TRANSCRIPTS

November 2026 Videos

A complete transcript of every recorded talk from the November 2026 Videos.

3	41:42
TALKS	RUNTIME

November 27, 2026

CONTENTS

Full transcripts of all 3 recorded sessions, in the order they appear in the archive.

01	2026-11-24-16-01-50-KFMCA-Veridian-intro	00:37
02	2026-11-24-16-01-50-KFMCA-Veridian-part-1	19:06
03	2026-11-24-16-01-50-KFMCA-Veridian-part-2	21:57

TALK I

2026-11-24-16-01-50-KFMCA-Veridian-intro

November 2026 Videos · talks · 00:37

SUMMARY

AI generated summary.

Karla McKenna introduces a reprise of Veridian's IIW demos for the KFMCA meetup. Jaime Caso and Roberto C. Morano will show two pieces of the CESR-based Veridian wallet on KERI: biometric smart cards that replace the seed phrase so a user can move the wallet to a new phone with one card, and KERI credentials used to log into websites through OpenID Connect.

Thank you Roberto and Jaime for reprising some of the presentation that you did at IIW about the Veridian wallet. In particular, there are two demos that will be presented today. Smart cards replace the seed phrase and the user can move the wallet to a new phone with one card. And then the second demo is KERI credentials used to log into websites with OpenID Connect.

I'll turn it over.

TALK II

2026-11-24-16-01-50-KFMCA-Veridian-part-1

November 2026 Videos · talks · 19:06

SUMMARY

AI generated summary.

Jaime Caso, Innovation Engineer at Veridian, demos wallet recovery that replaces the seed phrase with biometric smart cards. In the Veridian wallet every key still derives from a bran, shown to the user as a seed phrase, but that phrase is easy to lose or steal. The design separates daily signing keys on the phone from rotation power held offline on cards.

For the demo, rotation is a threshold one-of-two multisig: either of two fingerprint-backed cards can rotate the controller. One card might stay at home and the other with a custodian. Jaime backups a populated Samsung wallet onto a gray factory card and a black HID card, then recovers the same profiles, credentials, connections, and notifications onto a fresh iPhone with a single tap. The old phone's keys rotate out and the Samsung shows that the wallet moved, so the lost device can no longer act as controller.

The cards store a 32-byte master seed and a rotation index in the secure element, not raw keys, which never leave the chip. Hierarchical derivation from that seed avoids issuing a new card on every rotation while still keeping the high-risk material off the phone. Q&A with Samuel Smith covers why HDKM on external hardware cuts seed-entry attack surface without forcing a new card per rotation.

Over to you. Okay. Thank you, Karla. Thank you for having us today. I can start. My name is Jaime Caso. I work as Innovation Engineer at Veridian. I'm here with Roberto, Lead DevOps Advocate. And we are going to present Veridian Wallet with KERI protocol and a CESR-based wallet. And we're going to present two demos. Same demos that we presented three weeks ago, I think, at IIW. And yes, as you say, Karla, the first one is about wallet recovery using biometric smart cards. And the second one by Roberto is going to be about OpenID Connect login with ACDC.

I'm going to reuse my slides from IIW. Let me share my screen. Share screen. I sent a request. Okay, perfect. Perfect. Thank you. Thank you. I guess you already can see the screen, right? Yes, we can. Thank you. Let's go for the first demo. Before going to the demo, let's explain a bit about it.

So far in Veridian wallet everything starts from a secret. In KERI we call this secret the bran and in Veridian wallet we call it the seed phrase because we use a human-readable representation of this bran, of this entropy, and then from this entropy we can derive all the keys, signing keys and rotation keys. But this seed phrase has a few problems like if you lose the words, you lose the wallet or if someone sees the words, they can take control of the wallet. And it's true that sometimes it's hard to keep safe this seed phrase over time because maybe you lose the USB or the paper or even you can forget where you put the seed phrase. The idea is to separate the signing key from the rotation key. Being the signing key, as in the sig, which stays in the phone and you can use this key for your daily requests as the controller to sign and secure the communication between the wallet and the cloud agent and get access to the identities. In Veridian Wallet, we call these identities profiles. And then we have the rotation power, the rotation key, which is a multisig with a threshold one of two in this case for this demo. This means that you can use one card or the other to rotate the wallet, to rotate the controller. It's important to mention that these cards stay in different places, isolated, offline. And we don't expose these cards, these private keys, as we expose the signing key every day. We just use them when we

need them on demand. We can think that for example one of the cards could be kept safe at home and the other card could be kept safe by an accredited custodian like a bank. We have these two cards here and the gray one comes directly from China, from the factory, and here you can see the biometric sensor for the fingerprint. And then we have the second one, the black one, again with the fingerprint. And this comes from a US provider, I think it's from HID.

Let's go for the demo. For the demo, I'm going to use two devices, a Samsung and then an iPhone and these two cards. I'm going to hold back the camera of my laptop a bit so you can see how I handle the phone and the cards. Just a second.

Okay. Let me remove the blur background. No, it's perfect. Cool. Okay. And also let me share the screen of the first device, the Samsung. I'm going to start with the Samsung device, which is this one. I already onboarded the wallet and as you can see on the screen, at the top, there is a banner that says that I should verify my recovery phrase, the seed phrase. Actually, we can take a look quickly.

Okay. And these are the words, the ordered words. We want to separate the signing key from the rotation key. We want to do the backup using these two smart cards. Okay. Let's go back to home and now we need to go to settings. And in settings we have a new option here which is called backup with a card. And this is the option we need to select in order to go ahead and do the backup with the smart cards to move the rotation power from the seed phrase which is actually in the phone right now to these new smart cards. Let's go to the option and now we're going to present these two cards to the phone in order to do the rotation. And let's say activate these cards for the rotation power. Let's go for it. Let's go one by one. I click on start and now the wallet is asking me to tap with the first card. Let's go for it. I touch with my fingerprint on the sensor and I already presented the first card. Next I need to do the same with the second. I click on add as a second card. I tap. I touch the fingerprint sensor and it's done. Now, I'm ready to do the rotation and to go from seed phrase to smart cards. I can click on confirm backup and voila, it's done. We can actually double-check it. Here we see that we have the two cards, the gray one and the black one. Before the demo, before this call, I already populated this wallet with… Jared? Sorry, can you on mute? Thank you. Sorry. Thank you. No worries. Thank you. Before the demo, I populated the wallet with a few credentials, one connection and notification. So the recovery is more interesting. Now let's go for the situation where I lose my phone or someone takes it. What is going to happen is that I don't have access to this device anymore. I lost it. What I'm going to do is recover the whole wallet, all the identities, credentials and everything on a new phone. For that, I bought a new phone. Let me show this one, which is an iPhone. Oh, sorry. Oh, sorry. Let me do it.

Which is an iPhone. And I already installed the Veridian wallet. In this case, I need to do the onboarding. And I need to select that I already have a wallet. I click on the option. Accept the terms. Create the pin code.

I'm going to enable biometrics. Okay. No password this time. And now I need to do the recovery. I need to present the cards. In this case, as I said, this is a threshold one of two. I just need to present one card. I'm going to use the gray one in this case. Let's go for it. Now the wallet is asking me to tap. I need to touch the sensor with my fingerprint. And now it's asking if I still have the second one. I have it. I click on I have it. I touch again the sensor.

And one more time. And boom, the recovery is done. I can get access again to the wallet. I can go to the credentials. I have everything, both credentials. I have the connection and I have all the notifications. And one thing that is really important is that in the other phone, in the Samsung, you see the white screen saying that this wallet was moved to another phone. You recovered this wallet on another device.

For security, the keys on this phone were rotated. This phone can no longer be used. Your identity and credentials are safe on the new device. What we did is we rotated the controller and the old controller, the one in the Samsung, can't use the wallet anymore. It lost access to it. This is basically the demo. It was quick.

In IIW, there were some questions like if the signing key is a single sig. It's not a multisig. As I said in the beginning, it's a single sig. Also, there was a question about how many keys can I store in the card. And the thing is that we don't store keys. I mean the raw data of the key in the card. What we store in the card is the master seed, which is 32 bytes. Also the counter, the index of the current rotation key. And if we do the numbers, the math: in one slot of this card, we can have more than 65,000 rotations. And this means one rotation per day for 170 years.

That's a lot. And each card has actually 256 slots. This means if you do the math again, this is 17 million rotations. Of course, you will never reach that number. Probably you will lose the card before or rotate to a new card before.

And that's the demo. It was quick. I'm going to pass the ball to Roberto. Maybe… – Question, Jaime. – Yeah, definitely. – Key rotation in KERI is agnostic about… You don't need to use hierarchical deterministic key management there. It's not one seed that you have to use. But in this case, you use one seed, don't you? You use the HDKM functionality of it. – Yeah. And this seed is in the secure element of the chip. It never leaves the secure element.

And you use the index to go to the next rotation key. – Yeah. And that is a derivation from the master key. – Yeah. That's it. Got it. Can I comment on that? Yeah. By putting the derivation of the HD keychain in an external hardware HSM, then you eliminate a lot of the attack vectors where you're attacking the client that's running in a web browser or something like that when you're entering the seed phrase to be able to regenerate your keys. This eliminates a whole host of attack vectors by putting it externally. It's a balance between security and user friction. Whereas if you had to use a different seed phrase, then you'd have to have a new smart card every time you rotated. That would be really a lot of friction. Yep.

Thank you, Sam. That's all for the first demo. I can pass the ball if no more questions to Roberto. I would say this is great. This solves one of the friction problems with pre-rotated keys in a nice way using inexpensive hardware. – If I understand it correctly, you have two smart cards because if you lose one, you will not lose your access to your pre-rotated keys. – Yeah, you can have an arbitrary number of cards. This is just to simplify the demo and to demonstrate the multisig. You can have four, you can have 10. Yeah. Okay. As much as you feel safe. Yeah. And you could also do a different threshold. You could have instead of one of n in this case one of two or one of three or one of four you could do two of six and then you'd have six smart cards and any two would be required. Yep. Got it. And we can use different custodians for them. You diversify the risk of them being stolen and pulled together. I mean, they are protected with biometrics, but… No, but a lot of key recovery algorithms are based on sharding a single key. And in this case, you're not sharding a single key, you're sharding control amongst multiple keys, which is actually a more secure path. For example, you could think of your shard as each card is one of the shards of the control over your identifier. And you could have social recovery where you gave a card to your relatives and then have, instead of one of n, have three of a minimum of three that they have to get together to do it. You can replicate the same sort of social recovery, but in a much more secure way without sharding a single key, but by effectively sharding your control amongst a bunch of cards. I don't know if you guys ever thought of it that way, but that's obviously a place you could go with this.

Sorry, that's all I had to say. Sorry. Interruption. No worries. Yeah, there was one last thing I want to mention about these card providers. They are working also on the next version of the cards to support post-quantum signatures, specifically ML-DSA. The target is 2027. Thanks to KERI, to CESR and KIP-2 agility. Maybe at some point we can also support these kinds of signatures.

One thing worth mentioning: one year ago, I think September last year, in iOS 26, they included this type of signatures, ML-DSA, in the Secure Enclave of the iPhones. And then this year it was June. Also Android did the same for Android 17. This means that millions of users will be exposed to this new kind of signatures, post-quantum signatures, this new standard approved by NIST. It's nice to be aware of that.

TALK III

2026-11-24-16-01-50-KFMCA-Veridian-part-2

November 2026 Videos · talks · 21:57

SUMMARY

AI generated summary.

Roberto C. Morano, Lead DevRel Advocate at Veridian, shows a bridge that treats KERI ACDCs as an OpenID Connect identity provider. Ideal apps would speak KERI natively; until they do, his proxy verifies credentials against KERI infrastructure and emits OIDC tokens that ordinary web apps already understand.

He walks an organization manager that invites users, issues role credentials, and protects demo apps behind an OAuth proxy: an org-wide app that accepts any credential from that issuer, plus Department 0 (ACME) and Department 1 (Veridian) apps that require the matching credential. After OOBI connection and issuance into the Veridian wallet, presentation unlocks the right apps; revocation drops access on the next check even while other credentials still work. Once set up, sessions behave like single sign-on until TTL expiry or revoke.

He argues this supercharges OpenID rather than replacing it: Google or other OIDC logins can be backed by KERI credentials, including multisig profiles whose rotation keys sit on smart cards, so social and workspace access inherit stronger key control without rewriting every relying party.

Cool. As I was sharing in an ideal world… I have a bar here, but it's disturbing me. In an ideal world, we would use KERI everywhere. We could use web servers and applications that understand credentials and sessions in a different way. But unfortunately, we are not there yet. My name is Roberto Carlos and I'm a Lead DevRel Advocate at Veridian and I need to protect systems from… I need to authenticate and authorize access to systems. I decided to build something cool on KERI and that's what I call KERI ACDCs as OpenID Connect Identity Provider.

I built a bridge that connects to KERI infrastructure to check verifiable credentials and then emits OpenID tokens that applications can understand because they are native to this protocol that the internet uses broadly today. I'm going to skip the slides because they are kind of too technical and we'll go directly to the demo. I built an organization manager where I can invite people to join our organization, our KERI infrastructure, and where I can grant them credentials that they can use to access applications. I'm going to be showing how I'm leveraging my own OAuth proxy to protect some dummy applications.

In this case, an organization-wide application that will accept any credential issued by this organization manager. And two more that, let's say, they are Department 0 application for ACME company and Department 1 for Veridian, which will check if you hold a valid credential for them, not only if the issuer is valid. And then I will do a little bit more, maybe interesting for some demo that I will keep for myself for the moment. Let me invite what I call a KERI OIDC proof-of-concept user that we have under our organization. He has a Google account, but now we want to grant him some credentials he can use for some other services within our organization. In this setup, this is a dummy mail server. It's not a real email that was sent. But here we have the onboarding email he will receive where he needs to connect. And I will do that using my phone.

I'm going to skip biometrics and password for the moment. Got connected. I'm going to paste here the connection URL. This one is correct.

Cool. And now I'm going to introduce our Veridian wallet to my organization manager and the other way around. I'm going to add a connection, scan the platform QR code. The platform should automatically accept that connection and I'm going to do the other way around, also by showing my QR code. Cool. Now that we are connected, I can close this window and I can come back to the organization manager where I have a few people waiting for some credentials. Let's do that. Okay. Maybe we can try to connect to this org-wide application. And obviously it shouldn't work because we don't have any credential to present. It should say that. Let's give this user Department 0. Let's say we are an advisor for this one.

And I should get a notification for accepting that credential. Cool. Now the user is gone from pending issuance and it's now showing in the ACME org chart. This is how the credential looks. It includes an email that we could say is a verified email because we sent the user an email to onboard and he clicked on the link to verify and onboard and present himself. And this is how the OpenID Connect ID token will look. It's a preview. It's fake. But yeah, we can see there are some similarities between the credential and what an OpenID token would look like. We have here the organization and the role, etc. that our OpenID Connect native applications could use for authorization.

Okay. Let's start the new login. And now we should be able to log into this organization-wide app. Should be presented with a credential request, a credential presentation request from the app. You can see some details here.

And we're in. The demo apps, in the case of all three of them, are just a WhoAmI app where we can see we have a cookie from the OAuth proxy that checks the validity of the credentials and gives you tokens to log in. Okay. Then we can try with the ACME Department 0 and it's already logged in because the credential is valid. It automatically gave me a cookie to access that one. But Veridian Department 1, no.

This credential is not valid for it. But we could also grant this user a credential with a different role for this organization. Here it is. Let's accept it. Let me log in. Now that I hold the credential, let me log into the Veridian Department 1 application.

It requests me to present the credential and now that we have two of them, we can choose which one we want to present. I'm going to use the correct one, which is this one, showing the Veridian Department 1. If I use the other one, I will get the same result as before.

And we are in. We now have two credentials that allow us access to three different applications. What we can do now is… Oh, I forgot to show that these… and you need to excuse me because I'm an infrastructure guy and this was one of my first UIs that I put together in a long time. It's super ugly. I can revoke ACME, for example.

Even if it still shows here. This credential is gone. And that means I should still be able to access the first org-wide app. Not ACME, but yes, Veridian. Cool. That's the first part of the demo. I don't know if you have any questions, suggestions. This is a proof of concept I'm building as we speak. As you see there is a lot to polish. Yeah, I have a question about what is a one-time effort because I see a lot of switches between screens and QR codes and I wonder how an organizational user is going to be confronted with this all the time every day or is it just a one-off OOBI to do an out-of-band introduction and then you're up and running. Yeah, it's a good question. Thank you. It depends on the policy you set in your application. For example, for this bridge, I don't remember what the time to live is for the tokens. Could be one hour, five minutes, whatever makes sense depending on what application you are protecting. Each of these three run a different bridge slash OAuth proxy. So you could set different TTLs for different

apps. The good thing is this is a piece of infrastructure I will deploy for any application whether it supports OpenID or not. I'm just leveraging OpenID here because there is a convenient OAuth proxy that I use to manage sessions very easily. But it depends on your needs and requirements.

The good thing about this specific setup is when I revoke a credential, the bridge will check, for every request, if the token is bound to a credential that is still valid. So that's a lot of work and a lot of computing. I mean, not heavy computing, but per request, if you had a high volume, it can be costly. But it's also very convenient for me to revoke access immediately to any application that I'm protecting with this. That was probably too long an answer to your question. Sorry. – Well, what I got from it Roberto, is that as soon as you've set it up, it works like a single sign-on. – Yeah. It's actually as a… – Until it's revoked. Then you're done. You have to go through the process again. – Yeah. Once you are connected, your credentials become a decentralized, distributed identity provider for this infrastructure. But yeah, once you're set up, your sessions work like any other service and you don't need to do much in your wallet.

Only once the session is expired, you will need to scan to log in again, which is, in my opinion, more convenient than remembering and managing passwords which are, well, less secure, less convenient, less everything. – And why is this Veridian KERI approach an enhancement of OpenID, OIDC here. I like to say that it's a supercharge of OpenID in the sense of backing your OpenID sessions with a bit of extra security because you could… probably let me go for the next part of the demo, you could, for example, back your Google account with a multisig credential that is backed — whose rotation keys are backed by smart cards — to build more security into your current OpenID setup. Let me try this. I'll illustrate with… I should probably open a private window. Imagine that you have your marketing social media account, wherever they are, using Google to connect to X. For example, we would use OIDC POC at Veridian ID.

And now we are presented with a scan to log in that could be backed by a multisig where you will require two of N signers to present the credential. And then you're adding extra security for logging into your social media accounts. It's not the case. This is… I'm going to scan this. It's not the case because this is a single-sig profile.

Ah, sorry. I forgot Google Workspace login. It's asking me which one to present. I could present whichever because any of these has a valid email that Google will check through our bridge if it's valid. If I present my ACDC, I'm asked probably… Probably I should not put this into…

And now we are logged in to X. Instead of using a password, we're using our Google account, which is backed by some KERI ACDC. You could do that for whatever other service that allows you to log in with Google. That's a feature from Google Workspace that is from last September, September 2025. So it's kind of recent, but it's a possibility now. You will only need to configure your Google Workspace to connect to this KERI OIDC bridge and it's only four fields that you need to fill in and you are all set up to use that Google account or organizational unit connected to this IDP which will be the bridge for any service. Obviously, we are using here regular sessions.

We cannot do back-channel logout because Google Workspace doesn't implement the OpenID back channel for logging out. If we revoke the ACDC, Google won't automatically kill the session; whatever other relying party will still consider these sessions and these tokens valid. But hopefully they implement that or maybe we can — if you know someone at Google, it would be cool to be able to talk back to

Google from this bridge to say, hey, this session bound to this credential is no longer valid and you should kill any session and that should propagate. But again, sorry, probably too long an answer to your question. No, thank you.

That's all from my side for this KERI ACDCs as OIDC IDP. That's a lot of letters, but it worked. Thank you. Yay. Thank you both Jaime and Roberto for your demos. Are there any other questions from anybody on either? And just to be clear, the last demo imposed a different level of security on an existing less secure access capability.

So it's an interoperability opportunity. Yeah. Instead of using hardware keys, I mean passkeys or whatever, you could leverage all the features that KERI brings and offers to back your login with Google or any other OIDC-compatible system, which I think is very good. Great. Thank you. Any other questions?

Thank you.