but you could derive if you knew what domain you wanted to commit to, you could derive a DID document just from having a non-transferable identifier. – you just don't get the security benefits of a KEL.

It's did:webs, did:web that's no weaker and no more complicated than the did:key did:web. Just to repeat that for anybody that might not have heard it. If you're familiar with the concept of a non-transferable identifier in KERI, non-transferable means you can't rotate keys. And you might say, well, why design a key rotation protocol but not be able to rotate keys? There are certain instances where there's a lot of other things you get from KERI. There are a lot of instances where you don't need key rotation. For example… – Witnesses have non-transferable AIDs. They don't need KELs. The witness itself doesn't need a KEL because its KEL is implicit in the KEL of the controller. The witness's AID is its public key and it's a key in the KEL of the controller. It's rotated when you rotate witnesses. It doesn't need its own KEL. It is trivially just a key pair like did:key. We use them all the time in KERI infrastructure. Whenever we need just a bare key pair, we can use it and it's still a valid KERI AID. – And the interesting that we're talking about this because some of the open points on the spec are to make sure that we make exactly this clear in the did:webs spec is how the utility of non-transferable identifiers with did:webs. And I imagine… – And then there's another step. You can use a transferable identifier, but use it in indirect mode. It's transferable, but it does not have a witness, right? Which means it's equivalent to did:peer. You can do did:peer with KERI just like did:peer. It's essentially did:peer, but it's full KERI normative spec because it doesn't need witnesses. It has a key event log. It's better than did:peer. It's more secure, but it still doesn't require anything, any more infrastructure than did:peer does. And you just use direct mode to exchange your KEL.

You don't have witnesses to publish your KEL because did:peer assumes that you can communicate directly. If you can communicate directly, I don't need the witnesses for high availability. I also lose the other things that witnesses give me. But if I don't care about those things, which I don't if I'm using did:peer, then I just use KERI in did:peer mode and it works just fine. KERI does a bunch of more things… simply they're sort of throwaways because it's like, well this is easy we can do that. – I have one question. Maybe I just misunderstood.

What I'm primarily interested in is I've got a million people out there with a system based on KERI and I want to be able to enable secure messaging through DIDComm. – And you can do that. And that's one reason for did:webs is so you can bridge to any DID world because the worst case they'll just use it as a did:web and not the KERI stuff. They don't have to have KERI on their end because they'll just say, well, I don't understand KERI.

Give them the did:web. – I understand keys and I can get the key from the DID doc and I'm good. – I will say that I've been told by the staff that my presentation is done and my hour is up but I will be here at IIW and you can get me after this as well but I want to make room.

TALK II

2026-11-02-miniconf-promo

KERI Mini Conference November 2026 · promo · 00:51

What happens when you put the people building KERI in one room for a whole day? Hi, I'm Joseph Hunsaker. Some of you might know me as seriouscoderone. I'm the operational organizer for MINIKERICONF26. Which means if the coffee runs out, it's on me. Monday, November 2nd, the Computer History Museum, the day before IIW.

Keynotes, workshops, lunches on us. And the people building this stuff right there. Ask them anything. Okay, here's the part I love. Your tickets come with 20% off to IIW. You're already coming to Mountain View, right? Might as well. Just come one day early. It's already shaping up to be a great day. I want you to make it better. Come bring your hardest questions. Come argue with us.

TALK III

2026-11-02-ryan-hansen-evan-asakawa

KERI Mini Conference November 2026 · talks · 52:26

SUMMARY

AI generated summary.

Ryan Hansen and Evan Asakawa demo the KERISuite stack across multiple machines — witnesses, watchers, and related services as separate instances. Using Locksmith, they create a vault and identifier, install the browser plugin, and provision witnesses via a boot server. Graduated disclosure is in the infrastructure but not yet the wallet; the vault mailbox is unfinished, so the credential request demo does not complete end to end.

Okay. This is a little bit of an overview on what we're going to present. Some of you may have seen this or some that are watching may have seen this. I went through this in our first KERI conference. It's a little bit geared toward the infrastructure as it was back then, but I'll kind of update on the fly as we go through this to where we are now. We already showed you these repos that we'll be dealing with. There's a setup guide out there which I mentioned is a little bit outdated. This is geared toward running an infrastructure on one host and in a demo sort of way as opposed to what we have now which is four different hosts to spread out the services. There's the QR code to the… oops, sorry, if people want to follow along with the setup guide you can hit that QR code and pull it up.

This is a little longer now than it was originally. Core concepts here — I Obviously we've talked about KERI being all about keys, key events. I'm going to kind of jump through these fairly quickly till we get to… Let me just jump to the more important stuff. Okay, as Sam showed this one earlier, this is the KERI ecosystem, kind of the basics of the KERI ecosystem. What we're going to go through is kind of a setup guide of how we actually set up this infrastructure and what exists out in DigitalOcean right now and how you could actually deploy this kind of a stack yourselves. Any of you who are more technically inclined or want to get into some of this stuff. It's not terribly technical but there is some more technical detail for those who are interested in those kinds of things. The core components, we have our boot service, as I mentioned, we have witnesses and we have watchers. These are a good way to remember that. Just watch all these movies and then you'll remember that.

If I have questions, should I ever watch the movies? Yes. just go back to the movies. This particular guide is very opinionated, meaning there are many ways you can go about doing this with those services. You go looking at the code itself. There are lots of ways you can go through this and deploy it.

But this particular guide has an opinion of its own about how we've set it up. It doesn't mean this is the right way. It just means it is a way to get up and running. These are all Falcon apps, just so you know. It's all Python code. It's not as early-stage anymore. This was several months ago when this came up. But this is really a demo only or maybe not quite to the point of a staging environment that you could use, but it's definitely not a production kind of a deployment. Keep that in mind. This is what a production deployment might look like. Maybe-ish. If you were to imagine the wallet having connections to multiple different providers of plugins in the wallet itself and each of them is connected to some sort of cloud or data center and they distribute the services throughout according to availability zones and regions and all that kind of stuff. It might look something like this. Again, this is not something we say that you should do or you have to do, but maybe that's what it would look like. That is not what this looks like. Kent.

– Is it intentional to have only one kf-boot on each data center? – It doesn't really… well, there's two actually here. Two, but… and it doesn't really matter. Actually, there's two on both. You can run it […]. You can make it as redundant or as not as you want. Kf-boot can handle quite a bit, so you wouldn't necessarily need to make that vastly redundant. But, if you want a really resilient service, obviously, you want to at least spread it out across zones or something like that. – Can I say something? In a full production environment, every service is potentially subject to DDoS attack, which means that you have to have load balancers. That's the only way you're going to be protected from DDoS attack. All of these services have to sit behind a load balancer, otherwise somebody could just DDoS them out of existence. – For sure What the plugin architecture of the wallet allows is that you could have like let's say a kerify.net, which I don't know who owns that? It might be someone in this room that might own that particular domain. But that particular one might be two different data centers that's spread out across… their entire infrastructure is spread out across two entirely different cloud providers or something like that. Whereas a KERI foundation might be just in one.

There's a million ways you could figure that out and smarter people than me can come up with the best ways to make a really solid KERI ecosystem that is very resilient and powerful. But that's not really what we're solving in the foundation right now. Base requirements for this as it is right now.

Ubuntu 24.04 is what we built it on. I think that's the latest LTS or maybe 24.10 24.04, I think. Libsodium, which is the crypto library. Pysodium, which wraps libsodium. Nginx, which handles the reverse proxying and TLS termination.

Certbot to just handle the certificates and Circus to actually manage the services, the Python services themselves. Those are the kind of base requirements for this setup. The way we set it up was we created a service user called KERI. There's no login on that one. It's just key only and it's just kind of for that purpose.

It's only meant for the service itself to run. It's not really meant to be logged into. And this is the path that we put it in. Again, this is all just how we set this one up. It doesn't mean you have to do it that way. We also use uv and Python 3.14 specifically in this case. Anybody have any questions or anything you want to say, certainly chime in because I'm going to move through this fairly quickly.

And there's three installs that you can do. Now, this is based on releases that we have not actually done yet. We haven't actually released into like into PyPI or anything witopnet and watopnet as those names just yet or kf-boot for that matter.

But you can install them straight from GitHub. If you just point to the repos, you can install them at whatever point you want to. But eventually the plan is that we'll create formal releases for those and we'll release them out into PyPI and then they'll be available under those names.

Next, the configuration. Again, this is based on a single host with multiple instances running. In the real case that we have now, you wouldn't need four of these on the same host, but you would create one of these to determine what the URLs are that will be available to Locksmith when it connects in.

And the watcher one looks very similar. Just need one of those. Launch scripts. This is for being able to actually launch the commands that run the services. Again, anybody have any questions? – just some tips for our… – Is the witopnet and the watopnet just one per process?

It's not like something that provisions many, it's just one per process. – Just one. Just one instance will run, like one watcher, for example, or one witness, for example. As many of those as you run would be… it's a one-to-one, does that make sense? Okay. Thank you.

Okay. And kf-boot's config. Again, this is all in the guide itself. If you're following along or if you want to actually set this up, all this stuff is available in that guide. This is mostly just to kind of give you some view of it in this setting here.

Just one of the environment variables to… I think this is actually out of date. Sorry. I'm going to skip this one. Let me jump past this.

Okay. And then pretty simple Circus install. And then Certbot and all that stuff gets set up as well. That's the core pieces of what it takes to actually get this stuff up and running. All that being done, apologies for the issues there, but all that being done, you end up with four witness processes, watchers, and a kf-boot. And right now I have this running.

I'm only doing this locally right now, but in this one here is kf-boot along with a witness and a watcher. This is going to mirror more how we're actually running it now in our environment in DigitalOcean. The kf-boot is running on one host with a witness and a watcher. And on all subsequent hosts, there's actually three others. They're just one witness and one watcher instance. And with that setup, I can actually run Locksmith. I'll just give you a quick view of this and we'll get into a little more details about Locksmith in a second. What I'm doing, this is a fresh clean version of Locksmith that's just starting up with as if it was brand new. And initialize a new vault first of all. We'll just call it demo.

I'm going to add… this part here, I'm actually creating my ID — my AID. And now you see there that KERI Foundation is… that's the plugin in itself. And I have it configured so that it connects to my local Docker to hit those hosts. And you can see that the connection is connected here. It's able to make that connection. I choose that AID from the list. I'm going to do a three of four because I have four instances running. And when I click this, if I had all four terminals, it'd be cooler, but you'll see basically what's going on with kf-boot on the left side here on your right and the witnesses and watcher service on the other side. You can see that this is actually doing stuff for real. It's not we're not just kind of vaporware-ing this. This is actually doing what it's supposed to do. And on this side. – Can I ask a question? This is Locksmith on either your local computer.

This Locksmith I'm looking at and I want to make sure I'm understanding what I'm seeing here. This Locksmith you're running on your machine is targeting those digital ocean machines that you've got. And it's… – No, not right now. This is my local Locksmith that's pointing to my local Docker deployment of four hosts, which is exactly like we have it in DigitalOcean, but there was an issue with the DigitalOcean environment right now.

I'm just running it locally. But as far as the actual structure of it, it's set up the same way. – Got it. It's just using local URLs? Is that the only difference? – This is all done locally here. That's it. It's a little bit canned in that sense. But as far as the code, it's all running the same code. It's deployed exactly the same way. I'm just running it on my local host here. Any other questions about that? You see we got this QR code back, which basically means that our witnesses got connected. We can go back over here. You can see all the lists of the witnesses here and one watcher that got connected to the wallet.

– What is the QR code for? – You can scan it and it'll take you to like an authenticator app if you have it on your phone. Now everybody can steal his local witnesses. – Right. At this point I'm going to turn it over to you to give a little more.

– Because a controller controls their own witnesses, you can add MFA to your witnesses so that even if your signing keys were compromised, they couldn't publish your witnesses unless they also compromised your MFA. That makes it doubly hard for them to publish anything.

And even if they've compromised your keys, that isn't enough by itself if you add the MFA to your witnesses. That's built into the witness structures and MFA. You don't do passkeys right now. You just do authenticator apps, right? You have an authenticator app that you'd have to also pass in addition to sending a signed key event to the witnesses. Right. – Any questions at this point? Anything else you want to see about that? Particularly, I'm going to give Evan some time to show you a little more, maybe talk a little more about Locksmith itself. – Data storage is just local files. – LMDB. – LMDB. – You need to use a StatefulSet. – Use a what? – StatefulSet like in Kubernetes. – Yes, would need to. For sure.

– The reason he's using Circus is so that you could run your witnesses and watchers bare metal, which means that you're not exposing yourself to all of the Docker Kubernetes security vulnerabilities. Most people don't know this, but NIST released a report that says — I'm paraphrasing anybody that cares about security does not use Kubernetes, because it's a single point of failure for all of your infrastructure. One compromise of one IT person's keys compromised 100% of infrastructure. And one of the recent supply chain attacks was an AI that was collecting Kubernetes credentials. And if that attack had gone on longer than I think they caught it after a few hours, if it had progressed, it would have captured all the Kubernetes credentials for everybody's cubes over the whole internet if it had had time. It would have brought down the whole internet hard. And that's because Kubernetes has a single point of failure. A single credential is enough to take down everything. And that's why NIST says if you care about security, you don't use Kubernetes. I know everybody hates that if you're in DevOps, but we at least did it in a way that it will run on bare metal. You can't say, oh, well, you guys didn't give me a way to be secure. If you want to run in Kubernetes, by all means, at your own risk, but you can use this bare metal if you want to. – I was tired of running on bare metal. – I know bare metal hurts.

– It's true. All right. We'll give the mic over to Evan for a minute to show a little more of Locksmith. Maybe talk a little more about that specifically. – Locksmith as a whole, you can go back and you have these isolated vault environments where you can hold different sets of keys. You can have two-factor authentication on your vaults. If you download a password, there's no password. You can get remote identifiers. This would be how you connect to other wallets, for example.

If you were able to do some sort of out-of-band introduction with identifiers from other wallets, you could import them here. You would see them here. That's why when you see like these identifiers, witnesses and watchers, they'll show up as remote identifiers because they have identifiers associated with them.

And then you can add like identifiers on your account. The way we have created this plugin in Locksmith and our overarching idea is that when other people want to contribute their own infrastructure they're able to just add a plugin to this menu and add their own compatible infrastructure. And that way you don't have to rely on a single service provider within your wallet to get witnesses and watchers.

The method for like creating a plugin is really simple. You can kind of like if you want to you could just like copy the folder of code in the repository, register a new plugin here and then change your UI to your heart's content. – I'd really love that if somebody in here who's building KERI infrastructure writes a plugin for this so that we can publish that and then when we do a demo they'll have their logo up there and not just the KERI Foundation. Just saying.

– On that note, so is the plugin just essentially discoverability for let's say I had like a testnet infrastructure and I want to connect my Locksmith to my testnet. Would I then write a plugin so the Locksmith knows where to look or like what exactly is threaded through the plugin? It would be where

to look for the infrastructure. Like for example, the KERI Foundation plugin is looking at kf-boot, which is kind of our infrastructure orchestration. If you wanted to do your own version of that, you could just have it talk to your… – And there's config in the plugin itself that lets you specify those addresses or hosts or whatever to designate where it's going to look for that stuff. – Isn't the config just a JSON file? – Not in this one. It's code for this plugin code. It's just part of the code.

– We also are working through credential exchanges. You're able to load in schemas and then apply for credentials and issue credentials. And we're working towards getting Sam's full IPEX exchanges, which are issuance and presentation exchanges between two wallets working. – These are version two. – this is version two of KERI and ACDCs. – When you say credential, that's ACDC. – Correct. The IPEX exchanges will allow us to create those beautiful graphs of delegated authority that Sam was showing us earlier.

– At the SEDI summit demo, the graduated disclosure, which is what is supported in IPEX v2. The current set of infrastructure that the people are using ACDC for credentials like vLEIs aren't doing graduated disclosure. That's in the spec, but it wasn't fully implemented. We're fully implementing that part of the spec. We don't have it in the wallet yet, but that's our target is two weeks from now to have it running in the wallet. We have it very close.

There's one remaining, I think only one remaining edge operator that has to be implemented, which we have an actual PR for which hasn't been merged yet, but it's very close. We are well on our way and should be ready to go for the SEDI conference to have this whole IPEX exchange in place. That's coming along quick. – Let's say I'm a city and I want to set up a witness for the residents of my city. Would I set up one, two, three, four of those maybe. Our recommendation it's either one, which is not as secure, but it's either one or it's four. You can do a three of four. – But ideally again, if I'm just checking my knowledge here.

Let's say my city sets up one witness. And then let's say my bank sets up one witness and then let's say my university sets up one witness. Now I make an identity and then I say, okay, I want my city and my bank and my university to then witness my logs. And that's a secure way. Am I saying the right stuff here? – Each of those would be, currently the way it's built — each of those would be a separate plugin in the wallet and you could do a configuration of one per each of those. – My bank and my university and my… If they provided plugins to the Locksmith wallet, I believe it's right. You could configure the identity to use four witnesses from four different plugins.

– Okay. Different organizations. Follow-up question to that. Again, I'm a city. I set up one witness in the city and that one witness can support all of my residents. I can have… there'll be multiple AIDs and all those AIDs can use the one witness. I don't have to instantiate another witness per AID. – No, – I wouldn't recommend you do it that way. I would recommend that you have one witness service, and then that witness service is a host that's running witnesses, but each witness for each person could have a different identifier. But if you did one witness, it would work. – You can do one witness with multiple AIDs, but that's not recommended? Is that what you said? – Well… – Again, I'll have the other ones too. But if my university does that and my if all of those locations do that and I'm using all of those, then it's okay for just the city to do that. – It'll work. It's okay. It's not ideal. But you got to understand that I live in this ideal world. Right. – And you got to also make sure we make the distinction between the witness service, like witopnet is a witness service and each one that… – each AID gets its own witness within the witness service. – Within that service. – That's why I'm saying is that your bank would set up a witness service, but it wouldn't use the same witness AID for everybody's wallet. It would just spin up a separate witness. That separate witness is really just an AID that says I'm going to sign with

this AID. Everybody's witness is signing with a different AID. You're not reusing AIDs to sign receipts. But you have one witness service that is basically running an endpoint where people send events to it and it has one database and it's storing the stuff in the database. – […] had a question. – Have you made any load tests on this service? – We haven't gotten to that point of really hitting it with a lot of load. No. But I don't know.

Healthcare might have done that. We haven't done that as a foundation. We just haven't had the resources to really throw that much at it just yet. But it's a fairly lightweight service though. It's hard to imagine that it would… – It's in my to-do list. I would share results whenever I get to it. – For sure.

Actually it was more of a comment than a question. I was going to bring up the issue or the concern of consideration of scale because GLEIF set up five witnesses for the infrastructure for the vLEI and each QVI has to set up a total of five as I said before with two that they could share from existing witnesses that have already been spun up. Net each QVI adds three more witnesses to the overall witness service for the vLEI ecosystem. And that was in order to be able… one of the considerations with scale. – We haven't put it through major scale testing, but then again, to be clear that you're talking about witness services, right? Not witnesses themselves. – Witnesses themselves.

Witness service of every two of them. My own advice. Each ecosystem governance framework requires that each QVI set up five witnesses. And in total, it's one witness service with five witnesses or more of that witness service. – It's not a big code change to change the witness so that what you're now calling a witness could become its own witness service. It's the same endpoints and stuff like that. You're just allowing it to create AIDs on the for each wallet that wants to use it as a witness service. Right? It creates the AID for the wallet, not for the whole witness service. – Now, would that be a consideration right now? Mostly everything is running on a custodial wallet from the QVIs that offer that. Would they have to do that? – No, for a custodial wallet, they may want to do it a different way. Right? The thing is, it's really… we're not talking KERI protocol anymore. We're talking infrastructure deployment. KERI just says there is a witness with an identifier that you designate as the witness for your KEL and you need a witness service and a witness service could be one service with one AID that witnesses multiple KELs or it could be a witness service that allocates a different AID for each KEL it witnesses. Those are two different arrangements and there's some niggly differences between how they perform. But under the hood, the infrastructure is the same. You have a database that stores KELs and you receipt KELs when you get them and the service says, oh, I witnessed this KEL, right? And I sign it with what AID? If I have one, I sign every KEL I witness with the same AID. If I have one AID per KEL, then if I'm witnessing an event from this KEL, I sign it with a different AID than if I witness an event from that KEL. That's the only difference. It's just whether you multiply the AIDs or you have one AID. And multiplying the AIDs means that you have more flexibility in terms of adding and subtracting services and stuff like that because you're not locked to one AID for everybody. Because if that AID gets compromised, it's compromised for every KEL that it witnesses. That's why it'd be better to have an AID per KEL than one AID shared across multiple KELs. – On that note, is there anything with regard to provisioning or a provisioner in kf-boot or right now is kf-boot more just an interface for something like Locksmith to use that plugin to discover already provisioned infrastructure? – No, we do provisioning. Like the way that the code that we got was set up is that like the witness-hk to create witnesses you had to hit like its local endpoints. You have to put… we had to put the boot service on the same machine as the witness service so that we could securely create witnesses and watchers and coordinate all that. It does provision. – Oh, so

help me understand that. The boot service can talk to the witopnet and the watopnet to provision witnesses and watchers on demand. – Correct. Do you have… you can pull up like the witness and watcher repository. The readme is good.

Here's the HTTP API. To provision witnesses for these controller AIDs, you're hitting these boot server ports. And then to actually talk to your witnesses, there's a different port for that. – You said securely provision these things. What's the relationship from a security perspective between your Locksmith wallet and the boot server? Is that determined by something? Is that part of the plugin configuration? And is it just… okay? – You probably want to use like […] to talk to your things, but I don't know if we have all that set up. – Okay. The Locksmith talks to the boot server.

The boot server then is installed in the same infrastructure where it's allowed to provision through these boot ports. Okay, I gotcha. – That's the config file inside the Locksmith code and that's those are the two addresses that I gave it to point to my local Docker. Earlier I had those pointed to our DigitalOcean environment just to make sure I could connect to it and I could connect okay to one of the hosts but a couple others were having trouble which is why I did the Docker thing instead for this particular demo. But that's where you change that.

– Final question, is the same approach being taken towards essentially every infrastructure repo inside of the KERI Foundation is that there's like a boot port where you can boot one up and then it's just there to use for Locksmith? – Wait, say that again. Sorry. – Like it's the same provisioner approach being taken to the mailbox and the other infrastructure. – Those will be a little bit different, but we're still kind of in development on those, so I'm not going to speak to those too much yet. I don't want to give you an answer and then end up doing something else. We'll try to keep it as consistent as we can, but we're not sure yet exactly how that's going to land. – I have another conceptual question here. I asked about witnesses before now on the watcher side.

Now let's say like I'm a car salesman. Like I have a lot of cars and I want to you know get financing or somebody and so is that a situation where then I would need some watchers then to watch people, or is that not? – Well, watchers are watching witnesses. – Right, so then they would watch the witnesses of, like I said, those other ones like the university and the bank and all that stuff. – And if you want I mean, if you want the KERI ecosystem as it's designed, you're always going to want at least a watcher watching. – But my question though is like, because if I understand correctly, the witness is a controller concern primarily. It's like it's my infrastructure as a controller and then a watcher is more of a validator's concern. Like I want the watchers so that… Is that true? – Yes. That's one exception. Every controller should have a local watcher that they use to watch their own infrastructure. You want a watcher you trust to watch your witnesses. That's how you detect whether or not your witnesses have been compromised. And then you have watchers that watch other people's infrastructure. And that's what normally people think of as watchers, right? The outside watcher network is watching other people's infrastructure to detect duplicity, but you want a watcher to watch your own infrastructure that you trust. That's why the default configuration is you set up one watcher you control. You control one watcher and four witnesses. That's your infrastructure. And that one watcher is mainly to watch your stuff. It can watch anybody else's stuff as well. But you want to have something watching your infrastructure. And the reason is that the only way you can tell whether your witnesses have been compromised is if it goes through watcher logic. Watcher logic only will see an event if everything is perfect. If it's delegated, it has to be delegated. If it's witnessed, it has to be fully witnessed. Whereas a witness will witness

something, any event that comes from a controller. A witness doesn't wait till all the other witnesses witness it to witness it. It just witnesses it when it verifies that it's in the KEL. Right? You can't tell whether your witness pool has been compromised without a watcher that watches your witness pool.

Now you could trust somebody else's watcher to watch your witness pool, but it makes sense to have your own watcher you control. If you're spinning up witnesses, you might as well spin up a watcher that controls your stuff. Or you could have a watcher run locally in your wallet, but then it's not highly available so that's a trade-off as well. Does that make sense? – Yes. And then to round out the watcher thing, like, what would be a reason why I might want to watch other people's witnesses? – To detect duplicity, whether they've been compromised. – Like in a business sense, right? Like if I'm dealing with somebody and I want to make sure that this person I'm dealing with is reputable. – because you can't tell the difference between somebody who's been compromised and somebody who's malicious. Both of them look like duplicity, which is a good thing because if you see duplicity don't trust. Because it doesn't matter whether they've been compromised or they're malicious not to trust it. – If I have customers and they and I OOBI them, then I know who their witnesses are. – That's right. – And then I could choose to watch those witnesses of my customers or whatever. – Okay. Thank you.

Sam has published a draft of some of the SEDI schemas in a repository called SEDI in the KERI Foundation GitHub. I grabbed them and built them in here so that you don't have to think about it. You can grab one of these schemas, load it in.

And then… I might have needed to grab the previous witness code. I'll show you. – And those schemas are again ACDCs? – Schema for ACDCs. They're JSON Schema.

I'm going to do an OOBI here. I just exported this local identifier. Let's see if I can import it into this one. – And why are you switching vaults? – Just to have two vaults talk to each other to prove that you know… because the two vaults are isolated key environments. It'd be the same as if I just had two wallets running.

[…] I think I need to also take this one, put it in the other vault. – Is this going through a mailbox or is there no mailbox involved here yet? – That's a good question. – Did you connect it up yet? We have the mailbox service, but we haven't quite finished with the implementation on that. – It might not work because we might not have the mailbox service running.

I can apply for a credential from… The contact isn't showing up. You'd be able to apply for a credential from one of your remote identifiers, and from there you'd be able to do an exchange. It might just be we need the mailbox set up to send messages between and […] – You've done it locally, right?

We have notifications here. You would send like an application message to an identifier. It would show up in your notifications and then you could see each of the steps in your exchange where you're disclosing which fields for a presentation that you would want or to create a credential and then send that credential to somebody who would be applying for that credential. And then you'd be just sending the notifications to the mailbox and they would show up here.

– The problem is: do they need to… these two vaults, I guess with AIDs in there, they need to OOBI each other or something? Is that… – We need to get their mailboxes configured to each other to send messages between each other. – The issuance exchange would go over the mailbox. It could use a different transport but the way Locksmith is set up is to use the mailbox. Just the contributed version of Locksmith didn't include the mailbox. We had to write our own mailbox and stuff like that.

– For example, for a presentation where you're going to the liquor store and you want to buy like a beer, you would show your credential and then you could choose to disclose that you were, for example, over 21, but then you wouldn't have to disclose some of the other information about yourself within that credential. Then, we would have a field where you would be able to choose which information you would like to disclose and send to a verifier such as a liquor store as an individual. – My question is on the original issuance of the credentials. You said that was also going to use a communication with an application through the mailbox. Did I misunderstand or understand that? For original issuance of a credential, there was going to be an application function that was then going to send a request to have a credential issued and then send that credential to the person who was supposed to get it. – Correct. – I understood that. In that, I think that what Joe was asking was, so for the vLEI, we have an identity authentication layer that does a challenge response process before actually giving the credential to the person who's supposed to get it. Is that part of this in the background somewhere? – Yes. I'm also working on like a identity proofing challenge response. That's just exactly what you're describing. Yes. – For SEDI, the state has defined their own challenge response mechanism that's slightly different from the GLEIF ecosystem challenge response. To support both of those, the wallet would have to support both types of challenge response. – Okay. It might be worth taking a look in order to be able to see what they've done and compare.

– The main difference is that instead of just signing both sides of the challenge response, which is what the GLEIF one does, which is simpler, because the state of Utah is planning to use external proofing agents for the identity assurance instead of a GLEIF authorized representative or a QVI, a qualification like in case of DirectTrust, they might just hire somebody that they're requiring that the response be anchored in the KEL of the user as opposed to just signed by the user. – It's more secure.

– It's more secure. – Because we're having issues with people coming to us and telling us that it's too much, the way that we're doing it. And this is on top of what we're doing. – The state is now going to have millions of people who are going to be really lax with their key management. Basically by anchoring, having the wallets automatically anchor, then we take that away from them. They don't get to be so lax. But it happens under the hood from their perspective. – Okay. Thanks.

What's that automatic anchoring you were just referring to? – Identity proofing. – Identity authentication. – Yeah. In an identity and an IAL, NIST IAL level three. It's not two plus, it's level three for the state of Utah. – We do the identity proofing first and then as a second step the identity authentication. – But then an IAL level three, the last step in identity proofing is you have to bind an account or an identifier to the proofing session.

It has to be done in session. In session, the proofing agent sends a challenge over a non-internet channel. Either an NFC or Bluetooth, something, well maybe not even Bluetooth, but optically, right? Like a QR code. It can't be Bluetooth or anything that somebody can surveil. And then you anchor the challenge in your KEL, which means you're signing it to anchor it. But now it becomes detectable if somebody showed up and impersonated you. Let's say somebody who looks like you stole your identity documents and showed up at the proofing agent. And they said, okay, you're just a person. You don't have all of the security that you have at GLEIF because you're dealing with people that represent entities that have layers of management for them to show up. You just have a homeless person off the street, for example, show up and say, Hey, I want an ID, right? And they go, well, here's my documents, right? And they may have a grainy picture that doesn't look really good and they go, well, it's the best you got.

This way, if somebody stole their signing keys, if they were compromised, then at least they could detect that they were compromised. Between the proofing session when they actually go back to the state and apply for their SEDI, there's an opportunity to verify those anchors. It adds some extra security that might be useful.

– I have one more question. The kf-boot thing. I noticed you had to add a schema here. I assume that means that when you booted stuff up, there were no… Is there a way to include schemas? Because for example, like again, if I was a university and let's say I made my plugin, I would probably have my diplomas or the things that I offer and everything like that. When I have my plugin and my boot, not only am I going to be like, okay, here you can set up witnesses and watchers for your university, but these are all the schemas and things that… – Schema discovery is what you're talking about. – Yeah. – The schema hasn't to do with kf-boot. It has nothing to do with your infrastructure. The schemas are only for the actual credentials. – but a credential, an ACDC credential. And if I'm a university, wouldn't that be the diploma? Like I'm… – Yes, but it doesn't have anything to do with kf-boot. You just said something about credentials for the schema for kf-boot. That's what confused me. – I guess I was bundling it in there because it's like if I'm a university and I'm trying to bootstrap my university infrastructure, I don't know. I would suspect that might be a good place to just put all the schemas in there. – We have a plan for schema discovery within the wallet. Doesn't really have anything to do with the booting and onboarding of the infrastructure. Just a separate mechanism.

You could publish, for example, or the university could publish their list of approved schemas and then the wallet could go find that. – They could publish all of their credential schemas as.well-known for example. And then anybody that wanted to talk to the university would go say, oh, here's the schema that the university uses for their ACDCs. And they can publish them in multiple places. And because each schema has a hash as its identifier, anybody changes the schema, it's a different hash. You can detect that it's been tampered with.

You know it's not the right one. You just have to get the authoritative publication of the hash someplace and say, this is the one the university uses. The university would have a vested interest in making sure that the hashes and schema they use are published in well-known locations that people can verify came from the university and then any wallet can be preconfigured with those if they want to or they can discover them on the fly. But there's an advantage for people in the university ecosystem to just pre-configure their wallets with those schema. – Kf-boot wouldn't be the right place for it anyway because it's even though they're kind of related in a way in the sense it's KERI ecosystem, kf-boot itself doesn't really have any understanding of schemas or credentials or any of that stuff. It's really not the right place for that kind of stuff to be.

Separate consideration. – Thank you. – Cool. Anything else? Any other questions? We pretty much run through the time. All right. Thanks everyone.

TALK IV

2026-11-02-ryan-hansen-samuel-m-smith

KERI Mini Conference November 2026 · talks · 55:50

SUMMARY

AI generated summary.

Ryan Hansen and Samuel M. Smith define open infrastructure: shared data without shared governance. A verifier who can check a key event log need not trust the sender or a consensus set. The cold-start problem means witnesses and watchers must scale first — a public identity utility is one bootstrap. Money sits in assurance on top of cheap rails, not in selling the rails themselves.

probably tired of hearing me talk. I'll try to talk lower because I understand a lower voice is a little more pleasing. I use my Samoan voice. Anyway, is that more pleasing? A lot of vowels, right? Rhythmic flow. Polynesian languages are kind of nice. Anyway, I have some slides, but I actually wanted this to be a little bit more of a discussion because this is the problem that we're trying to solve as a KERI community. And unlike some of the IIW sessions, we get a lot of people that are anti-KERI. Everybody here is pretty much KERI friendly, right? What is open infrastructure? Have you guys heard that phrase before? Open infrastructure? Okay. There may be more than one definition.

This is how I define it. It's based on open standards, open source. But what makes it infrastructure is the infrastructure is portable. It allows portability by the users. If you have vendor lock-in, it's not open infrastructure. There are lots of infrastructure systems that are based on open standards. They're based on open source, but they allow vendor lock-in.

They're not open infrastructure. For example, if you have a mail server. Is SMTP mail. Is that open standard? Are there open source implementations of it? Can you easily port your mail from say one mail server to another? – Theoretically, right? – Right? Does your mail client let you do that? Not usually..

You have to do a lot. It's not easily. Right. Okay. It's not. Despite those two, the portability thing is a problem. Now, why isn't stuff built portable? Because the incentive structure is wrong. To be open infrastructure, it should be designed so that the incentive structure fosters “coopertition” and portability. What do I mean by “coopertition”? Have you guys heard that phrase before? It's people are able to make money in a competitive environment where they compete on their implementation quality, but there's an incentive for them to cooperate so that everybody benefits if there's cooperation. In other words, the network effect is that if we cooperate, we make more money than if we don't cooperate. It incentivizes cooperation. For example, in KERI, there's an incentive for people running watcher networks to cooperate. Can somebody tell me what that incentive is? Why would one watcher share information with another watcher? – Detect duplicity. – Yes, well, they all detect duplicity.

But what incentive do I have to get stuff from another watcher and what incentive does the other watcher have to share that with me? – Ground source and reputation. – Yes. Right. A watcher is most effective if it has maximized scope. If I can see the whole internet, then nobody can eclipse me. Right? How do I see the whole internet? If I don't have the resources to physically see everybody's witnesses, but somebody has resources to see witnesses that they have an incentive to see, and if I have an incentive to see witnesses that they don't have the same incentive to see, to spend the resources, and I say, well, I'll give you the witnesses I watch if you give me the witnesses you watch.

Is there an opportunity to cooperate? Absolutely. If I'm building witnesses for the GLEIF ecosystem, I have an incentive to watch the GLEIF ecosystem. I don't have an incentive to watch the Utah State ecosystem. But if the Utah State ecosystem is watching the state ecosystem, the delta, the marginal cost to share with the GLEIF ecosystem becomes small for both. If marginal costs are lowered and sharing is just a marginal cost, then there's a large incentive to share because now I've doubled the scope for a marginal cost. Whereas if I had to watch both ecosystems, I'd have to spend much more if I had to watch it myself. Right?

Anytime you do an incentive structure, if the marginal cost is lower for cooperation than not to cooperate, then you incentivize cooperation. You build things so that marginal savings is on the side of cooperation. Watcher networks are inherently mutually incentivized for cooperation because the their value comes from scope and scope if I have to watch it myself is expensive. But if somebody else is watching it and they share it with me, all they're sending me is a bunch of proofs. They're just sending me a bunch of proofs of duplicity. Because they don't have to share me everything they watch. They just have to share with me duplicity that they found because that's all I care about. I can say, hey, do you have any proofs of duplicity? And they go, I got five of them. That's all I have to share with me. But to find those five proofs, they had to watch a million witnesses. Right?

And they can say the same thing. How many witnesses are you watching? I'm watching a million. Are you watching the same million? No. How many proofs of duplicity do you have? 10. Here you go. That's a marginal advantage that now means that everybody in their network, every watcher in their network is effectively providing coverage over 2 million witnesses. And everybody in my network is getting coverage of 2 million networks, but we only have to watch 1 million. We don't have to incur the cost of watching 1 million witnesses, but we get a thing. And guess what? That scales because the amount of duplicity is a small fraction of the total amount of traffic. If every ecosystem shares with every other ecosystem, then you get global coverage and the marginal cost is the cost of duplicity, not the cost of watching. Does that make sense? Okay, that's an intentional design, right? By design, the watcher network is designed because people would say, well, in the blockchain world, well, you have to have incentive for infrastructure. You have to have tokens, right? You have to be able to mint tokens because otherwise, why would you pay for it? In this infrastructure, if you have incentive to watch your local ecosystem, the duplicity detection lowers the marginal cost that it's almost free to share what you watch when all you have to share is duplicity with other ecosystems. Then it flips the incentive structure. You don't have to build expensive token-based incentive structures to incentivize people to do it. The problem is that nobody makes money in infrastructure that's open and portable until you reach scale. Nobody makes money in it, right? You have what's called a bootstrap or cold start problem.

We've kind of talked about this before, but everything in KERI is designed so that there is open decentralized, not shared infrastructure. You have shared data, but no shared governance. Okay, that's a critical property. What does that mean to have shared data and not shared governance? If I send data to you that you can independently verify without having to trust me, then you don't care. You can verify it. You don't care where it came from. All data is created equal as far as you're concerned, right? Shared governance means that I can't trust it unless I trust the governance. In a blockchain, everybody has to trust the governance.

Usually the governance is a distributed consensus algorithm and the token structure that incentivizes people to run nodes in the distributed consensus algorithm. The tokenomics of blockchain are critical to that shared governance working. KERI says, I can't afford that. That's too expensive. Shared governance is extremely expensive. What I need to do is make it so that it's truly totally zero trust. I

don't need shared governance because everybody can independently verify it and I need to make that independent verification cheap enough that they afford to do it, but it's not zero. That's the problem. It's still not zero. We've reduced the cost floor as low as possible.

The cost floor is as low as we can make it because we got rid of shared governance. We made everything independently verified, but it's not zero. We still have the problem. We have a cold start bootstrap problem. How do we get enough infrastructure to enable scale and the people providing the infrastructure to be able to afford to provide it until it reaches scale so that they can make money because they won't make money until it's at scale. If I'm running a witness service, I'm not going to make money until it's at scale.

Because I'm not going to charge people $100 a month for a witness, right? Nobody could afford that. They want to pay a dollar a month or maybe 10 cents a month, right? How do I afford it? That is the hard problem. What we're trying to do in Utah in SEDI is make the case that if the state of Utah primes the pump, everybody else that wants to be in that ecosystem is incentivized. A Public Identity Utility.

That's the term that we came up with. What is a utility in a regulated environment? What's a regulated utility? – Electricity. – Electricity. Okay. Power. Garbage. – Telecom. Right? Garbage. Sewers. Right? And why do we have utilities? Now, if you're from the state of Texas, you probably are going, we know why you have utilities. Because the state of Texas has a mishmash of a power grid because they didn't create regulated power companies. But if you're in most states, there's only one power company. The power company is a regulated utility.

It's a public-private partnership. They get to make money, but they can't make money without getting approval for any rate increases, any changes in infrastructure. They're guaranteed to make a fractional profit, which if you're big enough, a teeny bit compounded over years amounts to a lot of value, but they get a monopoly, right? They get a monopoly, so they don't have to compete with anybody, but in return, you get to regulate them.

They're not guaranteed the most efficient, but you do guarantee that you don't have non- portability because everybody's power is portable, right? It doesn't matter which house you go to, power just works, right? A public identity utility says you're going to have an identity system that is portable and interoperable everywhere, not because you've forced people to interoperate, but because you basically said, we're going to have one. We're going to have a utility. It's going to be an identity system and the state is going to regulate it and they're going to standardize it and they're going to ensure that there's enough money to build out the infrastructure.

Okay. That bootstraps it. What are the bootstraps? One is IAL 3. That's Identity Assurance Level 3. Okay. Is the state of Utah or all states in the identity assurance business already? They already are. They already do it. They don't do it to IAL level 3 because they don't have to. But if I'm going to have a persistent identity that lasts a lifetime, I want to make sure that when I start it, I start it with a really high level of identity assurance because guess what? With KERI, I can amortize that one identity assurance session across the lifetime of that one citizen. No other identity system allows you to do that. Every other identity system, every time you rotate keys, you either re-incur this cost or you're insecure. And guess what they've chosen to do? Not re-incur this cost. Which means the first time you rotate your keys, you've now become insecure. That's that describes the certificate authority system. CAs are supposed to repeat that level of identity assurance every time they reissue a X.509 certificate. That's what the spec says. Do they do that? How many people have issued new certificates on CAs? How

many of you actually went through any kind of identity assurance for any reissuance of a certificate? That's not what the spec says. Certificate authorities are supposed to redo, re-verify everything about every piece of information in that certificate when they reissue it. What's that called? “Let's encrypt”?

The problem is that if you can't amortize this cost, you're incentivizing insecurity. KERI allows you to amortize that cost and the state in SEDI is going to pay that cost. Okay, but that isn't the infrastructure. That's just bootstrapping the identities that need the infrastructure. I'm not sure how they're going to pay for the infrastructure. There's been discussions about them paying for infrastructure for disadvantaged populations where the state will actually subsidize infrastructure providers to provide witnesses and watchers by issuing coupons to disadvantaged population to be able to get accounts on those services. That's one way to bootstrap it. Okay. Another way to bootstrap it would be for providers of witnesses and watcher services to have a market opportunity that allows them to make money on something else that is enabled by them giving away witness and watcher services. If I'm a bank and the state and the federal government (this is a big if, but I think they just passed a law allowing it) allows one bank's KYC to be used by another bank… What did I just do? I lowered the cost for every bank because every time a customer changes banks right now that bank has to incur the full cost of the KYC every time. Portability is expensive. And the banks have to do it. And the thing is they can't really use it to stop the customers from moving because they have to do it one way or the other and the customer can just say, I want to move to a new bank. And the bank says, okay, we have to move your money, but we're only going to move it to another bank. And that other bank won't ask us to move it because we can trust that the other bank is following the regulations or there wouldn't be a bank. There's already enough trust that you can move your money between banks, but every bank every time you move your account is having to incur the cost of KYC. If you have an IAL3 and a regulation that allows one IAL3 to move between banks, then all you have to do as a customer is issue an authorization to the new bank to use your existing identity in the IAL3 that the state paid for to do that. Now the banking industry has an incentive to issue to create free witnesses and watchers because they get KYC savings.

Because your portable KERI SEDI-endorsed identity now becomes a savings to them. That's an example of how, if you do this right, if you think it through, you can start to build incentives, right? If you have to have witnesses and watchers talk to the state and the state says, well, we don't want to pay the full freight for witnesses and watchers because we have lots of citizens who make enough money and so we want to impose on them a hidden tax, right? States do this all the time. You're going to have to pay for witnesses and watchers, right? But we'll pay for disadvantaged populations. But everybody in the state also has a bank account and the banks say, well, wait a minute. If we just spin up witnesses and watchers for our customers, then we can get free KYC because the state paid for our KYC and we didn't have to pay for it. And if you look at the cost to the bank of the KYC, you are in the banking industry, how much does it cost to onboard one person for KYC? What does it cost the bank? – I don't know what the exact figure is, but just multiply it for many times within the bank because you have to KYC yourself whether you're a person or an organization for every service that you apply to. – Every service you apply to. Let's say, could it be $100 a year? Okay, well, $100 a year pays for some really awesome web scale infrastructure. I mean, how many witnesses can we run at $100 a year? – It's a year. I couldn't even guess, but it's, you can run a lot. – Right. We have four droplets that cost us about 100 bucks a month for four droplets on DigitalOcean.

Each droplet has, I think, is a four or eight cores. I can't remember. – Four cores each. – Four cores. A ton load of space, quite a bit of stuff. They could easily run 10,000 witnesses and 10,000 watchers and not burp. And that's a hundred bucks a month, right? If a bank wants to run a witness and watcher

service for a hundred percent of their customers, it's $5,000 a year max. They're spending a hundred times that on KYC. If they do the numbers, they go, you know what? We should just spin up free witnesses and watchers to incentivize people to KYC us using their SEDI identity instead of KYC in the old-fashioned way where we have to carry the freight, right? Every industry that does identity has a similar cost trade-off where this identity assurance is the savings that the state provides by giving them portable identity assurance. Right now, identity assurance is not portable. Without KERI, it's not portable. Everybody has to redo it. But with KERI, it becomes portable. That they should be able to justify the cost of spinning up free witnesses and watchers just to so that they can gain that savings for their customers. That's the idea of this. That's the motivating factor for the state funding. And for this to work, a bank has to be able to say, well, I've got an IT person at the bank. Show me how I would spin up free witnesses and watcher services, because I don't want to pay somebody to do it. I just want to download this open source, open standard software and just run it on my own servers, right? That's what the KERI Foundation's attempting to do, is build enough open source code to set up so that people like a bank can go and do it. What other industries can you think of that benefit from onboarding?

– Insurance. – Insurance. Absolutely. One of the most fraud-filled industries in the world is right now? Insurance claims. And why is that? Because I can deep fake an insurance claim, especially in the healthcare industry. It's easy. I know what's on the form because the forms are public. I know your address. I know the address of all of your providers that submit insurance claims. And you don't have a good way of KYC-ing the submitters of the insurance claims.

Every day you'll read about, we lost a million dollars because this fraudulent pretended to be a clinic someplace and submitted all these insurance claims for all of this healthcare work and they had all the right numbers the claim numbers and stuff like that because they stole them from me the insurance payer because my IT was so bad that they could steal all of that information so they could submit fraudulent insurance claims. Insurance is a good one. Another one? Anybody?

– All kinds of messaging. – but money. Messaging tends to be free. We need something where somebody is making money and they're spending a lot of money to combat. Because the only thing the state is providing is this. We have to leverage this free thing the state's doing to incentivize people to say, well, to get this, I'm willing to spin up free witnesses and watchers. – All of employment. – All of employment. Yes. Every HR department of every company has to KYC their employees to onboard them. And they have to verify their transcripts. They have to verify their employment history. They have to verify their claims history for prior… what is it? – Convictions. – prior convictions.

Background checks. Yes, they have to do background checks. They don't want to hire somebody who's a criminal, right? Okay. – Government service. – I was going to say government service. – There you go. – Like EBT, SNAP, web service. – Yes.

Since the state provides those, the state has an incentive to use its own IAL3. We're talking about people that aren't the state because the state's going to do that anyway, right? Who else? – A lot of times now what I'm seeing is that they are sharing that information. Like in the state of California. Or in my city anyway. In my city, if you've got PG&E care, they'll share that with EBT or EBT will share that with such. It's not necessarily just government. It's because once they've gotten it from one, now you can get all these other services more easily. – And that's one of the reasons why healthcare is one of the adoption vectors for KERI because it's probably the worst for that. It's the single biggest sector in the

economy and it has the worst fraud of any of them because people have to have healthcare, right? And they have to deliver it. They're required to deliver it. And they have to deliver it even if they can't really verify the person because they can't turn somebody away.

Somebody comes in and says, I'm dying. We can't verify all of your stuff. They're still going to have to deliver the care, which means they either get paid or they don't. And sometimes they just the payers have to deal with the fact that they don't have really good provenance. – All consumer services that have to verify something about somebody who buys their services? – Yes. – Like buying cigarettes, buying alcohol, buying… – every regulated denied service, right? Alcohol, pornography, and in the state of Utah, social media for 13 plus or under 13 is denied and 13 plus is allowed, right? The point is that if you think about it, and you think about the core thing, is you're getting this thing for free that's never been free before. And I wish Phil was here because he wrote a white paper where he said, (I didn't prompt him to write it) the thing that's always killed digital identity is that the identity assurance was too expensive. Nobody wanted to pay for the identity assurance. As soon as somebody like the state pays for identity assurance, for individuals, then it unlocks the thing that stops identity from digital identity from being used in general, which is now you have the most expensive thing that's a gating function paid for. Now with KERI, since you can amortize that gating function across the lifetime of an identifier, which means if it costs you a hundred bucks to do KYC, that hundred bucks gets amortized across a hundred years. Whereas every other identity system can't amortize it. It's even worse for them. Right? If I'm doing DNS CA, every time I churn my certificate, I have to re-incur that KYC. But KERI not because perpetual identity.

Here's the equation. Perpetual identity equals IAL3 forever. Okay? You do it once, the cost is amortized across the full lifetime of the perpetual identity. It's the minimum cost to bootstrap. It's not zero, but if this is valuable enough, then people should bring free witnesses and watchers to enable them, their customers that they're, that this is a cost center for, to benefit, for them to benefit from that by enabling their customers to use a perpetual identity system. Yes. – I think there's a lot of synergy too. Like even industries that don't necessarily rely on identity assurance right now, if it's free, they could, and it would just augment in a very synergistic way, I know in the insurance industry, the insurance companies have approved people who fix things for whatever was damaged. That could be used by adjusters or even the auto mechanic shops themselves. Or the shops that do home repair and things like that. They could even use that. – Yeah. There's a bigger ecosystem. The point I'm trying to make is that if you're thinking of starting a company, with KERI, you're not going to make money spinning up witnesses and watchers. I'm sorry. — Only the people that can operate at huge scale are ever going to make money doing that. But if you can benefit from IAL3 plus forever by giving away witnesses and watchers, you have a business model because now the infrastructure that stops KERI from working becomes free. KERI infrastructure becomes free because the state of Utah is paying for this. That's the opportunity. – What concrete ways has the state of Utah indicated they might be willing to pay for it? Is it just the… I know you mentioned… – No, no.

They've said they are. They have to. – You can't reach citizen. – Yeah. You can't get a driver's license without… If you want a SEDI, you're going to get an IAL level three identity assurance process. – Wow. Did they say they're going to like integrate it into the DMV or something? – Yeah. – Wow. Or they might hire id.me to do it for them. Or anybody that shows up and says we're good enough at doing identity assurance and we want to do it for the state and the state will issue them a contract and give them a delegated identifier to be an agent for the state, an issuing agent.

The point is that that's the place where you can make money in the KERI ACDC ecosystem, not infrastructure. You want to give it away. Essentially, you want to give away KERI infrastructure and then make money on the fact that you get IAL3 forever and the state paid for it. Yes. – I think what's making the state want to do this based on what you've said today, correct me if I'm wrong, is because of legislation that they put in place that had no mechanism for doing the social media stuff for the 13 and above. Now they backed themselves into a corner for that. – Yes, they backed themselves into a corner. – I'm going to look at the magic ball here.

The consortium that's coming up of multiple states that are going to support this. – They have similar problems. – Do you think there's a possibility that there would be kind of a cross-state line legislation? – Yes. They want to have reciprocity between states. That if one state does IAL3, that you can move to a different state and not have to repeat it. Does that make sense? It's not just forever in your state, it's forever in every state. – What is the identity proofing instruments that you have to produce? – IAL3. Says you have to use so many first class documents.

That means passport. And then you get so many second class documents and then you have to have a biometric and then you have to have a secure way within session to bind that proofing session to an identifier. – Okay. This is a future question. That's how we would have to back end into something like this.

But let's go to the future where the baby that was just born last week and they start to get their identity. At what point do they get an identity at a state level? Do they have to be a certain age or… – No, no. They could get it at birth. And the reason they can get it at birth is that in general states trust the chain of custody from the womb to the form that you fill out that says this is a person. – They trust the hospital, not you.

Well, I said the chain of custody. Which is you're in the hospital But you said the form you fill out. – To the form you fill out. – but you're filling out the form of stuff. – but the hospital is the one that certifies the form. – but mom doesn't fill out. – Okay. The hospital fills it out for the mom. – Actually, birth certificates are done differently in every state. That's what I'm getting to. – There will have to be a normalization so that states can trust birth certificates from other states.

That's one of the worst provenanced primary identity documents are birth certificates. And George McEwan, he likes to give the example of, he was in a state, and the way they did birth certificates was a paper entry in a log. That was a birth certificate from some little town where somebody signed it. – And even today, you look at, in California, you want your birth certificate altered. It's not even a court process or anything. You just ask for it.

– It won't happen overnight. Those sorts of things will take time to iron out, but yes, that's where it's going to end up. – The rumor is that there may be as many as 15 states in the consortium. – Yes. I've heard it's double digit. – If that is true, it's about a third of the country. That's too much to ignore by pretty much any industry in the identity space. If true, which we won't know until two weeks at the SEDI summit, it will be like a bomb went off in the identity industry.

It will be a really big deal. – I'll tell you the fear, is that they will say KERI and ACDC isn't mature enough. And that's a real fear. And what I'm trying to do is I'm trying to tell the KERI community, there's a reason to contribute to open infrastructure because open infrastructure unlocks all of the business opportunities that aren't going to be unlocked any other way because you don't have anybody else in any other vertical paying for the IAL3. But SEDI, you have that forcing function.

– I wanted to give some experience that we've been having with the vLEI and in seeing this situation here, the requirements that SEDI is putting forward, it's only going to work because there is regulation and legislation behind it and the states are behind it because the actual experience that we're having with the vLEI is that I'm being told time and time again that even though I say that these are long-lived credentials and we're trying to be able to put the work in at the beginning in order to be able to make them secure enough to be long-lived credentials. I always get thrown back that it's too complicated. It takes too much time. It's too costly. You can't take somebody who's a board chair or the CEO of a multi-million dollar company sitting down even for two minutes to be able to do this. And I'm there like, isn't the threat of compromise because they're important people incentive enough for them to want to be able to have this kind of credential and still the same arguments keep on getting thrown back because people are only familiar with the processes that not even not being carried out by CAs now.

They're comparing already flawed processes to something that's… – This flawed process is low friction. – Yeah. – Your flawed process is high friction. The problem is this is not a flawed process… that is, but they don't see the advantages. It's a challenge. It's a hard thing. User experience is a difficult thing to solve.

– CEOs are inappropriate people to be in charge of the company's basic infrastructure, right? They're not responsible people. You need someone like a CFO, a COO, someone who is a execution focused someone lowered down even. – We're talking about the CEO receiving one of these credentials and not taking the time in order to be able to get one. They're the receiver. They're not managing anything about the process.

– They shouldn't be involved at all. – They need to be involved in the process to receive their digital identity credential. – But why do they need a… – If they go to a board meeting and they want to vote as CEO and they want to prove their identity. – They sign regulatory reports. – they impose liability upon themselves every time they sign their 10-K or whatever it is for their annual report in the US. They don't understand the liability that they subjected themselves to when they… but it's a valid point. They don't think that they have to care about that.

Anyway, I don't have a… – Two things. One about the complexity of KERI. We've been hearing that criticism for years. And I think that now has to melt away because people use AI to do their coding and handle complex things. And my experience with the frontier models is they are big fans of KERI. When you ask hard problems about key management and if the model is not completely familiar and you give it material to make it familiar, it immediately becomes a fan and all of a sudden has answers for a lot of hard questions it didn't have. And if it's already familiar, it recommends it. Zach was here earlier today and the reason he's part of the community is because Claude recommended KERI to him.

And it's the reason why he even knows about KERI is because Claude has that. I think that complexity argument is going to start to go away. Secondly, the cold start problem that you referred to earlier. Yes. That's something that's part and parcel of any two-sided market, right? You have supply and demand and you try to get the supply going and the demand going at the same time and it's hard to juice both parts of the market. With Utah doing what they're doing and potentially 15 states, if it's could be 10 states, could be five states, might be 20, who knows what the real number is, but a good number of states joining them moving forward. What they're doing is they're priming the market with demand. And what we've been waiting for years and it's coming to IIW and talking about these different identity technologies, we've been playing on the supply side. We've been playing on, oh, there's this cool thing that we want everybody to want. We want everybody to have self-sovereign identity. We want them to have these technologies.

But we couldn't make them want it because we are all staring at the cold start problem of the two-sided market. There is a kind of a once in a lifetime opportunity where you have a bunch of states that are banding together potentially led by our home state of Utah, your and my home state of Utah, who are priming the pump and putting their thumb on the scales on the side of demand, which means money. That means revenue. That means business opportunity for those that can come in and provide this supply. It's going to happen. It's only a matter of which companies can come in and provide that supply and satisfy the demand, but the demand is absolutely coming. – I firmly believe that it's going to happen, but there is a risk that it doesn't. And if it doesn't happen, then it may never happen because continuous identity will be what the market provides.

And everybody will say, well, we have no choice. They will just say, there isn't another option. – I think it needs to be said why the state of Utah is moving forward because it was speculated a little bit earlier what the motivation is. And people need to understand what Utah's motivation is for doing this. It's not because they want everybody to have digital identity. It's because of privacy.

It's their chief privacy officer has gotten everybody motivated and frankly scared about the identity options that that exist. The alternatives are bad. And if they don't do something positive, then everyone's going to end up with something negative from a privacy and liberty standpoint. That's the argument that's winning. And that's why it's unanimous on both sides of the aisle, left and right.

– Because everybody cares about liberty. – Completely unanimous. – It's less about privacy, more about liberty. – it's more about liberty. They've gotten it that if you're against this, you're like, hate your mom and hate apple pie and hate baseball. And they've really gotten it politically toxic to oppose it from a liberty standpoint. When you say that it might not happen and that the continuous identity of AI, that surveillance identity, is what the market will offer, I can almost guarantee you, because you and I have been meeting with state of Utah for how many years now? – Three. Almost four years.

– Coming in March, it'll be the fourth anniversary or April of our fateful first meeting with them. And Utah, if they have to go it alone and it'd be one state. – They would do it. – Utah would go it alone and be the only ones with their own identity technology because they're going to do what's best for the citizens of Utah for privacy and liberty even if no one else follows them.

And others will. I think they'll get other states. But that is the principle. It's freedom, it's liberty, and it's privacy. That's what's motivating them. – I have a question about the consortium. This started with legislation and that's what's driving it, not the technology. – That's right. – And then the legislation had to find technology that matched what they wrote. – It's both ways, right? The original social bill was written without any technology.

Every bill since then has been written with the influence of people from the KERI community guiding it, right? They know the end point, but they can't jump there. They have to say, well, no, we have to lay the groundwork. We know that we want to get to there. We know the technology will allow us to get where we're going. When they did the social engineering one, they didn't know that. But the next bill, you That they wrote in 2024 and then in 2025 and then 2026, all of those were based on they now had a roadmap at that point. – Yeah. But this state has this legislation which has been developed over four years, right?

– Yep. – The consortium, is it more going to focus on getting similar legislation into other states? – they're using Utah as a model legislation. – So that they can go to their states and however their legislative process works. Cause so that to me it's several more years that these states will actually have the legislation. – they will roll out slowly over but they'll go faster than Utah. – Because they have a blueprint to follow. – They have a blueprint. They can jump to the head of the line. – I also think that

there's something else, another benefit that comes along with this. You mentioned earlier today that the paper will not go away. There will be an opt-out clause option for citizens. But you have to think that most citizens and businesses and the government itself will try to be able to digitalize as many services as they possibly can. Security will be up, cost will be down. – But in the SEDI bill in the state, every citizen has a right to opt out.

– But I'm banking on the fact that most citizens will opt in. What will happen is that… – It's not opt out. It's opt in. – you have to opt in. It's a positive opt in. Sorry, I said it wrong, but you're right. – Okay. – They have to opt in. – They will choose to get a digital identity. – Yes.

And what will happen is demographically as people die off, the opt-inners will go up. And at some point in like one generation, you'll have a teeny fraction of the population that won't opt in. – Then as a byproduct, a natural byproduct as more opt-in, then the cost of services will go down. The ability to deliver services faster will go down. And there will be a net positive effect of the state being able just to be more efficient. And to the privacy and liberty, that should also be pointed out to the other states. – I don't know if people know this, but for the last 10 years, the state of Utah has been ranked as the number one state as a business friendly, well-run state. They get that running the state efficiently matters. It's not lost on them that having a digital identity system that allows them to run it even more efficiently, but protects the liberty of its citizens is part of maintaining their lead over other states in terms of how the state is run. And I'm bragging on my home state, but you just go look, which states are the best run states? The state of Utah is always one or two every year. Go ahead. – I think the adoption is going to be slow and then fast. And it's going to be a tipping point that I think will be like tap to pay. I remember I was a slow adopter of tap to pay. It took me a couple of years before I was willing to do that. I can't remember why if I had any good reason. But as soon as I switched, oh my gosh, I would never go back. Every place I go, I can pull out my credit card, but I moan and groan if I have to go through that amazingly difficult effort of pulling out a credit card. And I'm exaggerating, of course. But now my wife, she saw me doing it and she was a slow adopter. And now she's got her Apple watch and now she just taps her watch and just taps her wrist.

And now it's like whenever we see someone else pulling out a card when we're doing it like the right way, the cool way, the fast way. And now everyone around us is tapped to pay and I rarely see someone pull out a card. And it seemed like it was slow and then all of a sudden, but there's still always the right to still use the card, but the tapped to pay becomes the obvious thing to do. I think that's going to happen and I think it'll be a five to 10 year. – it'll probably be five to 10 years because we have to work out the user experience and it'll take a while just to figure that out. – I was asking you before regarding running this open infrastructure at the edge and the problem I was thinking to solve was this code stack. Do we have a plan B? Does this sound like a plan B?

Like growing this infrastructure more organically, more sovereignly? – Like for example, healthKERI doesn't need SEDI for it to be successful in the healthcare sector. Organizational identity will pay for KERI-like infrastructure in those things. It's just that it's a two-sided network problem. A startup has a lot of stuff they have to do to bootstrap that. It takes time. Right? That's always been the problem. – Let's say if we start bundling witnesses on watchers in our wallet. – absolutely. – And suddenly companies or organizations can tap these sorts of services. – if forward-looking companies do something like we're going to raise, if you have the ability to say raise an extra 10 million, right? And you say, what we're going to do is we're going to bootstrap by scaling, by hyper scaling, right? How you hyper scale is you give everything away, right? It doesn't cost anything for anybody. You just give everything away that you have a runway that says we're going to spend 10 million or in some cases 100

million. If you're an AI company, it's 100 billion that you're going to give away to bootstrap the adoption. And then once you reach the tipping point, then you can start selling tokens, right? ChatGPT didn't cost anything for how many years, right? A couple of years. And then all of a sudden it's like, oh, well, if you want really good ChatGPT, you're going to buy tokens, right? But they spent billions in those first two years. Right? If you have that kind of resource that you say, you know what? Digital identity that looks like this, that it's secure, that solves fraud, that does all those things. We have a business model that we'll be able to monetize. You're not going to monetize the infrastructure. You're going to monetize all of the services you build on top of that infrastructure. Then you would just give it away. You would just say what? We're going to go do what the state of Utah is doing. We're going to find 10 million people. We're going to give them wallets and we're going to give them witnesses and we're going to give them watchers and we're going to then, you Let them play with it. And they'll go, what am I doing with this? And they go, well, here's a place you can use it. And it'll be like the tap to pay. Well, maybe I don't want to go there. But at some point, people will find the utility and then all of a sudden it'll be like, why weren't we investing in your company three years ago when you were giving away millions of dollars worth of infrastructure and it looked like bad business model. But you got to have that kind of money and I don't have that kind of money.. And I don't know how to convince people to have that kind of money. But you're welcome. If you can do that, I say go for it. – I was thinking of our phones as infrastructure. – Yeah.

The way I imagine this is presenting credentials from my phone, not from my desktop app most of the time. You will be running this mobile wallet anyways, if we could expose this. The biggest problem is exposing the services running in your phone to the outside internet. Might be doable. – Every phone does have an IPv6 static IP address. You technically could run… – Snap networks using IPFS or … Technically you could do it. That's great. – I think the desktop would be a good thing for the CEO from our CEO's conversation. – Yeah. – If they had a nice desktop app. – I'm just offering. Anyway, before everybody leaves, should we close the session? Thank you.

TALK V

2026-11-02-samuel-m-smith-1

KERI Mini Conference November 2026 · talks · 57:27

SUMMARY

AI generated summary.

Samuel M. Smith opens on the KERI suite as minimally sufficient: control lives in a key event log, not a shared chain. Cryptographic agility retires broken hashes without rewriting the stack. Fault-tolerant key management keeps systems working after compromise; issuance and observation stay split so revocation is not a correlation point. That same infrastructure makes software agents accountable through a chain of authority to the liable party.

We start off today with Sam who's going to first talk about the state and the future of the KERI suite. All right. Thank you. You guys. And since we're recording, if you want to ask a question, we'll pass around the mic so that your question gets on the recording. We have a couple of mics. I recognize most of the people here, but there's some that I don't. Is there anybody here that has never heard of KERI before other than to come to this conference. Okay. Everybody's heard of KERI. Is there anybody here that just really doesn't know that much about KERI? Oh, come on. All right. A couple. Okay.

KERI is sometimes described as complicated and I like to say it's no more complicated than it has to be to achieve the properties that it provides. It's designed to be minimally sufficient to achieve those properties. But those properties are significant and difficult to achieve. If something is confusing or something like that, part of this conference is to try to explain those things. The KERI Foundation has been publishing videos of a lot of the talks. We now have a fairly big set of videos that you can go watch and learn from. And every time we do a presentation or a mini conference, we're attempting to add to that corpus of knowledge there. I want to talk a little bit about KERI.

2026, we think is a milestone year for KERI and ACDC and the KERI suite of standards and there's some reasons for that. Some things that have already happened this year. We first had the suite of protocols released as a Linux Foundation set of standards. We finalized the specs. It took several years to get to that point and some of the people that were instrumental in getting there are here today. Thanks for all that. The KERI Foundation was founded in 2025, but we received funding, both private and public funding to start building open infrastructure for KERI. We got a grant from the state of Utah Governor's Office of Economic Opportunity. The telecom industry is using KERI and ACDC to stop call spam. There is something called the Open Verifiable Communications Alliance and that is functionally in production. Microsoft has gone public and announced that they're using a software stack developed by company called Provenant to solve call spam.

That's just growing organically that eventually call spam will be a thing of the past. Five years from now, people won't be complaining about call spam. That's how significant that is. And that includes text as well. It's going to go into text. The Utah State passed a legislation, Senate Bill 260, that implements what's called state-endorsed digital identity. That's a first. No one's ever done that before. It passed both houses unanimously. It's a significant achievement. Lots of states have been trying to do various forms of digital identity and have been unsuccessful. The reason Utah State's passed is because the digital identity was based on constitutional principles as opposed to other principles. Some other things. There's a vital initiative in the healthcare industry and actually that's now deepened GLEIF and DirectTrust.

DirectTrust is the organization that vets health information service providers and also certifies certificate authorities and that issues certificates and identities in the healthcare industry. They are going to issue, they are going to become a qualified vLEI issuer. I can't… – Qualification agent. – Qualification agent, which will then qualify qualified vLEI issuers. This is an example of deepening and extending the organizational density Root of Trust that GLEIF provides that was the early adopter of KERI now to a sector that is the largest single market sector in the United States, the healthcare sector. And they've done some proof of concepts pilot projects and now have large payers like CVS I don't know if you know how big CVS is, but revenue is the same size as Microsoft. It's not worth as much as Microsoft because it's not a high-tech company, but that's how big it is. It's now starting to use that to connect payers to payers and participants in the healthcare industry. There was a summit earlier this spring at Utah State where a significant amount of intention was drawn to the SEDI project. We had the first ever KERI conference. Some of you attended it. Those who did attend it, you'll get some of what was at that conference today, much reduced because the KERI conference was two days with multiple tracks. This is one day with one track. We released open source infrastructure, witnesses, watchers, and wallets and continue to develop those. And I think and we'll be shortly releasing observers and registrars. This is the KERI MiniConf. And in two weeks in Utah will be the fall SEDI Summit, which will have about 250 participants. And there is an anticipation that there will be multiple states that will announce their participation in a multi-state SEDI consortium. Those are significant things that have happened just in 2026. 2027 looks to be even better. To give you a little bit of history, from my perspective, the journey down this KERI Suite path started for me back in 2013 and 2014 when I developed an open source, end-to-end encrypted and signed communications protocol called RAET, Reliable Asynchronous Event Transport. The insight that I learned from the development process is the key management was the problem. I could do all sorts of things, but the key management was always a sticky thing that I like we didn't really work. The key management, if the key management is not done right, it doesn't matter anything else we do. The protocol will fail. I then worked on something called the open reputation system, published a couple of white papers, and that's where I identified that persistent identity was the hard problem of identity. People were building and people still do. Most of identity is not are not persistent or perpetual identities. They're ephemeral identities and it's easy to build ephemeral identity.

It's really really hard to build persistent identity. I didn't know how to build it. I knew that I needed to build it and so I started down that path. And then I started publishing some white papers as part of Evernym and the Sovrin Foundation on this path to perpetual identity. That takes us to 2016, 2017, I call it the era of the ledger wars. Everybody decided that digital identity based on blockchain was the coolest thing. And one of the things that blockchain does is it does give you a form of perpetual identity. You can put the key state on the blockchain so that you can maintain control over an identifier indefinitely using the blockchain as your root of trust. The problem is everybody had their own ledger and everybody wanted that identity be on their ledger. And as great as the idea of Sovrin was, and we have founders of Sovrin here, everybody said, well, it's too great an idea. We don't just want one ledger for identity.

We want every ledger to have their own identity. And that meant that you didn't have interoperability. You didn't have portability and just created a host of new problems. And those are all governance problems. I call that the ledger war period. And I have lots of scars from that war because I spent several years fighting that war and actually GLEIF adopted KERI because they fought that war and they retreated off from the battlefield and said, we don't want to be on that battlefield. We want a ledger agnostic approach to identity. We have it here and we have Phil's here too. I pointed out Timothy, but Phil was the founding chairman of Sovrin. We have some of the original ones there.

The result of that was that in 2018, they started developing, actually solving the problem that identified years before, which is the hard problem, which is key management. Wrote a white paper on fault tolerant key rotation. And then started looking at building ACDCs with… if you had perpetual identity, then you could have perpetually verifiable issuances and ACDCs did that.

And published the first KERI white paper in 2019. The core result of that white paper was you didn't need blockchain to have perpetual identity. You could do it securely without a blockchain. And that was a heretical concept at the time. I remember every time I talked about that, the blockchain bros would beat me up and just say, no, no, no, you got to have a blockchain, right? I fought, I have a lot of battles from that fight, right? And then after a year of having the paper in the wild and getting lots of feedback and finding all the mistakes and bad things, I revised it and came out with version two, which is essentially what KERI is today.

There have been more incremental improvements since 2020, but the core KERI idea hasn't changed much since the version 2 white paper. And the version 2 white paper went through multiple iterations over the period of the next couple of years, but they were small things. Quantum Secure DIDs was a paper published in 2020, which is how we solve the ability to basically be protected from surprise quantum attacks. That provides surprise quantum attack protection, which means that we've pre-migrated KERI identifiers today are pre-migrated to be protected so that you can migrate to post-quantum secure things, but your existing identity is protected from a surprise quantum attack. In 2020, late 2020, GLEIF work began. In 2020, published a white paper on universal identifier theory which talked about how you solve organizational identity with KERI.

We started the Trust over IP (ToIP) task force for ACDCs in 2020. A lot of these protocols got started in that day. And then in 2022, GLEIF vLEI went into production. That was the origination of it. And then since then in 2023, I wrote a white paper called SPAC, which has now resulted in the trust spanning protocol. And we have Wenjing Chu, the co-author there, and it's now in final draft. We voted on it, right? It should go to spec. This is the fourth spec in the suite of protocols. Although usually you don't think of a trust spanning protocol as a KERI suite of protocol, but it solves the communication metadata surveillance problem, which is what you need to start using some of these things in different applications. 2023, as I mentioned, SEDI work begins. It took three years. I was appointed to the Utah State Privacy Commission back in 2023, and we spent time trying to figure out how to get legislation through the system. It wasn't an overnight thing. It's not like in January of 2026, some legislators showed up and said, well, I have a bill. It's called SEDI. Let's vote on it, right? That's a result of three years work of setting up the legislation.

And there are other people in the room that were essential in doing that. And then 2023 is part of that work I wrote a white paper on sustainable privacy. And then we formed the ToIP KERI Suite Working Group to get the specs finalized. The Cardano Foundation adopted KERI. We have some of the Cardano. Well, I guess they're Veridian now, right? Cardano has spun off a company called Veridian. They're here. And then in 2025 was the first legislation where we did digital identity rights. And then the KERI Foundation was established at the end of 2025. And then as I mentioned before, the Linux Foundation standards went final in 2026. That's sort of how we got started. A lot has happened and it's happening faster. The rate is accelerating and it's been it's challenging to keep up with it all. But everybody that's helped get it to this point, I think there's a real sign that it's going to stick and it's going to build because it does some things that nobody else can do. And I know we have some people in here

that maybe think other identity systems are great. I'm going to be biased. I'm going to think no other identity system is anywhere close to KERI and ACDC, right? But at least I'll make a good argument for it, right? Even if you disagree with me. We started with cryptographic agility from the beginning.

It had to be everything is 100% cryptographically agile at the granular level. I don't know of any other identity system that started with cryptographic agility as a first order property of the whole system. The CESR protocol is how we attain that cryptographic agility, which means that we're future-proofed. New crypto comes along. We don't have to start from scratch. Anybody's… I mean, the people behind Git and GitHub and the various organizations that use Git have a difficult problem because it uses SHA-1 which is broken. But there's so much code based on SHA-1 and there's no cryptographic agility that people have tried to move it to SHA-2 or SHA-3 and they can't because there's just the lift is too heavy. They keep patching it around it. But fundamentally, if they had built it with cryptographic agility, they could have just switched to a different hash and not have to change any of the tooling. You get that for free anytime you build stuff on top of the KERI suite of protocols. Cryptographic agility is baked in everywhere. Fault tolerant key management. The answer to security is not more crypto. The answer to security is not more sophisticated mechanisms and protocols. The answer to security is to fix key management. And the only way you can fix key management is to make it fault tolerant because you can't make it perfect. Every type of crypto has side channel attacks that are viable.

No matter what you do, you will fail at some point. The only way that it actually will work is if you make it so that it's fault tolerant. A lot of times this is a difficult concept. It takes a time for some people in the security industry to understand that you really want to build things with fault tolerance using first principle fault tolerance. A lot of people hear the term fault tolerant and a lot of people in the blockchain world are familiar with a brand of fault tolerance like Byzantine fault tolerance, but in the critical systems infrastructure world, in the reliable engineering world, in the survivable engineering world, these are worlds that I came from. I spent years in those worlds before I ever looked at identity. The building systems that are fault tolerant as a principle design… there are a well-known set of techniques. What I did is I took those techniques and I applied them to key management. And once I applied them to key management, all of a sudden key management became a solvable problem. The first solution is you have identity that is perpetual. And it's perpetual because under the hood, it's fault tolerant. If I have a secure enclave like on my phone and I store my private keys in the secure enclave, it's a piece of hardware. It's really, really hard to get to those private keys. But it's not impossible. If I don't have a way to recover from losing those private keys, my identity is only survives as long as the first exploit of my private keys. And then once I lose my private keys, I have to start over.

It can't be perpetual if I can't recover from a compromise. I have to build compromise recovery into my system. The very first thing we did is we assume in KERI that your signing keys will be compromised. It's not like the blockchain world where if your signing keys are compromised, you're done. It's a catastrophic failure. That's what it means. If you lose your Bitcoin address, your private key, then anybody can spend all your Bitcoin money and you can't ever recover it. You're done. Done. We can't allow, we can't for identity that is attached to a person, it has more value than the money in your bank account. It is you. It is your identity that you want to control perpetually. It has to be fault tolerant to the faults that are going to happen. The most likely faults and those are your keys are going to get compromised. Your signing keys are going to get compromised. Now that you have perpetual identity, you say, well, what am I going to do with it? Right? I have an identity. I can improved control over identity, but what am I going to do with it? How do you do things with it?

You issue things. People issue things to you. You issue things to other people. You present things issued to you to other people. You need to have issuances that have similar properties. You need issuances that are perpetually verifiable. And the interesting thing is that if I have a mechanism that makes my identity perpetual, I can use that same mechanism to anchor and make issuances perpetually verifiable. We got a two for one. It's like we built one set of infrastructure and it solved a whole other set of problems without having to build additional infrastructure. We had to build some additional infrastructure, but the core infrastructure we didn't have to rebuild.

And then we said, well, in the real world, if I'm going to do things with identity, especially in organizations, I need delegation. Delegation wasn't an ad hoc thing. Delegation is a first-order property of both KERI with delegated AIDs and ACDC with edges and delegated issuances. We have two types of delegation that can work synergistically. And they're not something that people like right now this year, I'm seeing people talk about delegation in all sorts of identity contexts. We were talking about delegation since 2020. We've been talking about delegation for six years and we have hardened the concept of delegation to where we know how it works, how to make it work and what to do with it. We're not just inventing it this year. Lots of people are reinventing delegation this year. I'm seeing lots of examples of, well, how do you do delegation? It's like, no, no, we've been doing it for six years. And delegation, the same mechanism that allows you delegation, allows you to do any kind of relationship. It doesn't have to be a delegation relationship. It can be any kind of relationship and that relationship can be provable, secure, verifiable, all of those properties. And we have six years of doing that. KERI is no half measures for security. We don't compromise on security. We go all in on we're going to make this as secure as you can practically make it.

Another way to think about what KERI and ACDC do, I call the four A's of data value, right? The whole point of doing this is to have data that's more valuable, to be able to exchange data that's valuable. The first thing in the four A's is it's authored. You want to be able to have a stable source of that information that you can, that that who authored that data. You want to then be able to authenticate the data and know that it came from that author. You can securely attribute it. That means that everything that everybody does in the ecosystem, they can be accountable. Now, you don't necessarily have to make it accountable to a person or an entity. It can be accountable just to the identifier and the identifier can have some degree of privacy associated with it, but it's still accountable to the identifier. And this is important when you get into the agent world because agents are these things that are operating, but who's accountable for what the agent does? Is the agent accountable? This question came up this morning as we were driving here. I noticed we passed, I think it was seven Waymo autonomous vehicles that did not have drivers, right? When I was here last spring, I didn't see that many. There was like seven just in like a mile. And Caleb asked me the question, so if the Waymo gets pulled over, who does the policeman issue the ticket to?

Who's the accountable party for that agent that's driving that car? If you can't solve the accountable party problem, you can't practically have agents in the real world because they're things. They're not people. Our whole legal system, our whole society is based on the fact that people are the accountable agents. You have to associate agents with the people that hold the liability. You have to make them accountable. Authorized, authority chains, accountability by reference, chains of authority. All of those things are part of making information valuable when you're talking about permitting things to happen with data. And then finally, authoritativeness. Can you trust the data? Is it reputable data? Is the information there something that has value? And often we can't reproduce the process of determining whether a statement has veracity. We can say, well, we know who said it, but we don't know if they're

lying when they said it. We just know who said it. How do we trust the data? One of the ways you trust is by reference, by reputation. If somebody that I trust says that you can trust Alice, if I trust Bob and Bob says, I know Alice, you can trust Alice, then I trust Alice by reference. And so much of trust in the real world, the bootstrap of trust comes by reference. For example, GLEIF is a global root of trust for the ultimate beneficial owner of a legal entity and the roles of the people that run the organization and the roles of any of the members of the organization. And you trust because you trust by reference. And if you can trust by reference, and you can securely attribute and do the other ones above it, then you have a basis for trust. […] – I just wanted to point out there's a recursive element here too because knowing that the reputation is trustworthy all goes back to the start from the authored part and goes back in. – you can't have reputation if you don't have the other ones. And then once you have reputation, then you can start to trust the authoritativeness of the information. Like I have a PhD, right? And so you would say, well, I can trust you because you have a PhD. Why can you trust me? Because the institution that granted the PhD was accredited to grant PhDs. Now, if you said, well, I'm going to tell you how to, I have a PhD.

I'm going to tell you all about how to make explosives. And you might go, well, wait a minute. What's your PhD in? I say, well, it's electrical computer engineering. I don't think they teach electrical computer engineers how to make explosives. That would be like a chemical engineer. Already the authoritativeness of what I say is a function of the reputation that tells you whether or not there's some basis for trusting me. Right? Without this ability to security attribute up the chain to have reputation and systems of reputation, we don't have a basis for trusting data that comes in over the internet. And we certainly can't trust agents that are operating electronically without that. We have to solve that problem. The mission of the KERI Foundation is to foster open infrastructure. It's the I in KERI. There's a reason KERI is infrastructure.

Because to solve the perpetual identity problem in a fault-tolerant way requires infrastructure. There's no way to solve it without infrastructure. But we want to minimize that infrastructure. We want to make that infrastructure work in a certain way. I use the phrase, I use the acronym DKMI for decentralized key management infrastructure to describe what KERI is. KERI is a decentralized key management infrastructure. It's a set of protocols that allow you to do that. And it was designed to be web scalable to solve the cost floor problem of blockchain.

They said, well, blockchain is way too expensive. The cost floor is way too high, right? We want it to be web scalable. We want an ultra low cost floor because it's infrastructure and infrastructure costs money. It's not free. You want to at least use infrastructure that's as cheap as the cheapest possible infrastructure right now is web scale infrastructure. That was a design decision because we couldn't get away from it. A lot of people have complained about KERI and said, well, it's too complicated. You got to have witnesses. You got to have watchers.

That's too much. I want an identity that doesn't have that infrastructure. Well, you can have an identity that doesn't have that infrastructure, but you can't trust it. It's not secure. It doesn't satisfy the minimal properties. You either decide that you're going to live in a world where you're hacked continually or you're going to build the minimal amount of infrastructure that need to be secure.

And I can't think of any less infrastructure than a web-based open source open standard things that run at web scale. The idea is that we have a cooperative model so we can get meta network effects. I wrote a white paper on meta network effects which says that if you're sharing value that transfers across trust

domains that you get a meta network effect of that value. Reputation and the four A's of data, all of those things are about secure attribution, identity. Those are all things that move across trust domains. It doesn't matter. It's like I use this example.

Reputation, some really good people have talked about reputation as being three things. It's competence, honesty, and reliability. If I can trust somebody, I have three reasons to trust or not trust. Somebody can be very competent, but if they're not reliable, I can't trust them. Somebody can be very reliable, but not competent. Like I have a younger brother who's an engineer. He's really reliable.

He's really competent in engineering, but I would not trust him to do open heart surgery on me. Now, he actually has a son who's a doctor that I might trust to do open-heart surgery on me. He's not quite yet. He's just barely out of residency, so he hasn't reached that point yet. And then you have people and a lot of people think, well, it's are they dishonest? Dishonesty is only one part of being able to trust. What you want is those things that move across trust domains.

If I hire somebody to mow my lawn and they're reliable, like they show up on time, they mow it, don't leave weeds, stuff like that. And then they come back at the end of the season and say, hey, I'm starting a business where I'm going to be shoveling snow. Okay. Right off the bat, I know that they're reliable. They may not be competent at shoveling snow, although it doesn't take much to be competent at shoveling snow. Would I trust them? I would trust them a lot more than somebody who was not reliably mowing my lawn, right? It's not a single property that we have to worry about. But there are lots of things like that do move across domains and a lot of them are identity-based. You can reconstitute your control of your identity from any copy of your key event log. And I haven't gone to any detail inside the things. But this is an important property because it means that as long as there's some infrastructure out there and you lose all of your copies of your key history, as long as somebody has a copy, you can reconstitute it. That makes it portable, right? It's truly portable. There's no wallet vendor lock-in. We have true portability across all the infrastructure providers. We have non-cooperative key custodianship. A lot of people that say, well, keeping my private keys is too difficult. Grandma can't manage your private keys. If I have a keys custodian that's managing on her behalf, I want to be able to have that management happen and be able to move to a different provider without requiring the cooperation of the original provider. If for some reason they die in a snowmobiling accident and they lose all of my grandma's keys, I can still move to a different custodian.

Witness pools with no common mode failure sources benefit everybody that uses them. And then the watcher network makes it so that you can tag duplicity. It was an interesting thing. I used the term duplicity, but that was the biggest aha moment in writing the KERI white paper is that if I can make every fault exhibit as duplicity, then all I need is one mechanism. I need a duplicity detection mechanism and now I can detect all faults. And the key to fault tolerance is you have to detect faults. One of the in my early days, I attended a seminar by the original team that was working on the Mars lander. And we were talking about they were talking about a fault tolerance. Like the seminar went to a conference and I was working on autonomous underwater vehicles, which aren't as cool as a Mars landers, but they're still pretty cool. Right. Anyway, I was trying to learn from them and they said and so the whole thing was about fault tolerance. It's like how to build fault tolerance systems, how they put fault tolerance in the Mars lander. And the key insight they said is we realize that we just can't build any system that we can afford to put on a rocket and ship across space that is imperfect, that is impervious to failure and impervious to fault. There's always going to be faults that we can't account for. We just can't build them all in. But they said, but what we could do is we could build a system that never failed to detect a failure. We can't prevent it from failing, but we can certainly always detect the failure. And then if we have

enough time and resources, we might be able to recover from failures. And I don't know if you pay attention to the early days of the Mars Lander, but they recover from pretty severe failures. They had a botched software upgrade and the system just like didn't work. And then they figured out that if they did some things, stuff like that, they could boot it in a particular mode where they could download a recovery program and they were eventually able to recover it so the point is detect failures.

Build failure detection as your first and most important piece of infrastructure. Then you can build on top of that. Duplicity, once I only had one thing I had to detect, then failure detection got easy. I just have to detect duplicity. I force everything to look like duplicity. KERI is designed so that every bad behavior that an attacker could successfully benefit from exhibits as duplicity. The duplicity detection then detects it so I can recover from it before they gain full control over my identifier. We have something called the five W's and we're going to have some more talks about that. These are shorthand for wallet witness watcher. Those are the three primary infrastructure components. We have the web runs on the web and the wizard is smarts behind the scene. Now we have two other ones that didn't fit in the five W's. These are registrars and observers, but we'll say that they're part of the web infrastructure and we'll talk about those. That's what KERI infrastructure looks like. You see that it's always two parties, You have entity A and entity B.

Each party has their own set of infrastructure. The infrastructure that's shared is the watcher network. Entity A controls their own witnesses. Entity B controls their own witnesses, which means you can't trust them. The watcher network observes duplicity and therefore you can trust them because you only trust them as far as they're not duplicitous and as soon as they're duplicitous you stop trusting. It solves the governance problem because in blockchain the watcher network and the witness network are the same network. You then have to have shared governance between entity and A and B. And if they don't agree to shared governance, then you have the ledger wars. We got rid of the ledger wars by separating witnesses and watchers and separating the network that manages trust into two pieces, one that's controlled by the entity and one that's not controlled by anybody. It's just anybody can be a watcher and all watchers do is detect duplicity. They don't have anything else that they have to do.

Any watchers is good. The watcher network, is mutually benefits everybody. And nobody, nobody has to, nobody cares about how it's built because it doesn't benefit them any more than anybody else. That, that was the, that was, so we call that deconstructed blockchain.

I told that to some people and they go, Oh, okay, that makes sense. Now we deconstructed the blockchain into a, into two different networks, a witness and a watcher. Then we have mobile wallets. We have some wallets. We'll do some demos on those that run on the different platforms. And then we have the other two are registrars and observers. And these are necessary for perpetually verifiable issuances. And in the SEDI example, they're essential become essential participants registrars for ACDC state. And the observer is how you isolate the registrar from the points of validation so that you don't have any surveillability back to the issuer. We solved the privacy problem, but yet we still have revocation registries that aren't correlatable because of that separation.

We also did a split, right? We split the registrar which is controlled by the issuer that publishes state and then the observer which is controlled by the verifiers can observe the state, but the issuer can't observe the verification action.

It also deconstructed blockchain because those used to be on the same network as well. And then this is the process of presentation. An issuee gets an ACDC, they present it, the verifier goes to an observer who has the ACDC state from the registrar. They can verify whether the state of the ACDC at the time of

presentation is still valid without creating a correlation point. There are lots of systems that when you go to revocation registries, you lose all your privacy because the revocation registry becomes a super correlator. And there was a lot of work trying to build revocation registries using cryptographic accumulators, but nobody's after Sovrin, nobody's really done anything serious with that. Lots of people just now use like a like the W3C is their standard is a bitstring status list, which is perfectly correlatable. Is there a problem with cryptographic accumulators or just too difficult? Too difficult. It's just they're just too difficult. The performance, I mean Sovrin spent years trying to get the performance to where it was reasonable. Technically a cryptographic cumulatively would work, but practically they just it just never worked out. I said, well, I got to do something that's web scale that works cheaply. Separating into pieces actually solves the problem in a and we get something that nobody's ever had before. In fact, I'll do the next slide.

With bulk issuance, we can get herd privacy with sparse Merkle trees, with bulk updates, We can get herd privacy and bulk issuance allows you to together it solves the vendor to vendor linkability problem. But here's a cool thing that we came up with in the specification. And I call it perfect forward privacy. By going to this registry approach, which you can't do with a cryptographic accumulator. This is better than a cryptographic accumulator is that you can blind the state of the ACDC so that the presenter or holder can unblind it, but they unblind it at presentation. And then you issue the issuer can issue an update that's a vacuous update, no change, right? You can have a no change update. It doesn't change the state, but because the state is blinded, the verifier that verified the previous state doesn't know the current state anymore. Whereas a cryptographic accumulator once they know how to verify the state, they can verify it indefinitely. That is a unique feature of this approach that we use now with KERI is that you can present the state, the verifier can verify the state today, but then tomorrow they don't know what the current state is. All they know is there's been an update to the state. It might be an update that doesn't change state. It might be an update that doesn't change state. It might be an update that does change state. They don't know. The only way they can know is then to have a new presentation with a new unblinding that tells them what the current state is.

It solves all the privacy problems associated with dynamic updates. It doesn't become it's a really good solution, I guess is the way I'm going to say it. But we went even further because remember I said the key to key management is detecting faults. In the conventional world of verifiable credentials, there's no way for the user to detect whether or not their keys have been compromised and an imposter is now presenting their credentials fraudulently.

Like if the keys that I use that control the identity that an issuance has been issued to and I do the presentation and I sign it. I sign that presentation and they go, okay, yes, it was signed. At the time of presentation, whoever's presenting it controls the keys that control the identity that to which it was issued.

We'll trust it, right? The verifier has to trust it. We can do better than that. We can make it so that an issuance requires a presentation registry. It's a registry just like an issuer registry, but the registry in this case controlled by the holder, not by the issuer. When the holder presents the credential, the verifier has to check the user's presentation registry. It has to check two registries.

Sorry. It has to check the issuer registry to know that it was correctly issued. But it checks the user's registry to see that the user has actually anchored it in their KEL, the presentation. What that means is now the user can check their KEL and if any presentation show any anchor show up in the KEL that they didn't in this registry that they didn't approve, they know their keys have been compromised. Now they can recover from it. They can limit the damage. You don't have unmitigated impersonation fraud

potential based on key compromise. You can you force the detection of the key compromise by the use of the keys result in something that you can detect. You can't stop the key compromise, but you can make it so that anybody to benefit from the key compromise does something that you can detect. And that you need to fix your keys. Here's how it looks like.

Here's the issue has a has their key event log. Here's their registry. Each time that there's a presentation, they anchor the presentation in this log which gets anchored in their KEL and if their keys are compromised and somebody is able to anchor in this log in order to do a successful presentation, the holder can detect it because this has to be anchored in their KEL. They can't put it here unless they can get it in their KEL. And if they get it in their KEL, then they know their keys have been compromised because the only way it gets it in their KEL is their keys are compromised. And the witness pool allows them to detect that because they're forced to get it in the KEL. They have to get the witnesses to witness it. These are things that don't happen in key management. We got a couple minutes left. Right. I'm going to I'll cover more of this stuff, but these detailed fault tolerant things don't happen in key management in general. When you ask people, how do you manage keys. They'll say things like, in fact, I see this in almost every protocol that is in the identity space. You go to the security section and they say security considerations. And what did they say in the security considerations? We assume that your private keys have not been compromised. We assume that. If your keys are compromised, then all sorts of bad stuff happens and this protocol isn't secure. They already assume away the hard problem. They assume away the problem that makes everything insecure. They just say it can't happen. And if it doesn't happen, then everything else we do is secure. Great. That's the easy problem. The hard problem is how do you still survive even though your keys are compromised? That's always been the hard problem identity and nobody has solved it until KERI did. That's our claim to fame.

We solved the hard problem of security because we solved the hard problem in key management. I try to come up with a simplified way to think about KERI. KERI is about reputable autonomic pseudonymity. Big words, but if you remember the words and you remember what the words mean, then it translates.

Reputable means that the identity is perpetual. You can attach reputation to it and the reputation sticks. You can trust it. Autonomic means the user controls it, right? Self-governing. You the user controlled identity, not some other entity. And pseudonymity means that it's an identifier. It isn't your name.

You can have some degree of separation between the identity and the natural person or the identity and whatever. You can have as much privacy as practically possible, right? It's pseudonymity and you use a cryptonyms. The other side is control over confidential context. This has to do with what do you do with this side? If I want to exchange data with somebody, I want to be able to have a high value exchange. I want that exchange to have the four A's.

Author, authenticated, authorized, and authoritative. How do I do that? I want to do that in a way that two of us can exchange data like that, but we don't want any third parties to see that. I need a way to control a confidential context. Now people say, well, that's privacy and it's not. There's two definitions of privacy. One of them is really intimate knowledge that only you know. When I say private key, that's using the true definition of privacy. It's a private key, only you know it, you never share it, right? The other definition of privacy, which has changed in the last 20 years since we had the internet, is really confidential. I'm exchanging data and you're not going to share it with somebody else. Confidential is a very clear meaning. You share data with somebody. Once they have the data, they have the data. I can't stop them from knowing what they know. And if they share it with somebody else, I can't stop that. There's no way for me physically, technically to stop that.

There's no technical solution to the problem of sharing data. Once the other person has it, they can share it. I can't technically stop them. I might be able to stop them with liability, with regulatory, with incentive structures, with taking them to court. I can do all sorts of things to try to prevent them from sharing because I de-incentivize it, but I technically cannot stop them. Once they know something, they can share it with anybody they want. There's no technical solution to the privacy problem if privacy includes sharing data. You can technically solve privacy if it means not sharing data. I can make it so I don't share data and I can protect it so nobody else can see it. But once I share data, I can't prevent them from sharing it. Technically, I can only do it regulatory ways. Yes.

I call this control over confidential context. Anyway, the shorthand for this is RAPC3 or Rhapsody. My only comment Sam was if you had a mechanism in the data you shared that required some type of key that you could rotate to access it you could share that data temporarily but then change its access after the fact. Couldn't you not? Well if they decrypt it and copy it then you can't stop that. Right but there is a window. But like I said it doesn't fundamentally prevent the hard problem which is they have data, they know the information, they can share it in a different form. The information leaks out, somebody else can correlate it, re-identify it, all of that works. There's lots of little tricks that people have tried but they don't fundamentally solve the hard problem. And if we think of it in the wrong way, then mentally we never think about the right solutions because we're always thinking about the wrong solutions. More questions? Yes.

What do you think about zero-knowledge proofs? There are lots of different types of zero-knowledge proofs. Technically, a zero-knowledge proof is any proof that discloses some data without forcing you to disclose other data. A Merkle tree is a zero-knowledge proof. A digital signature is technically a zero-knowledge proof. Any every you know just standard digital signatures are technically zero-knowledge proofs because you have a private key and a nonce, you can prove that somebody who knew the private key and a nonce created the signature without disclosing the private key or their nonce. That's technically a zero-knowledge proof. Now, there are families of zero-knowledge proofs called anonymous signatures, anon-creds, things like BBS+, which attempt to make it so that you can prove that something was issued by somebody without the issuer knowing at the time of proof. And those are only good for ephemeral applications because as soon as the issuer's keys are rotated, you can't trust any of those proofs. They've been misused. When used appropriately, they work just fine. If I have a situation like let's say a blind auction where I issue the permissions to be in the auction the day of the auction and they expire the next day, perfectly fine. Because I expect that those keys that I used to issue them are going to be strong and not compromised for a day. But if I'm issuing a driver's license, or a birth certificate that has to survive for years or a whole lifetime and I'm using an anonymous signature to do that, the issuer's keys aren't going to last a lifetime. That means every single issuance is automatically invalidated when you rotate keys. Not when it expires when the keys rotate. And what determines key rotation? It's something called a crypto period. A crypto period is a statistical concept. But for example, if you read a special publication 857, from NIST, one of the things that determines the maximum crypto period for a set of keys is your IT staff turnover. Let's say you're an issuer and you're issuing a bunch of driver's license or a bunch of birth certificates and you hired an IT staff and they have access to the private keys and one of them says, oh, I don't really like working for you. I'm going to leave. And it's like a week after you issued a million birth certificates. What are you going to do with those million birth certificates? If you're going to be secure, you would have to revoke them and reissue all a million because that one guy left because he had access to the private keys used to issue them and therefore he could issue fraudulent ones and you have no control over them once he walks out the door. The idea of using those sorts of zero knowledge proofs for long-term issuances was always a bad idea. And even

though I would argue with people about it, they'd say, oh, no, no, it's not a bad idea. It's because it's a bad idea for key management reasons, not because of the algorithm. The algorithm is just fine. I don't care about it. The algorithm works as an algorithm, but the key management assumptions don't work. In fact, the W3C BBS plus standard in its security section, you know what it says? If the issuer keys are compromised, there's no security. It says that. They assume already that your key that the issuer's keys can never be compromised during the lifetime of any of these proofs. And then they never talk about what does that mean practically for somebody to issue proofs. It means you can only use them for a ephemeral applications. Those type of zero knowledge proofs. Now there are other ones. There's SNARKs, the STARKs, all of those all have different properties. They have different applications and they work really well in those applications. Often what happens is people take them over there and say, here's some magic. Here's some black crypto magic. We want to apply it to identity. And they ignore all the assumptions that say how you can use it.

They ignore those assumptions and they're often key management assumptions and they apply in a domain which is identity where none of those assumptions are valid. And so it's not because the crypto is bad, it's because the assumptions don't apply to the application and so you can't use it in that application. That's what I think is zero knowledge proof. I hate it when people say Sam doesn't like zero knowledge proofs. I love zero knowledge proofs when they're used correctly, not when they're used incorrectly. They're really, really bad and just because they have a really cool name like zero knowledge proof, you shouldn't be using them just because it's a cool name. Right? That's a bad idea. Anyway, are we done? Okay, I'm done. Thank you.

TALK VI

2026-11-02-samuel-m-smith-2

KERI Mini Conference November 2026 · talks · 57:28

SUMMARY

AI generated summary.

Samuel M. Smith links Utah's SEDI public identity utility to AI anxiety: automated attacks win through side channels, and continuous identity becomes continuous surveillance. KERI's answer is nested threshold structure — separate signing and rotation thresholds, delegation, and witness pools — so identifiers stay secure without surveilling people. SEDI credentials are registered ACDCs anchored in key event logs; guardianship is ordinary delegation.

Okay, this is a really long title. What I wanted to do is I wanted to take something that's happening in the SEDI world and relate it to what has now become people call it AI anxiety, right? In fact, I had my daughter send me a text and say, so when the world ends because of AI, what should be our emergency plan? And then they live a couple miles away and it says, how do we communicate? How do we get food?

What do we do? And I'm fully off-grid capable because I have solar panels and batteries and stuff like that. And I have a swimming pool. I have years worth of water and a bunch of stuff. I just said, no just get from your house to my house and you're good. Right?

You got two miles, you got to fight off the zombies for those two miles, but otherwise you're good. But it was just because of all the AI anxiety. This is a little bit to address that problem. What is the AI surveillance problem? Well, people have been using this term and it's now gotten to be a bad word called “continuous identity”. What is continuous identity? It means that I determine who you are by continuously surveilling everything that you do. And if I continuously surveil everything you do, I can build a persona of your behavior that an AI can't deep fake or can't deepfake today, but might deepfake tomorrow, which means the level of surveillance has to get more pervasive. The assumptions always is there are things that I can permit somebody to know about me that an external AI attacker can't get to yet. As long as it can get to stuff that they can't, then it can build a higher fidelity surveillance model than the AI can.

The AI can't deep fake me. Like, does the AI know my weight? Right? It knows my image because my image is everywhere. Right? But does it know my weight? Well, if I have a monitor in my bathroom scale that's measuring my weight every day and that monitor is air-gapped, then I could build a model of me that had my weight data that the AI can't fake. Right? I have to securely present that data and then they go, oh, okay, well that's within the range of what we've been measuring over the last year for your weight. We'll trust you. That's the example of what you have to do to make continuous identity work is that you have to be able to surveil stuff that the attacker AI can't surveil, at least for now. And then you can build a better persona than your wallet or whatever it is that you're authenticating against or whatever host that you've allowed to surveil you presents on your behalf, right? Does this scare anybody? Is anybody worried about this? Okay. All right. Another term they use, which is a little friendlier term, behavioral biometrics. It's just another type of biometric. It means we know everything about you and that is your biometric, right? It's a behavioral biometric because it has nothing to do with your DNA or your skin or your fingerprints. It has to do with everything else about you besides those,

right? Another friendlier term, participatory identity. You participate in your identity by letting your identity provider know everything about you. Isn't that fun? Persona-based authentication, right? That's the more technical term.

It's essentially authentication via statistical correlation. You basically use all of data science to statistical correlate more about you than the attacker can correlate, right? If you think about what authentication factors are. Things things you have, things you are, right? They're just data points that you can present but that the attacker can't present, right? But since all of the standard authenticators have been very easily hacked recently, right? Then it's like, well, how do we make authenticators that are harder to hack? And if you're a company that lives on correlation, what your natural thing is, well, wait, if we could just surveil everything about you, we can build a really high fidelity persona that the attackers don't know as much as. And guess what? You can't do anything or go anywhere unless you share everything about you with us because the way you authenticate is to ask us to present this persona that we've generated from surveilling you. Does that feel like a world you'd like to live in? But you understand that that is the world we're in right now. You go look online and you go to security companies, look at the biggest billion dollar security companies and guess what their new products are? “Continuous identity”, “behavioral biometrics”. That's what they're selling. Everybody you want to go to, this is well, you know what? The AI can fish everything. They can attack everything. What we need to do is have a counter AI that fights against it. And what does the counter AI use? What do AI's use? Correlation, right? That's how they work. An LLM is just a correlation machine. You build a model that correlates you better than the deep fake can correlate you. That's all it is.

Guess what that turns into? The more pervasive surveillance, the higher the fidelity of the model, the more likely it will distinguish a real person from a fake. But that just means you've created an arms race of surveillance. You've basically given yourself a way to just being surveilled to guess what's going to happen. You're going to start having monitors in your body monitoring things that are essentially somebody in North Korea can't monitor because they have to attack, but in order for it to be useful you have to un-air-gap your body in order that the AI that is creating your model so you're actually weakening your ability to protect yourself because you're un-air-gapping everything about you. It just becomes an arms race the more they know the more your AI needs to know. And guess what? Any breach of your AI immediate makes the attacking AI that much smarter. Guess what? You have to be even more pervasive in order to have a delta. It never ends. You just become perfectly surveilled. And once you're perfectly surveilled, there's nothing left. And then nobody has any security. The end state is zero security.

But somebody will make billions of dollars or trillions of dollars in the process. – I thought there were just more and more turtles. – No, no. At the bottom, there's an egg. There's an egg and once that egg's cracked, no more turtles. All right.

The solution cannot, must not be more surveillance, but that's what the world is telling us. The only answer to AI cyber attack is AI defender and it's an arms race of surveillance because they're just surveilling yourself. Here's some facts that we can use to fight back. Strong crypto is not vulnerable to brute force attack. It is computationally infeasible. It doesn't matter how big the AI is. It doesn't matter how many servers there is. Strong crypto is designed so that a trillion servers operating for a trillion years can't break it. Does that make sense? That's what computational feasible means and AI doesn't have any advantage in that world. All of the atoms in the solar system would have to be computing

machines before you'd start to reach that level of 120 bits of cryptographic strength. How does AI work when it attacks? It's not brute forcing any of the crypto. That's not none of those attacks are brute forcing the crypto.

Every once in a while somebody will publish a thing that says, oh, AI cyber defeated some crypto algorithm. And then you go read the fine print and no, no, it was a side channel attack. They didn't actually brute force anything. They didn't invent a new attack. They just figured out a better side channel attack. Right? What is a side channel attack? Everybody familiar with the term? Do I need to find that? It's any other attack besides brute force essentially.

Phishing is a side channel attack. If they phish your secret, that's a side channel attack. If there's a bug in the software running on your phone that allows them to get access to your private key, that's a side channel attack. They didn't brute force the public key to get to the private key. They just found the private key because you had bad software. – The policy engine emits its own credentials.

Yes, something like that, right? Or an IT person walks out of the building with the secrets and shares on the internet, right? Those are side channel attacks. Or for example, there was a recent crypto wallet that had multisig. By the way, multisig has never been successfully attacked other than through code supply chain attacks or bugs in the wallet. If you eliminate wallets that have bad bugs, attacks that can successfully do like a three of five where they have to get compromised three keys or even a two of three. They don't, there are no published examples where they're successful to do that other than a code supply chain attack. A recent crypto wallet and exchange had a successful attack and here's the attack. The three people that had the keys, it was a two of three, right? Got an email that said, Hey, there's a bug in your wallet.

Click on this link and upgrade your wallet to the new one that fixes the bug. All three of them clicked on the link. And what did the link do? It downloaded a wallet that exposed their private keys. Okay. It wasn't an attack on the strength of the crypto or a strength on the multisig. It was an attack on the strength of the gray matter between the ears of the holders of the keys. Right?

That's a side channel attack. If it comes in here and comes out here, it's a side channel attack. Right? All right. Okay. What's the solution? If I have key management that detects and can recover from successful site channel sites, because guess what? People will be tricked. You can trick people. You can cause them to do things, right? Even experts in moments of weakness, in moments of stress can be tricked into doing bad stuff. I can count probably half a dozen times were in a moment of stress or worry or hurry or something like that, that I've accidentally clicked on a link without thinking about it and almost compromised did something. And then I go, wait a minute, that's not right, right? And caught myself. And there's been a couple of times where I've lost like a couple hundred bucks or something because I did something stupid, right? It's not hard to trick people into being attackable with side channel attacks. If you want to actually protect it from AIs, you have to allow the fact that the AI is going to be successful in side channel attacks. What you want to be able to do is detect those and be able to recover from them to limit the damage or limit the blast radius or limit the extent of the attack so that the damage is minimized, right? Right?

If I have multiple complementary threshold structures that have combined with the detection and recovery from side-channel attack, then I can make a system that is practically unassailable from AI cyber. That's a very strong claim, but I will back it up. Right? There is an answer and it's not continuous surveillance. It is just key management. If we do key management well, AI cyber can't attack us, can't attack you, can't attack anybody. There is a solution and it isn't spending more money on correlation

machines and super aggregators who want your money and your services and all of your data. And they will say, well, give me all your data and then you will be secure. But I'll know everything about you and I will sell you to the highest bidder. What does that look like? Wait, wait.

Oh, that's just sorry. I forgot to delete that slide. That's a duplicate slide. Okay. Threshold structure. Okay. What is a threshold structure? I use that term. A threshold structure multiplies the number of attack surfaces that must be simultaneously compromised for a successful attack. If you think about normal armor, right? It doesn't have threshold structures. It has a helmet. If I breach the helmet, you're dead. It has a breastplate. If I breach the breastplate, you're dead. Okay? That's not a threshold structure. A threshold structure would be a shield and a breastplate. I have to pierce if the arrow comes through the shield, it has to pierce the shield and pierce the breastplate at the same time.

If it's over here and the arrow comes in that direction, it goes through the shield, I'm not going to do this, right? I'm going to go, oh, wait, there's an arrow in my shield. I guess I need to break it off before I use it, put the shield in front of my breastplate. You see the difference? In other words, the arrow has to go through two in order to kill me. It can't just go through one. Right? An example of threshold structures. Okay, I'll get to examples. A fault-tolerant system. When I use the term fault-tolerant, detects and recovers from a partially successful attack, therefore reverting the attack progress. Let's say that they shoot an arrow at me. It goes through my shield. Right? But my shield is out here. I got this arrow sticking out of my shield, right? It didn't happen to go through my arm. That would have been bad, right? But it went through the shield. If I break the arrow off, pull it out of my shield, I reverted the attack progress. The only thing now is I have a hole in my shield. If they can put an arrow through the same hole, then I got a problem because it'll probably come right through and hit me, right? But otherwise, they have to start the attack over again, right? Because they have to go through both the shield and my breastplate to kill me. As long as I can nullify the first attack before the second attack is successful, then they have to start over, right? That's the whole idea is that you reset their effort to attack. It's like, okay, we got 90% of the way, but we failed. Now we have to start from zero and go all the way again every time that it's detected and recovered from. What you have with a fault-tolerant system is potentially a perpetually secure system. It's never breached ever because the cost of a breach is too high for them ever to fully succeed in a breach. And that's the only type of security system that is going to be AI resilient. It has to be perpetually secure. That means it has to be fault tolerant. In general, security systems are not fault tolerant. They don't detect and recover from attacks. They don't even detect attacks. Usually the attacker says, Hey, by the way, six months ago, we breached your system. We exfiltrated all your data. We've now gotten all the value out of it. We just wanted to let the world know that we breached you so that we can get higher ransom next time because our reputation goes up. That's how most breaches are discovered. They're not discovered by the people that are breached. They're discovered because the people breaching them tell them that they were breached. The average time to detect a breach is six months. The average time to contain a breach is two months or three months. You have nine months of time where the attacker can benefit from the breach. That's not a good thing to have. And that's why people can legitimately say, well, the only solution is continuous identity because we have to do something about that. An allowable fault is a side channel attack on signing keys. In KERI, we say side channel attacks on signing keys are allowable. Our system is fault tolerant to those faults. We can detect and recover those faults. Loss of control over the identifier is the system failure which is not allowed. If they gain control, of your identifier, then we call that a failure. You have to start over.

We want to make that really, really hard. And if we can do that, then we can minimize the damage. How do we do that? We do that with threshold structures. What are the threshold structures? Well, we talk about this a lot. Signing keys and pre-rotated rotation keys. That is a threshold structure. It's like the shield, my signing keys, my breastplate, my rotation keys. They don't get control of me. They don't kill me until they breach both of them at the same time. They breach my signing keys. Great. I can throw the shield away, get a new shield. Throw the shield away, get a new shield. Every time they have to start over, right? They never get to the breastplate. Why won't they just attack the breastplate? I can protect my rotation keys much better because I don't care about friction, right? If I'm signing stuff all the time, friction matters. It's not usable to the user if it's too hard to sign. If I said to you in order for you to sign to log in to your account at your bank, you have to go to the bank, go to a safe deposit box, pull out your safe deposit box. Inside your safe deposit box is a sub-safe with some keys on it, take that home, open it up, find your login and then log in to get to your bank. Okay. That's too much friction. But it would be a hard for an attacker to attack that.

because one it's air-gapped. Two, they have to be physically present. They have to impersonate you physically. They have to go to your bank. They have to be able to get into your safe deposit box all before they can log in. Is that really, really, really, really secure for almost all the attacks on your identity? Well, unless you're a really, really high value individual, that's probably more friction than any attacker ever wants to execute. Right? Some attacker in North Korea or China is not going to come to your bank. Right? Unless you're worth a lot of money and then they won't come to your bank, they'll hire somebody to come to your bank. Right? But it still requires somebody to come to your bank. And that means that somebody, when they fraudulently impersonate, you could get arrested and go to prison. Right? The cost the attacker is potentially to be imprisoned. How many hackers want to be in prison? Why are most hackers in foreign countries on the internet? Because they're not liable to be imprisoned because they're not in the jurisdiction of the place where they're attacking. I can make the pre-rotated keys as secure as I need to because the friction of recovery only occurs if they're a successful side channel attack. There's a differential that allows me to make it so the overall security is really, really, really secure and still have low enough friction that it's practical to use the system. Okay, multisig signing keys. I already mentioned the multisig is really hard to do. I can have two of three. Now they can phish one set of keys, but they got to phish another set in order to get to me. And if they try to use the one set, Well, if my system's designed right, it might be detectable that they're trying to use one.

I know that one set of keys has been compromised. I can do multisig for my rotation keys. I can have single-sig signing keys for low friction, but I can have multisig rotation keys when I need to recover. Now they can say, well, okay, we know that you have a rotation key here. Oh, but you have two or three rotation keys and the other rotation keys over there. Now we got to attack two different places. The return on our investment isn't good enough there. Delegated AIDs. A lot of people don't understand that. A delegated AID is a threshold structure. If I have a three-layer hierarchy, each layer is a threshold structure for the lower layer because I can compromise the lower layer, but if they don't compromise the layer above, the layer above can recover the layer below.

And if I compromise two layers, the top layer can recover both layers below. They have to compromise all three layers at the same time. If they only compromise one layer, it's detectable recoverable. They have to start over every time. Now, if you're an attacker and you go, okay, I'm going to spend all this time and efforts. I'm going to compromise one key. And then as soon as I try to benefit from that compromise, it becomes detectable. And then they're going to rotate me away.

They're like, crap, I've got like literally a day at most, not nine months, a day, maybe an hour where I compromise a key I can do something with. And as soon as I do anything with it, they're going to detect it like the minute, the second I do something with it, like seconds, network propagation time, they'll detect it. And then it's however long it takes for them to figure out to recover from it. Then they're looking at, well, wow we can't it's like the attackers that compromise Google in 2025. You guys all know this, right? That every Google email account at Google was compromised in 2025. All five billion of them. Everybody that has a name, an address, a phone number, and any other data, they didn't compromise your password, but they compromised the account data. Now they know who you are. They know whatever information else that Google has. They can try to phish you, try to impersonate you, whatever. But all five billion of those were compromised and they had several months to do it and Google didn't know that it was happening and then it got detected after all the data had been exfiltrated. Okay, that's the problem that AIs make all of those attacks cheap, but they don't make attacks that are threshold structures where you differentially protect things that are hard for them to get to. They don't make those cheap. Witness pools. If I have a witness pool that say a three or four in order for an event to get witnessed, I have to compromise three witness keys.

That's a threshold structure. What's the final one? The watcher network, right? In order to not be detected, but duplicity, they have to compromise every possible watcher that a verifier would talk to. And they don't know which watchers verifiers are using. It's a really hard compromise. But let's look, let's say that I have a two of three signing key, a two of three rotation key.

I'm using a two layer delegation and a three of four witness pool. How many simultaneous side channel attacks do they have to execute to be able to compromise me? Huh? Like 10? Do you know how often anybody's ever been able to simultaneously compromise 10 keys? Ever? In the history of the world? I don't know of any. It's really hard to do. Compromising private keys is really hard to do, but it's not impossible. And we just made it practically infeasible by nesting threshold structures, which means you actually can have security without continuous surveillance if you use KERI. Perpetually secure. KERI allows… We've talked about perpetual identity, perpetual verifiability. You get perpetual security because you can control your identifier even against cyber AI. Give an example from SEDI. This is a draft proposed example for what the SEDI identity structure looks like for the state of Utah. What we have here is the state of Utah has a root ID. It issues an organizational unit delegation credential that delegates a unit AID that issues an issuing agent delegation credential which delegates an agent AID. This is the issuing agent that issues SEDI credentials. It's in some ways similar to GLEIF's. This would be the this would be the QVI. The agent would be the person the so this so organization's unit would be like driver's license division or the health and human services, whoever it is, right? Issuing agent could be an employee in the driver's license division. It could be an identity proofer like id.me or whoever they want to proxy that role out to. And they have an agent AID. They're going to issue a core identity credential to a citizen AID. Now the citizen person creates that AID.

The state doesn't. If they don't want any of these, they just don't go to the state and say, give me these credentials. They just use paper. Okay. One of the things that people don't understand is that if you don't allow people to opt out of digital identity, then there will be whole swaths of the electorate that will go out of their way to make sure your bill doesn't get passed. One of the things that make it so that SEDI was passed unanimously to everybody that said, no, no, no. You create this identifier. We don't create it for you. If you don't create it, there's you have there is no digital ID for you. You have to use paper.

If you decide not to create it, there is no identifier. There is no way for the state to issue you digital stuff. You just go paper, right? And they go, oh, okay. I guess that I guess we can't complain now, right? Like, really?

Is that's it? That's it? There isn't some subterfuge that you're doing that you didn't? No, no. It's state-endorsed digital identity, citizen sourced, right? Citizen creates the identity. The state just says, I'm going to apply reputation to it, right? This core identity is issued to the citizen AID. And then we have three other credentials. We have a proof of residence, we have a proof of age, and we have a high resolution image because in some cases your biometric, the biometric using the core identity is a facial image and that's because state law requires it. And in some cases you need a high-res image. That's the basic idea.

To attack this system, they got to attack a two-layer delegation. The citizen can decide how secure they want their own AID to be. Do they want single-sig everywhere? For most people, that's probably secure enough because they at least have one threshold structure. They have signing keys and rotation keys, right? They all have to have witnesses. They've got a second threshold structure.

Now they have to attack really hard to make it so you can't detect that your keys have been compromised when you're using your credential. That's a simple structure that gives the citizen AID and then they can just add other entitlements.

We call these entitlements. These are the core ones and then you can throw in like your like if you're on food stamps whatever, you want a driver's license, you want a hunting license, you want a food handler's permit, all of those things just become more things in this list of things that are issued, right? And we're using edges. These green binds, those are edges in the ACDCs. And edges are a normative feature of ACDCs. We spent years figuring out how to use edges correctly. The edges make it so that the relationships are just as secure as everything else. An attacker can't forge relationships. That's another threshold structure. If I have multiple edges that are cryptographically bound together and each edge is anchored in a different KEL in order for an attacker to recreate an edge structure they have to compromise multiple KELs. And if the state of these ACDCs is in a registry that's anchored in KELs, then for them to recreate this structure, they have to compromise multiple registries and multiple KELs. Those are all threshold structures. If I went back, I should have added registries and edges as threshold structures, right? How many threshold structures do you have to break before you can break this identity. It's now like 20 keys. I don't know. It's a big number. It's an unassailable big number because the likelihood of compromising one private key is really, really low. And now I've got to do it 20 times simultaneously, not 20 times in sequence, but simultaneously. It's a different set of it. As you're multiplying the probability. If I got 0.001 times 0.001 to the 20th, how small a number is that?

Yes. Yes. I'm probably missing something, but you said each edge has a different KEL. I I know that the citizen AID has a KEL, agent AID has a KEL, and all of those edges are products of those two things, right? But this ACDC, its edge, this ACDC's identifier is a hash of the ACDC, which includes the edge, right?

When you hash that identifier in order for you to fraudulently issue that credential that has this edge, you have a registry that's anchored in the KEL that you have to then you have to fraudulently change the registration that is bound to the set of this ACDC in order for you to forge the ACDC. You have to compromise the registry, not just the ACDC. Okay. Got it. Does that make sense? Now, if they weren't registered ACDCs, then that wouldn't be true. But all of the ones in this state one are all in registries and the registries are all anchored in KELs.

And all the edges are bound to the SAIDs of the ACDC and that SAID is anchored in a registry which is bound to a KEL. It's doubly bound. It's bound in the issuers, no, it's not doubly bounded. It's issued. It's turtles all the way down. It's turtles all the way down. In this case it truly is turtles all the way down.

Now let's go to guardianship. This is the guardianship thing. Now I know this looks really complicated, but under the hood, it's no more complicated than guardianship really is. Okay. We have the same delegation chain, but now this agent AID is issuing a guardianship delegation, right? This is a guardianship delegation, which is a citizen that designates the citizen ward AID, but it's delegating to a citizen guardian AID the rights to be a guardian.

And now this AID, which is not part of the state hierarchy, but it's part of the hierarchy to get this social media access authorization. The social media access authorization is protected in this delegation chain that the ward can now present to gain access. You have the same structure and then you have for the citizen ward, this is the citizen ward's AID and his core identity and stuff like that. It's all nested down here. But now you've introduced the guardian. You have additional edges in here that bind it all together. All of the cryptographic verification across the edges and the edges are anchored in registries because you're putting these things in registries to register the state. All add security that an attacker has to compromise multiple things at the same time to be able to forge the forge this these set of relationships. And if the relationships aren't normative, then you have no guarantee that the security properties apply. You're just dependent on somebody's bespoke software to make it happen. But in ACDC land, those edges are normative parts of the ACDC that operate in secure ways. They're essentially, this is essentially is a Merkle tree. People don't understand. This is describing a type of Merkle tree. We call it a it's not a tree because you can have edges can be many, but it's an M-ary Merkle tree if that makes any sense. Where the branching can be as many branches at each layer of the tree, but you basically have a hash tree. You have to so if you forge any hash, it would break the tree because the root hash would be different, right?

You got this root hash up here for this credential well actually it's the root would be is this one. The bottom credential is the root because it goes backwards. The edges point backwards. To forge this one, you have to forge those and to forge this one, you have to forge all the ones there for it to be valid.

It's a very secure, very hard to attack data structure. – Yes. – Sam, this might be a detail, but that core identity credential, that's the first one that you got on the other slide? That one right there. Is that the birth certificate or? – No. – Okay.

Where does the birth certificate come in? It would be another credential. – Okay. But it has to come before the […] identity. – The SEDI legislation only authorized these. – Okay. – That was going to require additional either legislation or what they call it rules from the agency. – Okay. The rendering that was on the slide before is for an existing person who already exists, not somebody who was just born. – There will be more legislation. – Yes, there will be more legislation and rulemaking that happens. – There will be lots of rules. – There will be lots of rules. But the legislation in 2026 only authorized two things: guardianship and the other one. And you don't need the other stuff to do guardianship. Guardianship was the driving factor. They passed a law in 2023 for social media access with no idea how to implement it. And then there's some people in the room in 2023 in like January, I think Timothy and I went and talked to people at the state of Utah and said, well, you can do it with KERI, right? And it's like, oh, okay, how do we do that? Right? It took a while to get there. – Sam, a related topic. – Oh, microphone. A related topic just of interest to people is this whole idea of guardianship is that this is a great technical structure, but the real problem is that there is actually no database anywhere that says who is whose guardian. That's right. It's all in court documents. Yes. If you're lucky, they might be PDFs.

Yes. And this would protect children better than the court documents. Because this one doesn't show two guardians, but most children have two guardians. The guardianship relationship, the state could revoke one of the guardians based on a court proceeding that then doesn't allow the revoked guardian any opportunity to pretend to be a current guardian. If they show up at the school and say, hey, I want to check the kid out of school, they can check that they're no longer a guardian instantaneously and say, no, no, you're not guardian anymore. You can't check the kid out of school. It's, it really helps protection. You had a question. Back when I was first learning about all this, the legislation does not define the technology that will be used. It's the guardrails, right? I'm wondering now, this was your proposal to them. Has Utah in one department or whatever said yes, we're going to use KERI or is it still fighting against other technologies? Okay. In multiple meetings, they have effectively said that, but they haven't officially said that publicly for political reasons. But the, but when you come to the SEDI summit, if you come to the SEDI summit in two weeks, then they, then they are going to say that. That's the plan. Okay. I have to be careful because it's very political. Okay. I've got 20 minutes. I think I'm going to skip through some slides here because I think I want to get to… You need to AI delegation? – Huh? – Are you going to show that same slide that you just had in the form of AI delegation? – Yes.

All right. Okay. I'll show this slide because that's a good thing. Okay. Agentic control is no different than everything that we've done. Everything that KERI has been doing for the last six years with delegation maps perfectly to agentic control. There's no change. The only difference would be that you might have an attribute in the delegating instrument, the ACDC that says the AID of that I'm delegating to is an agent instead of a person. That's the only thing that changes. The security, the properties like from my talk this morning about Author, authenticated, authorized, and authoritative all applies without any changes. We've already solved the agentic control problem from the standpoint of the secure infrastructure that you need to do it. You just take what's already in KERI and ACDC and just have the AID belong to an agent instead of a person. There's fundamentally no difference in how it works. It's a solved problem already. It's just a matter of talking about it in different ways.

An agent is a verifiable authorized delegate of a natural person or legal entity. Agent could be a natural person or legal entity or agent could be a piece of software or it could be an IoT device or it can be an AI harness. It doesn't matter from the standpoint of the tooling. It's the same. An agent has its own AID. This is the original meaning of the word agent. When they call AI agents, you're not doing anything other than saying, well, agent is something that you delegate to, right? It's just that right now the delegation is broken. There's no accountability of AI agents because it's a prompt. And you say, well, do this for me.

And it's like it goes and does it and then like nobody checked that it was doing it for you. They just say, well, some agent showed up and did this. Right? People are now trying to put in delegation to solve this problem. We've been we've already built the infrastructure for that over the last six years. That's really all amounts to. The details are how do you do the specifics of the authorization?

That's still something you have to work out. Like are you using MCP? What are the APIs look like? What are the details? Are you bridging from KERI ACDC land to OIDC, OAuth land, right? Those are all valid bridging points. Those are valid things that require work. But the core property that gives you the security that makes your system impervious to AI attack is not that. It's what we've already talked about. That you have this identity, this AID that is protected via key management that is fault tolerant with multiple threshold structures that an AI can't attack. You can always securely attribute who the accountable party is.

The agent isn't the accountable party. The agent has no legal standing. The person or legal entity that authorized that agent is the accountable party and we can make that securely attributable and bound in a way that an attacker can't forge.

That's the problem. Everything else is just in the just details. It's just it's execution details. It isn't solving any hard problems. The hard problem is a solved problem. Edges are how you authorize access to resources in an ACDC, right? You can use those edges to bind resources to access with. And that edge is part of a cryptographic verifiable data structure which is a graph of ACDCs. It is a graph of ACDCs is a M-ary Merkle tree in construction. That's what it looks like.

If I change any node in the tree, it changes the root. It's detectable. Any attempt to attack it is detectable. It's not just an integrity graph. It's an integrity authenticity graph because every entry in that graph is not just hashed, but it's signed and anchored in a KEL. It's bound to key state. It is secure in a way that no other system secures it because people don't hash and sign and anchor except in the blockchain world. But nobody uses blockchain anymore because of the ledger wars. We have all the good properties of blockchain and none of the bad properties. That's what KERI ACDC is, well there's a lot of good stuff on blockchain, but there's a lot of bad stuff that we just can't get around.

Let's keep the good stuff and throw away the bad stuff. We kept the good stuff. Now we get all of those security properties without any of the governance problems. Every issuer is potentially both a resource authority and an edge.

Mutually authenticated edge accessor and resource authority prevent impersonation attacks. This is what it looks like, right? Right? And ACDC is this graph fragment. It has attributes and edges. And then I chain them together. And then I get a graph. And this is the root or the origin of the graph. And I verify this and then I walk back. Right? And these are the leaves of this graph. And I can't change any of the nodes without changing this one because this is literally a hash graph.

It's a Merkle tree. That's how a Merkle tree works. It's just that it's a very flexible Merkle tree that isn't a tree. This is shown as a tree, but each of these each ACDC could have any number of edges, right? It's an M-ary tree.

Edge verifiable agent control is we already do that with ACDCs. Anybody that's delegated is an agent. All I have to do is put the word AI in little quotes there and then I've solved the agentic AI problem from the authorization authentication security point of view, not the details of what specific things that it does. You still have to manage that. I'm not claiming that I've solved all of the problems. There's lots of really good meaty problems there. But the problems that most people aren't solving, that they're hand-waving, that they're just saying, oh, we assume key management is secure. Right? When as soon as somebody does that, when they go, key management is secure, I know, okay. Okay. You just avoided the hard problem. You just assumed the hardest problem to solve. Somebody else is going to solve. And now you're going to work on the cool sexy problem that you want to solve, but you basically left yourself open to all forms of attack.

You're insecure because you didn't solve the key management problem. That's my contention. Anytime anybody does that, I just go, okay, I'm not interested in what you're doing because you didn't solve the hard problem. Once you solve the hard problem, then I'm interested in the other stuff that you're doing. Okay, let's see how it works. Right? Here's an ACDC. These are the top level fields. You have an issuance registry. Okay. Here's the fields in the issuance registry. You have a usage registry. These are the fields in the usage registry. And then you have a resource map. Routes and capabilities. This would

be like a object capabilities type authorization structure. This is a well-known thing. Not reinventing the wheel here. What we're reinventing is that how this gets secured is through these key management based security threshold structures.

Okay, I'm going to skip these slides because those are just getting into detail. I'm going to skip ahead to these slides. I got just a few minutes. This is going to be somewhat controversial. The reason it's somewhat controversial is because it's contrarian to the way the world thinks. Now, this isn't contrary.

The internet is a dangerous place, right? Everybody agrees? Dangerous place to be. Right? What we need is a digital refuge, an online shelter, an internet sanctuary, a surveillance asylum, right? Cool word, right? We want to go to the asylum to be protected from surveillance, right? Safety requires accountability.

Does anybody disagree with me that safety requires accountability? Can you think of a way that you can be safe from people who are going to harm you if the people harming you can't be held accountable for harming you? If they can get away with harming you, what's to stop them from harming you?

You could say, well, you personally could hold them accountable, but it's still the same thing. Self-defense is a form of accountability, right? If they're not held accountable, harm will happen. Fundamentally safety comes from accountability. It doesn't come from anything else, right? You could say, well, you could hide. They can't find you. Okay.

That's one way to protect yourself is to hide. But if you're hiding, what else aren't you doing? Everything else that matters in your life, right? Anybody in here ever lived off grid for any significant amount of time? Okay. What was it like? […] You're completely off-grid? You're one of the ones that doesn't want to be on the camera? Okay. What things did you have to give up to live off-grid?

[…] We're talking about food and clothing, right? You can't use a bank online, right? […] But not be off-grid. Okay. – Off-grid more than just power. – I don't mean off-grid power. I mean truly off-grid, not using the internet for any purpose. Because I'm talking about the internet as a dangerous place, not whether your neighborhood's a dangerous place.

I know people that are off-grid on the internet and they live in the mountains and they can't go anywhere or do anything because as soon as they leave their enclave, they become observable. They become surveillable. – Like Stephen Hawking in WarGames? – that kind of level.

Okay. “Safety requires accountability”. That means “privacy requires accountability”. This is where it gets controversial. People don't think that privacy requires accountability. Think about it. “Second parties to repeated interactions can correlate the first parties.” That's true.

If I never interact with somebody, then there's nobody to hold accountable, right? As soon as I interact with somebody, there's somebody now that can harm me. And if they can correlate me by repeated interactions, even if I'm as anonymous as I can make myself, if there's a way to correlate me, then eventually they knew who I am, which means that eventually they can harm me. You heard the phrase: “I know where you live”. What does that mean? It means, well, I know where you live, which means if you wrong me, I know where to go to harm you, to get revenge, to get restitution for the harm you did me because I know where you live. I know where you're vulnerable, right? If you're on the internet and you're interacting on the internet, you've made yourself vulnerable to harm. The act of interaction makes you vulnerable to harm. That means if you truly want to be private, you need to be able to hold anybody you interact with accountable for what they do because one of the ways they can harm you is through a third party. They can just share whatever data you have shared with them with a third party and the third

party is the one that harms you. But the source of the harm came from the party that you interacted with who didn't respect your privacy. Using the wrong definition of the word privacy. “Any interaction system that does not enable first parties to hold second parties accountable is fundamentally incompatible with privacy”. That's my argument. If you can't hold people accountable, you can't have privacy, which is counterintuitive to the idea that privacy comes from hiding.

If you're interacting, you can't hide. There's no way to hide if you're interacting. Everything you try to do always leaks information. And when you have surveillance that has sufficient coverage, sufficient scope, every piece of information you leak eventually will be able to correlate to create an identity of you. What is continuous identity? It's just using correlation. They're already doing it. The reason they can even propose it is they've been doing it for years and calling it something else and then said, Oh, well, heck, we can re-identify people. That means we can identify people. We'll use it for authentication. We won't use it for re-identification. We'll just use it for first…, you go to their websites and say, oh, use continuous identity. What it says is, well, you can improve the user experience because as soon as the user touches your website, we know who they are so that you can guide the user knowing that you know who they are. And the user didn't have to click a button that said cookies or not cookies or login or anything like that. You don't even have to log in. They know who you are just from what browser you used. It's lower friction because it's like, well, you don't have to give me a password. I know who you are because I know that you're logging in from this cell phone and this cell phone happens to be in this location and it correlates positively to being you.

Somebody didn't steal your cell phone and log in because if they stole it, they wouldn't be in your bedroom at 4am, right? They'd be someplace else at 4am. We know that when you're logging in from your bedroom at 4am, it's highly likely you, so we can trust it and we'll remove that friction of you supplying us an authenticator. That's what continuous identity is saying. If you think it through far enough, you realize there is no such thing as privacy without accountability because privacy really isn't privacy if you're interacting. It's confidentiality. You have to get people to respect that the data you share with them stays between you. It's confidential to you too. Nobody else has any right to it. And that's a type of privacy, but it's intimacy between parties, not truly private. How do you enforce accountability? There's a protocol called the trust spanning protocol, which is you can use the KERI suite for, but you create and control safe relationships where interaction is protected from surveillance by third parties and second parties are held accountable for any information shared with or by them. This part isn't part of the TSP protocol, but if you're using secure authorization, then you can then layer on accountability. And one of the things in this Utah state law, and I know we're almost out of time, is it called a duty of data loyalty. Duty of data loyalty is the latest and greatest legal principle for reversing the harm on the internet. And what that says is that succinctly anybody that interacts with you or uses data about you or from you that comes from an interaction has to be loyal to your best interests. They can't use that data in a way that isn't in your best interest. That means they can't do something like trick you into saying things like, oh, we're going to de-identify this data and then it's not data about you. That's a lie. But what can you do, right? If they've agreed to use it in your best interest, then they no longer have a safe harbor. If they take that data and sell it to a third party, whether it's de-identified or not, and they come back to harm you, you have recourse against them because they were disloyal. And the burden of proof of disloyalty is different from the burden of proof for harm. Disloyalty, all you have to prove is that they self-dealt. If they benefited from it, whether you are harmed or not, they're presumptively disloyal and you can sue them. Your ability to enforce increases exponentially. That's in the Utah state law. And we snuck that in, honestly, because the data aggregators don't like it. And I'm done. All right.

TALK VII

2026-11-02-samuel-m-smith-ryan-hansen

KERI Mini Conference November 2026 · talks · 58:28

SUMMARY

AI generated summary.

Ryan Hansen covers the five Ws of KERI infrastructure: wallet, witnesses, watchers, web, and wizard. The wallet holds keys and the key event log; witnesses receipt events offline; watchers make duplicity provable. Web is discovery and transport; wizard is usable defaults. With Samuel M. Smith he covers Locksmith, Fortweb, kf-boot, and why relying parties need their own watchers — a controller-only witness pool can eclipse readers.

Please go ahead and get going on this one. All right. For those most of you who probably don't know who I am, my name is Ryan Hansen. I'm a member of the KERI Foundation Development Team, the small team that Sam put together to kind of help bring the five W's to life. I'm going to give you a quick rundown of what the five Ws are. Sam already introduced them, but we're going to go through them a little bit more. And then I'll give you a little bit of a state of where we are in the development of these. This alternate title is the state of the key state. This is my state of the key state address to this community. That was weird. Okay. I've got to make sure my slides line up with these slides.

First of all, I'll give you a quick why when it comes to the five W's. KERI, as those of you who are very familiar with it and as Sam has already established, KERI is all about key state. Implied in that is identity and keys themselves. But what good really is an identity and keys that you if you can't prove the state of those keys that they're still within your control and are still verifiable. Proof itself requires witnesses. It's all well and good for me to assert an identity or assert that I've done something, but it's not as strong or it's weaker without witnesses that will attest to that particular assertion of some kind.

Witnesses also need to be independent, meaning something outside of me or the control of that particular ID, whatever it might be. And we'll get into that a little bit more. Highly available, meaning that state can be determined whether like I particularly am online or not. And to wrap that up, the Ws are a big part of the I in KERI. There's more to it than just what I'll be talking about today, but infrastructure in KERI is what the Ws kind of are about. With that, the five W's as Sam already has kind of gone through. We have the wallet and I won't spend a lot of time on each one of these in this slide but we'll move through these pretty quickly. Wallet, Witnesses, Watchers, Web, and Wizard. Each one of these are critical part of the KERI overall ecosystem as Sam already illustrated before. Let's get into each one of those little bit more. Starting with the wallet. Wallet has some primary responsibility and function which is first of all it's the home of the identity and the keys. That's where all these things live. It's that secure enclave as Sam said to house the identity itself and the keys. It is also responsible for the KE in KERI which is the key events, all the events that take place relative to that particular identity and the keys that it controls. Those events are inception, rotation, and interaction. There's a few more when we get into the details of the specifics of the KERI protocol, but effectively these are what your wallet will do. The wallet does in the KERI ecosystem is to generate those events and publish those or keep the record of those in the key event log in its own key event log.

Next with witnesses, their primary responsibility is a witness of the truth of the key state or to assert that the key state is what it is. Is that a fair statement, Sam? – Yeah. It's the R in KERI because it's the receipts. – Funny you should say that. Exactly right. When the key event is issued, let's say it's created

an event and is published, when the witness then basically gives back the receipt or creates the receipt on that event with its own keys, but then establishes more firmly that that key event log is intact, that the math lines up, everything is in order. Obviously that's the R in what KERI stands for.

Again, high availability witnesses are designed to be able to be highly available to run in a situation where that those key events in the key event logs that they're witnessing for are available at any time whether the wallet itself is online or not or available or not. That it that any verifier that needs to verify that key event log can do so, can get it from a witness.

And choose your own meaning independent. What I mean by this is that when you establish, and I'm using the word you, any controller really is what I'm talking about, but when in choosing what witnesses are going to witness for any given ID that you might create with your wallet. The idea is that there would be many different witness providers or witness services available that you could choose from. And in an ideal scenario, you would have more than one and all of them would be independent of each other. There's no collusion between that, no possibility or less possibility of collusion between them about anything that might be… they might want to assert. No way to kind of fudge it, I guess. It makes it less likely that you could do anything untoward with the assertions of the key state itself.

Okay, watchers. And I'm kind of going through these fairly quickly because mostly I want to get to what the state of development on these particular tools. Watchers are independent witness observers. They're always watching as our little friend the monkey from Toy Story 3 if you guys recognize that one. It was always watching. It was always there. Their entire role is really duplicity detection and just so that we're all on the same page about what that means, is: duplicity would be when any two different records of the same key event log disagree about what the state of that identity is or the keys of that particular identity. If there's any kind of disagreement between them, that would be duplicity. Watchers' whole purpose is to watch the witnesses to make sure that they all align and agree about that key state of that particular ID. And if there is any kind of duplicity, they detect that and then you can take action necessary to rotate and do what you need to do. The idea becomes trust me turns into see for yourself rather than just trust me on telling you this my identity. This is with all these mechanisms you can actually verify for yourself effectively is what that's suggesting. – Can I add? – Sure.

– Duplicity is independently verifiable. It isn't something you ever have to trust somebody on. Anybody that detects duplicity can prove it to anybody else. If I detect duplicity, I can prove that it's duplicity. You don't have to trust me. – Thanks for that.

– Really the key thing for watcher is somebody bothered to watch with. – It doesn't matter who's watching, any watcher is as good as any other watcher because you don't trust the watcher. It just proves to you its own duplicity. The worst that a watcher can do is not watch. – To that point, if you want to weaken the security and anti-fraud properties of KERI, then as they say in the movie, get rid of that monkey.

We've got to get rid of that monkey. All right. Web. There's a number of things that are implied by web or meant by web. Discovery, as we all know, the web is a series of tubes. That's a common old joke for those who are old enough to remember that one. But there's a number of things that web implies in the KERI world. The ability to discover, discovery meaning other… I don't know, how would you describe it, Sam? Discovery of other witnesses… – Discovery of infrastructure. We don't need a secure discovery mechanism. We just need one that works well and the web already is there. Why not just use the web for discovery instead of inventing a new discovery mechanism. – The expert, ladies and gentlemen. Communication. Once discovery has happened, obviously it's about communication between which

also implies transport, which includes messages, any kind of messages, events, and things like that between these different parts of the infrastructure wallets and witnesses and watchers and all of this. All this done over web protocols. No other special infrastructure is needed for that. It's all web protocols and agnostic of the transport itself, whether it's HTTP, Bluetooth, NFC, whatever the transport mechanism might be that KERI can travel across all of those mechanisms without any particular issues or any special needs. – It doesn't trust anything that the web does. It doesn't need to trust, but it can certainly use it to send stuff. – Now, it's interesting because we talk about trust a lot when it comes to identity and all these kinds of things. But ultimately, as many of what we're going for is not having to trust anything and not trusting anything and making it verifiable and provable. And that's what this whole ecosystem and KERI itself is designed to provide. And then finally, wizard. And wizard becomes a little more, I guess it's a little harder to describe exactly. The idea would be sort of one-click KERI. That maybe is a little bit of ambitious. I don't know that one-click might ever be the thing, but that's the idea that we're trying to suggest with wizard. Making KERI as usable as possible for as many people as possible by determining some things like safe and useful defaults when it comes to creating IDs and setting up witnesses and things like that.

Abstraction APIs for developers. KERI itself, the protocol can be pretty dense and pretty difficult to get up to speed on quickly. But by providing potentially some abstraction type APIs and libraries that wrap some of the lower level functionality of KERI machinery in a little more accessible calls and things like that, then we get a little bit closer to making adoption of KERI an easier thing, an easier path. And ultimately what wizard concerns itself with is trying KERI more into product than protocol. Building product around the protocol to make it something that average individuals can at least get their head around enough to recognize it's something that they want to use and then make it easy for them to do so. That's the intro to what the five W's are. Now as far as the state of those five W's itself. The KERI Foundation has a repository on GitHub under https://github.com/keri-foundation and within that organization on GitHub we have several different repos that are of interest there. First of all is Locksmith. Locksmith is our reference implementation of a desktop application wallet. This is kind of where the core work of the wallet goes in and then that funnels down from Locksmith to these other repos. We have one called Fortweb which is… – Yeah and I just want to note that Locksmith was donated to the KERI Foundation from healthKERI which is a company building KERI based stuff in the healthcare sector. – good call-out. HealthKERI has given us a lot of good boosts. We'll talk a little bit more about that in a minute. But Fortweb is the same wallet but in a web browser type of environment, WASM-based browser environment. And from Fortweb, we're able to bundle and create Fortios or FortiOS, whatever you want to call it. And FortiOS, so the iOS and the Android mobile apps of these two things as well. And then in more of the infrastructure side of things or the deployment infrastructure, we have kf-boot, which is… You see my beautiful, really detailed architectural diagram here. We have kf-boot, which is kind of an onboarding service that communicates with the wallet to onboard an ID and set up witnesses and that kind of stuff.

We have witness-hk and we have watcher-hk, both of which the HK in those once again is healthKERI. They were the originators of those libraries and donated them very generously into the community and open sourced them for us to use and to start to build on even more. And then we have also kf-mailbox which has a little asterisk because this is something new. It's not quite ready for use just yet, but it's something we're introducing now as a mailbox service that can be used within the deployment infrastructure.

The state of the key state without KERI is… Who else? We don't actually know what it is. That's my way of asserting that and joking that KERI is the way to go when it comes to being able to prove our key state. And with it, that's where we are. Silly pictures.

That's pretty much it for my particular slide. Sam, unless there's any questions, I got through that quicker than I anticipated… – We can certainly go into in more detail. Let's ask, but I think this is a good place to ask questions because I know I cover things very quickly and then there's lots of latent questions and saw him use the whiteboard to answer those. – I've got a question about kf-boot.

For kf-boot, you said that it communicates with the wallet when you've set your identifier up. I looked at kf-boot. My understanding is that there were five standard and you could dive into this. There were sort of five main pieces of infrastructure helps you stand up and then there was a little bit more definition. If you could, if you could dive into that, I had never heard what you just really said, which is when you set up an identifier, it walks you through some things. Can you detail that for me and then we can sort of come to some questions to the other parts? – Evan could probably address that a little bit easier. Evan does a lot of work with the wallet itself and that communication process. – Yeah. You came from when we got this infrastructure from healthKERI, they had their like healthcare platform that they use to orchestrate this sort of stuff. Like they stripped it out, sent it to us to be open sourced and we had to kind of fill in the gaps of the like identity infrastructure orchestration. That's sort of what we have kf-boot for now as like an open source identity infrastructure orchestration like Ryan's. – Kind of like a plugin infrastructure, right? Anybody that creates witnesses, anybody that creates a watcher service could register with kf-boot, right? Do we have a registration yet? We don't, right? But we have to have a way to discover. Is there a way to discover their services? Or we have an implemented data?

– Like when we set everything up on DigitalOcean. – They'd have to add up like a connector. But if you have that connector, then any services that kf-boot knows about, then that wallet could say, Oh, I want witnesses from this service and I want a watcher from this service. – Part of why I ask is since we at GLEIF work with the ecosystem with lots of players and there's going to be potentially many different providers of infrastructure. I wonder is this generalizable or generalized infrastructure discovery tool as kf-boot and it interacts with sort of a set of infrastructure providers and a set of wallets that recognize that discovery protocol? Am I characterizing it correctly? Am I going in the right direction or am I misunderstanding?

It's early enough in its development that it could go a number of ways. All it does for us right now at its current state is it's the first landing point that the wallet uses to discover what witnesses are available to add those to your configuration for your particular AID. That's the first step and that's really what it's doing at this point is kind of facilitating that initial onboarding to be able to set up your witnesses. Where it goes from there, I I guess there's lots of different ways you could take it. It's fairly simple little service right now and it's all work in progress still. There's still much to be done on that. – I guess I have one. I guess for GLEIF, the important thing is to be able to offer to holders of vLEIs, for example, and to allow companies who want to implement the vLEI credentials to have alternatives to custodial wallets offered by QVI. We're looking to be able to build up in the ecosystem alternatives for standalone wallets. With the capability that this has at this particular point, are we talking about functionality that would allow a holder of a legal entity level credential, an official organizational role credential, ECR credential to accept that credential and then be able to present it in whatever.

– That's not kf-boot I think. – No, no, that doesn't sound like it to me, but we probably need to go into that a little bit more because that would need to understand implications of what you're… – I think what she's asking is it that's really the wallet the open source wallets would be able to do that. Kf-boot is

bootstrapping the infrastructure the witnesses and watchers so if you set up your wallet and you want to have four witnesses where do the witnesses come from if you're not standing up your own witnesses you're just some person but let's say company A has a witness service and company B has a witness service and company C has a witness and a watcher service, then you would want to be able to have a way to click on a list of service providers and say, configure me a witness in this data center that has these properties. And then you could create your witness pool from providers. But right now, it's just really basic. It's just allowing the KERI Foundation wallets to configure witnesses and watchers. The KERI Foundation stands up. Certainly, the idea is that eventually we want to have that sort of plugin architecture. It's just it's a resource thing. – As you see here, this KERI Foundation, that's actually a plugin within this wallet infrastructure. The idea would be that many providers could provide plugins that could plug right into this and you could just choose from them and see what services they have available and set up based on that. – We do have the plugin stuff, but not the other provider. – Thanks for the clarification. – I'm always interested in the wizard part of it, the user interface. I'm curious, is there… what's your vision for… Say there is a duplicity event by launchers or observers.

How does that surface to the user? To the verifier, for example? – It's pretty open to however an organization wants to implement that kind of capability. We don't have anything built into it, just yet to handle that as it is, but that's something that's pretty open to whatever tooling you want to put in place to be able to watch for that event that's emitted by the watcher to say that there's a duplicity thing and then whatever path you want to send it down at that point, whether it's a pager duty message or whatever it might be, or even an automated process that connects in and somehow rotates the keys automatically. I don't know. It's kind of a decision-making process within the implementing organization. – The mailbox can help with that because you can post duplicity notifications into the mailbox and then your user experience would go, this particular AID that we're interacting with, their key state has evidence to duplicity. And duplicity will be no duplicity and at some event there's duplicity. And is it reconciled or is it unreconciled? At that event, you shouldn't trust anything that you do with that AID. You can trust everything that happens before. If somebody presents you a credential issued before the duplicity, you can trust the entries in their key event log. But at the moment of duplicity, anything new you wouldn't trust. You wouldn't trust something they signed because you don't know if the signature is with the correct key state because the current key state is uncertain. You don't do any new things with them. Your user experience would have to say, okay, you can't do certain things with this entity because we can't trust their key state until it gets reconciled. And in some cases they can reconcile it and then you can start trusting, but then you'd have to see that it was reconciled. You'd have to look at their key event log and go, it was reconciled. They did a rotation. They did a recovery rotation. It cleaned up their key state. Now I can trust it.

It is an important user experience that we haven't… we're just at the point where people building wallets need to address that. It's a really good problem to solve. – And just to reiterate this, we refer to all this as reference implementation because we are as a foundation fairly resource constrained. We can't build all the things that we want to build right out of the box. We're working our way there as much as we can. This is all meant to give an idea of what can be done with this and to give some baseline to start from. It's all open source. It's all things that people could take and start to build from. But obviously we're not going to be able to build a fully fleshed out production level capability, at least not as we are right now in the foundation. – I want to go back to the point about the wallet being the source of key events and the log. And two questions come to my mind. Is it the kind of thing where if I run different devices, a laptop, a phone. I could have a single wallet that shared persistence of where the wallet data was in the cloud and my app was running. And in order for someone to inspect my key of that log, you have to have

access to my devices. In the past, I never really thought about where the log was. But now that the concept of the log being somewhere and can be looked at, can be verified by a copy of the events. – well your witnesses hold a copy of your log. – Witnesses have a copy. That's recovery aspect of things. – No, no. That's for anybody to verify. They don't have to look in your wallet to know what your log looks like. They just ask your witnesses. – Do I only have one device where my wallet is or can I actually access a wallet from my laptop or my phone? – There's different wallet architectures. I know the rooting guys are here and there's an open source wallet that GLEIF uses called the signify-ts architecture, which is a hosted wallet. You have what's on your device, which is your private keys. Everything else is in the cloud. – Right. That's kind of what I'm thinking is I've been in the world of Apple where the photos and things get copied between device you can access your cloud account securely from multiple devices. I would certainly like to be able to do that with my identity in a KERI sense from a log that had access to the persistent state of my log somewhere under my control. – As we're implementing right now that's not possible just because what it requires is you to actually copy the keys to that other device. You're going to create any kind of an event from some other device. You've got to have the keys to be able to do that. If you want to spread out your footprint a little bit, then that would be possible. But we haven't implemented anything like that yet. – What if my keys are stored securely?

– Secure key storage is on a spectrum, right? There is no such thing as perfectly secure key storage. The question is, what are you trading off? If I store it in a secure enclave, they have the property that you can't export the private keys. I could store them in a hardware security module like a ledger or a YubiKey or something like that. Some of those allow you to export them. Some of them don't. It's a property of what they can do. Because KERI supports multisig, you could horcrux yourself so that you have multiple keys that control your identifier, each key stored on a different device. Then you'd have to do like a two or three or something, or you could do a one event, right? You could say any one of the keys can sign. I just need one key and that just changed your security property, right? Or you could decide that you're going to allow your private keys to be exportable and then you can share them across devices but you share them in a secure out-of-band way. – For example the hard drive is on my laptop and my secure enclave in my laptop in my room.

But the extension of that where that was now on Apple cloud service is less secure just because those keys are not encrypted there. – No, you would want to do that. – I know, but is there a way? – If you do end-to-end encryption on Apple, then they're encrypted and Apple can't see them. – Okay, that would be a viable… – Yes, one of the things is you have to look at the likelihood of exploit and the susceptibility to exploit. And this is all part of survivability engineering. Survivability engineering says, how susceptible are you to exploit? If you're a high value individual, you're way more susceptible to attack than grandma is. Grandma's got $100 in a bank account.

Nobody's going to spend more than $100 in resources to get it. Right? It's not worth it. Which means that her security doesn't have to be as much as security at somebody that has 4 million. Now like I know somebody whose phone was SIMjacked. But that person had a Coinbase account with millions of dollars in it. They went to a lot. They knew that that person had a Coinbase account with millions of dollars in it. They went to great lengths to SIMjack them. I know millions of other people who have never been SIMjacked and nobody's ever attempted to SIMjack them because there's no ROI to the attack. Storing your keys in Apple's cloud end-to-end encrypted, Apple doesn't have the keys. The risk to you is, do you lose the decryption key? That's your failure point. Not being in the cloud. The cloud itself isn't the failure point. Strong encryption isn't breakable. They can't break the strong encryption. What they break is your key management. – I'm just looking for a way to make the use of the wallet manageable.

Yes. As opposed to confined to a device that might lose. That's right. Some people will complain and say you have to use hardware keys that are stored on a device. And that's not a viable position to take because it doesn't reflect the actual totality of the threat environment. For most people, storing your keys in the cloud encrypted with an end to end encryption is a viable place to store them. And now what you have to do is not lose your decryption key. You have not forget your decryption key. If you have 1Password and you're storing secrets in 1Password, then you have to say, well, is 1Password sufficiently secure, right? For your threat profile, if it is, then there's all of those things that are available to you as a wallet designer. When people are doctrinair about certain thresholds of security without looking at the total landscape, then they're not valuing the trace space correctly. And what they're either doing is they're over securing things and adding too much friction to users who don't need the security and they don't benefit from the friction. Or to solve for that, they make everything the lowest common denominator, which means for people that need more friction and need more security, can't have it. – Is there a way for KERI to actually play a part in rotating those encryption keys? – KERI is a protocol so it's not a person. There might be somebody in the hallway that's named KERI. We can ask her… – As a separate application of a KERI service to keep track of other forms of encryption. – that's not part of the KERI protocol, but certainly it's something that people could… I talked about hosted key management. KERI enables hosted key management. If people want to provide a hosted key management service, absolutely. – I could have a separate KERI account just for managing my own encryption keys. – that's right.

That's what I'm talking about. – All of those are available. Good questions. Let's keep passing the mic. Oh, he's got one. – Speaking of discoverability, I think DNS is a beautiful protocol. Do you think it makes sense to embed some of the witness endpoints into TXT records or SRV records. How are you thinking about weaving this network? – The way we use DNS right now is… Part of DNS is the “Well-Known” protocol.

If anybody that has a web service can make it so that all of their infrastructure is discoverable by publishing those infrastructure endpoints as “Well-Knowns”, right? There's a well-known protocol called a “Well-Known” protocol that you can publish and just say, witness and here's the identifier and here's the endpoint. We take advantage of that and that's crawlable. People can crawl the web, discover everybody's witnesses and everybody's watchers from the Well-Knowns. That's one thing. And then we have something called an OOBI, right? The out-of-band introduction and you can publish those and use those for discovery. We do a lot of that kind of stuff. DNS itself, yes, you could publish in X.509s anything you want, right? It's a flexible document protocol. I could see people publishing if they wanted to. I A certificate authority might want to do it that way. They may want to publish stuff in X.509s that enables discovery because the people are discovering the X.509s already. I know he perked up because he's a CA. – We want to talk a little bit about Merkle trees. We got five minutes left and lunch right here. We want to talk about Merkle trees or what's that we were doing next? Question. – Since it's about the current state of these things, what watcher and witness infrastructure networks are available that people plug into that are live today? – Okay. Sorry, I forgot to mention that we actually have a handful of servers that are running on DigitalOcean that are available.

I don't have those up yet, but we'll publish that information of where that stuff is. But there is a witness network and watchers available that people can use as kind of an entry level into testing it out. Sorry, I don't have that information up here. I don't have it in my slide deck, but we'll get that out onto the repo, make sure everybody knows what IP addresses and all that kind of stuff. – But that's a good question. HealthKERI, they provide witness and watcher services to their customers. It's not publicly available.

You have to be in their space. The GLEIF ecosystem, there are witnesses that various people run, but they're primarily for that. The main, I think one of the main outcomes of the state of Utah SEDI initiative is that it will be a forcing function for providers of KERI infrastructure to all of the citizens in the state of Utah. It doesn't exist yet, but one of the things that we're doing is we're open sourcing the core libraries so that other people can start to stand up those services. But nobody's spun them up yet because it's a chicken and egg problem right now. It's like the state of Utah isn't using SEDI yet. They're about to, right? And in order to do that, people have to start spinning up witness and watchers and wallet services for citizens. They meet weekly to foster the community to start talking about these things. It should happen over the next couple of years. – That brings a follow-on question. You said healthKERI has theirs that are working in their ecosystem. GLEIF has theirs that are working in the vLEI system, but they're not publicly available at this point.

Then my question is as this… – They're publicly available to the extent you're using a vLEI. – Okay. But what are the advantages and disadvantages for ecosystems such as healthKERI, GLEIF's setup to make those publicly available for other ecosystems to be able to use as the watcher network?

Advantages, disadvantages, why you would have your own versus putting it out into the global infrastructure. – That's a really good question because it's about how do we bootstrap infrastructure, right? Who pays for the infrastructure, right? That's a challenge problem. And Phil wrote a white paper, some presentations about this. One of the challenges in the identity infrastructure who pays for identity insurance. The state of Utah is going to pay for identity insurance so that everybody gets identity insurance for free, right? And then the question is who pays for the other infrastructure, which is witnesses and watchers, right? The state of Utah has a vested interest in making those available for disadvantaged populations in the state of Utah. They're going to fund some of that. Every business that wants to leverage the free identity assurance now has an incentive to make witnesses and watchers available to their customers for free.

It's sort of how do you bootstrap that kind of infrastructure. – With SEDI, do you see as this goes beyond the state of Utah and this new consortium that's… – That's the idea. If other states adopt it… – they'll start helping that to happen. – it's the classical cold start problem for infrastructure, right?

How do you start infrastructure that benefits when the infrastructure exists? And until it exists, everybody's like, well, if I stand it up and I spend money on it, when do I make money? I don't make money until I get to scale, right?

You have to you have this cold start problem. The state of Utah is a forcing function to enable to solve the cold start problem. That's one of the things that's really cool about SEDI is that we have somebody who has an incentive to solve the cold start problem. – And this state is also both an issuer and a recipient of stuff. They actually can create an entire ecosystem that then other go, oh, that's working. Maybe I should sit on top of this or you sit beside it and use it. – The state needs witnesses and watchers at scale to interact with millions of citizens for no other reason than just the state to conduct its own business but if you then at that point have witnesses and watchers at scale for millions of citizens just to do business with the government it then becomes easy for other people to say well wait there's already millions of these out there millions of users, I can start to compete in that space. I will offer a witness service because the state already created a market for me. That was actually the name of the proposal that I wrote to the state, which is a public information utility, a public identity utility. We're building stuff in the KERI Foundation that we're donating to anybody. We're not going to build those utilities. We're just providing the stuff for somebody else to do it.

Somebody else has to catch the vision. And I know the reading guys are trying to do that, right? Is that fair? They're not. Okay. – I will contribute though from a blank perspective. It might not be evident, but we have some internal bootstrapping going on in the way that the governance framework is written for the vLEI issuers. When anybody comes into the infrastructure they can share witnesses. They have to set up five, spin up five, but two of those can be shared. And then they have to increase the witness pool by three. The overall vLEI ecosystem infrastructure by three at that point.

That's how we grow our witness pool. And we have not even discussed at this particular point in time whether it would ever be a possibility that we would offer something to anybody outside our ecosystem where the public sector is much better placed in order to be able to do such a thing.

– I wanted to add another. SEDI is an adoption vector that is trying to that has to solve the cold start problem. Right? And the state of Utah has the best industry. The healthcare sector has a similar problem. Now, it's the healthcare sector, but those two start to move together. And the reason is that right now in this administration, they passed new regulations that say all healthcare information, people that work in healthcare, all patient data has to be portable. They're saying we want patient data to be portable. Those regulations are causing problems in the healthcare sector because they have on one hand all these HIPAA rules that say you have to protect this healthcare data and on the other hand they just said but we have to make it portable and the reason is that providers in the healthcare sector tend to want to capture their patients and one way they capture them is make it so their data is importable. If I get a bunch of tests done and I have a bunch of scans and things like that and I want to go to a new provider, well, the new provider says, well, I can't treat you because I need to do all of these tests. And you say, well, no, my other provider did them. Just get the data from them. And the new provider says, they won't give it to me because they don't trust me. They have no way to know that they can send it to me and not violate their regulations. The new regulations are saying, no, no, you have to give it to the new guy. And the old guy says, but we can't give it to the new guy because these other regulations say we can't do it. KERI and ACDC solves that cross-trust domain transfer problem. These two can trust each other so they can move your data from one provider to the other. And that's patient data, which means the patients now have a reason to have a wallet, have a reason to have infrastructure, and that then starts to merge those two ecosystems. Because if I have a wallet so I can do business with the state and I need a wallet so I can do business with my healthcare provider, it can be the same wallet because it's the same protocol under the hood. And now you start to get that flywheel effect.

– Wasn't the case already HIPAA took care of that just by the patient authorizing the transfer, right? – They don't trust the patient authorizing the transfer. It's, you can do it that way, but it takes weeks. But I'm going to say that's the way it has been done. – Yes, it's manual, right? And it takes weeks. And if you're suffering from a serious illness and you might not live weeks, that doesn't really help you, right? – The patient's essentially the middle man. – Yes.

– Back on the topic of public infrastructure. I'm curious if you're familiar with the app protocol ecosystem and their identity system did:plc? – I've heard of that. – I see it's not KERI, but it's like a verifiable key history. It has some similarities. I think it's easier to use, but doesn't have the same security properties. But one of the things I think is nice or convenient about the ecosystem is they just provide a directory service, which is like analogous maybe to a witness. But I haven't really thought through kind of the security properties of it because like de facto it's a centralized directory. – that's right. – But it's interesting relative to this question. – If I centralize things, it makes it much easier. The world becomes really easy when I centralize things. That's always been a problem for KERI is that we wanted KERI to be fully, totally zero trust, decentralized and that means we have a hard problem to

solve. I get it. We created a hard problem by forcing it to be totally decentralized. But that's why we've been searching for these forcing functions like healthcare or state of Utah. Actually, Provenant telecom is not a forcing function because it all happens behind the scenes. When it stops call spam, it's your cell phone provider that's signing and checking the signatures. You, the user, never even know that it's happening. You just see something on your screen that says, this call came from Microsoft. You can trust it. And you go, okay. – And I would say that is actually a turning point for KERI because the early use cases were all organizational, right? – Yeah.

Provenant, those kinds of things. – Yeah. – State of Utah, healthKERI, those are things that are different than what has happened before. We need to watch them because that will say, how does this actually work? – Yeah. That's why I'm optimistic because I see those as real serious adoption vectors and there's drivers that aren't going to go away. There's regulations involved. All of those things are forcing functions that will force things. And let me say something about did:plc, did:webvh, there's a couple of other ones, have adopted many of the mechanisms in KERI for key state. What they have not adopted are the mechanisms for perpetually verifiable issuances, which means you can't use them for many of the applications that people want to use them for. It's similar to my criticisms of zero knowledge proofs. You can't use them for issuances that need to survive a long time because there's no way to verify them once you rotate keys. You know you rotated keys, but everything issued under the old keys becomes unverifiable the moment you do it unless you're anchoring those issuances to your key-state. That means they have to change their protocols and as soon as they do that, they now become as complicated as KERI and why do it that way then?

– I wanted to add to that to what you said. I'm involved in did:webvh and involved in decentralized social protocols such as AT protocol. I'm familiar with did:plc, etc. To add to what Sam is saying that there first of all, there is osmosis of KERI security properties to some of the DID methods for those exact same reasons because it's same shape of the problem, same security space, very similar shaped solutions and things like that… Part of the unsolved problem in both did:webvh space, in the social media space and in KERI as we've been discussing is exactly who's going to run the infrastructure, right? In social media space, it's always been Twitter. Okay. Twitter's running the infrastructure or Facebook or whatever in the hopes of monetizing. Bluesky came along who is operating did:plc, the servers. And they're providing the infrastructure and they're venture capital funded in the hopes of finding a business model and monetizing, right? What's the alternative? The essentially one and a half to two things that we have been mentioned, governments, right? On the federal state or city level like SEDI or what we've been seeing in the decentralized social media space such as Mastodon, basically small communities and volunteers. Right. It was run the common infrastructure of the… – It has its challenges. I'm not saying that it's not challenging.

– I was going to say, did:plc? Delegation and multisig are challenging to get right in a way that is usable. And those are two things that KERI does that no other DID method does. I've looked and I could be wrong. It's been a while since I've looked at did:webvh. But when the last time I looked at did:plc… – I read the did:webvh latest spec as of like three weeks ago and it doesn't have any of those in it.

– I… – Is that right? – It's it has a new user space. It has it in application space, not in the protocol. For those listening to the recording, did:webvh is working on multisig in sort of application space, not in the signature space.

They're trying, they're working on it. –Yes. Once you decide to accept you need multisig and delegation and anchored issuances, you have KERI. Completely. You've recreated exactly the same thing. – I could talk a little bit about how witnesses solve the detectability problem and the availability problem. You

guys interested in that? A lot of KERI infrastructure, a lot of KERI protocol does multiple duty. It's like for me a sign of a good design is that I design something to solve one problem and then serendipitously it solves other problems that I hadn't yet got to yet. And they go, oh, wait a minute, this also solves that problem. Delegation, delegated AIDs solve multiple problems.

Witnesses solve two different problems. The first problem they solve and Ryan mentioned this, the first problem is availability. Originally, one of the problems that people complain about is, well, if I have a wallet and all my keys are in my wallet and my wallet it's offline, nobody can verify my key state because my wallet's offline. I can't have a presence because my wallet has to be online. And a lot of people approach it and say, well, then I need a hosted wallet so that my host has my stuff. A lot of people solve it that way. I have hosted wallets. Now they go to the host to get it. But the problem is the host now becomes something I don't control and I can't trust. Now I've basically solved the availability problem, but I've introduced an even worse security problem.

Witnesses solve the availability problem without introducing a security problem. – What's the threat model on the host? You said something you can't control. Aren't the key event logs still secured by the user's keys? – My copy is secured, right? But the host could publish a key event log to anybody they wanted to and I wouldn't know they were publishing a duplicate copy. I wouldn't be able to tell that they were cheating. Because my host can eclipse me. I'm subject to an eclipse attack if I just have a host. I need an eclipse attack proof availability mechanism. The only eclipse attack proof availability building mechanism is a Byzantine fault tolerant distributed pool.

Now, it doesn't have to be total ordering. It just has to be eclipse proof. When I say deconstructed blockchain, I said I've got to solve this eclipse attack problem. The only way to solve an eclipse attack problem is a distributed pool, but I don't need the distributed pool to do a total ordered replicable state machine, which I need for double spend proof. I don't have to do a double spend proof. I just have to know that there's duplicity. Double spend proof is actually duplicity hiding. It says there's only ever one valid copy and if somebody cheats then we'll just take the cheat and say that's the true state because the cheating is they've compromised your keys. I need to detect that my keys have been compromised. Also have it be highly available and so that's why it's solves multiple problems because it solves availability with a distributed pool. There's lots of ways to solve availability. You can have master slave or whatever they call it nowadays. Sharded. You can have a sharded architecture. You have a host architecture. There's lots of ways to solve availability, but a distributed pool of independent servers that exchange data using a fault tolerant exchange algorithm is a way to do it. The reason you pick that way is because it doesn't just solve the availability problem, it solves the eclipse problem. It prevents you from being eclipsed. Now you need to control that pool so that you don't have to trust anybody. It's under your control which means that ultimately they can't trust what you published because you could publish, you could do an eclipse attack on the people reading from your witness pool, that's why you need to have a watcher pool that they control because they can then detect that you're not eclipsing them. It's just one step after another to go, okay, I have to solve this problem. And then once I solve it, then it's done. You have availability, eclipse attack, and then detection. The third one is detect key compromise. Now here's the thing. If I'm doing a key event log, the key event log doesn't enable me to detect key compromise.

There's no key compromise detection in a key event log. The witness pool gives me key compromise detection, but only if I'm anchoring issuances. If I anchor issues, the only way I can get anchored is it gets entered in the key event log that a witness witnesses. And if I have my own copy of the key event log and I check my witness and the witness has a different copy than I have and there's stuff in the

witnesses copy that is anchored and the only way I can get there is if my keys have been compromised then I detect the key compromised. But until you anchor issuances you can't actually detect key compromised. You don't get the most important security property of KERI which is I can detect key compromised. You have to trust that there's some other mechanism to detect your keys. You can rotate your keys, but you don't know when to rotate. With a witness pool when to rotate. But the witness pool doesn't tell you when to rotate unless you're anchoring issuances. Any issuances that aren't anchored are not detectable when you get your keys compromised. You don't need to anchor everything because not everything's important enough. But the important stuff you anchor and that allows you to detect it. Without anchoring, you've missed most of the security properties of what KERI provides. But it's complicated, right? You have to complicate your system. Okay, I'm done.