ICANN72 | Virtual Annual General Meeting – DNSSEC and Security Workshop (1 of 3)
Wednesday, October 27, 2021 – 12:30 to 14:00 PDT

KATHY SCHNITT: Hello, and welcome to the DNSSEC and Security Workshop Part 1 of 3. My name is Kathy and I'm joined by my colleagues, Kim and Andrew, and we are the remote participation managers for this session. Please note that this session is being recorded and follows the Expected Standards of Behavior.

During this session, questions or comments will only be read aloud if submitted within the Q&A pod. We will read them aloud during the time set by the chair or moderator of this session. If you would like to ask your question or make your comment verbally, please raise your hand. When called upon, you will be given permission to unmute your microphone. Kindly unmute your microphone at this time to speak.

All participants in this session may make comments in the chat. Please us the drop-down menu in the chat pod and select Respond to All Panelists and Attendees. This will allow everyone to view your comment. Please note that private chats are only possible among a panelist in the Zoom Webinar format. Any message sent by a panelist or a standard attendee to another standard attendee will also be seen by the session hosts, cohosts, and other panelists.

This session includes real-time transcription. Please note that this is not official or authoritative. To view the real-time transcription, please click on the Closed Caption button in the Zoom toolbar.

And with that, I'm happy to hand the floor over to Dan York.

DAN YORK: Welcome, everyone. Thank you for coming to this session. Thank you for being part of this ongoing DNSSEC and Security Workshop that we've been doing here at the ICANN events for many years now.

I'm going to begin by just talking a little bit about the session you're viewing today. My name is Dan York. I'm a member of the Program Committee. I'm from the Internet Society, but I'm part of a group—many of whom are here on the session today; some of whom will be moderators and speakers—that we'll be talking about.

But this is the group that's brought this program together for you. If you have ideas for future programs, if you'd like to be a panelist here, if you'd like to bring research that you have done—or ideas, demos, things you've done with DNSSEC or security—this is the group to whom you will be able to send a proposal if you know any of us. There'll be a call for proposals that we'll put out in the neck and the weeks ahead for the next session beyond this.

This workshop and the associated activities are organized by the ICANN Security and Stability Advisory Committee, otherwise known as SSAC, with some additional support from the Internet Society in a few small ways.

We want to talk a bit about deployments around the world. And with DNSSEC, we talked about the validation and the signing side. This chart that you see here comes from Geoff Huston and his team there at APNIC

and shows their stats. And it shows that we're at about 25% of all queries that are around the world are being validated, are being checked for DNSSEC [and the pieces that are there].

These are some of the numbers that we're seeing in terms of regions where there is more validation happening than others. And you can see the list on here that shows where the validation is happening the most and, in some cases, where it's happening the least.

We've also seen an increased growth in the number of DS records, which is the number of signed domains that you see here, in some way. And we continue to see a nice trend going up and up. This is from the DNSSEC-Tools site, the stats part of that that's there.

We're seeing also the increasing use of DANE records for MX records which is, again, the way that we've talked about in past DNSSEC workshops where you can set it up to securely send and receive e-mail. You can secure the transit of e-mail by using DANE to provide the TLS certificates needed to go and secure that transfer.

Moving into routing and RPKI, which is another part of what we've been talking about here in recent days, we're now seeing an increased growth in the number of prefixes that are that are covered by RPKI. So we're seeing some nice growth there, continuing again this nice trend. Seeing the path going up there.

We're also seeing a number of Route Origin Authorizations, or ROAs, that are being held there. Again, curves all going in the direction that we want to see them—continued growth all around. And this is showing

the five RIRs and where we're seeing a lot of growth there. Most recently a lot of growth through ARIN this year, but others all along in this path, as you see.

As part of this workshop, for a long time we've been tracking the deployment of DNSSEC in the top-level domains across the world. And right now we're seeing a significant number—pretty much all of the top-level domains, the ccTLDs in Europe, pretty much all of them are signed and are seeing operational usage of DNSSEC records, DNSKEY records underneath there. A substantial amount all through North America, through Latin America, through Asia, many in Africa.

That's an area where we're still looking to see more growth around that, but the good news is that we're seeing a lot of this happening all around the world. One thing that you will see for the first time in our last session, but now here again is another state here called DS automation, which is where there is an automation mechanism that goes and looks, that helps connect the new DNSKEYs to their DS record in the registry.

And this is something that, we have a panel here, as part of our session here, that Steve Crocker is organizing that will be talking about the continued road toward having more of this DS automation so that we can be able to have that higher level of certainty of things.

There are a number of different places where you can learn more. DNSSEC-Tools and stats element there. For history, you can look at the dnssec-deployment.org which has some of the historical information there. And also, we look at the stats from APNIC for DNSSEC validation

in particular. There's also number of RPKI resources. All of this available in the slides, if you take a look at that in there.

So with that I'm going to say thank you very much for being here. Thank you for participating. We would love to hear your comments as you go through here. Again, as Kathy mentioned, please raise them in the Q&A pod and we'll be able to answer those during the course of the session. So with that, thank you very much. And I believe I am turning it to you, Jacques. Correct?

JACQUES LATOUR: Thank you. So I had a question. On your second-to-last slide with the counts by region.

DAN YORK: Yes.

JACQUES LATOUR: One thing we should do is show how many are present with DS out of a total—like N out of 15 or 5 out of something—so we know if there's gaps or whatever.

DAN YORK: That sounds great. We should look at that for next time. Good feedback.

JACQUES LATOUR: All right?

DAN YORK: Thank you.

JACQUES LATOUR: Thank you then. So we have a panel now with three presentations, pretty broad topics. So we have Mark Elkins on DNS/ZACR with DNSSEC automation observations. Ash and Jason are going to talk about zero-touch universal IoT device identity and how DNSSEC and DNS are instrumental in there. And Duane is going to talk about ZONEMD.

Mark, you have the virtual floor.

MARK ELKINS: Good evening, good afternoon, and good morning, whenever you are around the world. So a few ICANNs ago, I gave two talks on DNSSEC automation. The first one was from a registry point of view and the second one was from a registrar point of view. And I would like to give some observations on what I have seen since and what's happened, if I can go to the next slide, please.

So the first talk as a registry was “Gathering the Children DS”. I happened to have the privilege of running the edu.za zone. It's a non-EPP, web-based system, etc. It's very, very small. Just designed for tertiary educational establishments which are not universities. For the universities, we have the ac.za said a domain instead. So it is small. It's also free. And the growth has been minimal over the last, basically a year.

So in that time since the talk I gave, we've added five new signed domains, taking up to a total of 12 domains, giving us a 7%-ish signed criteria. The nice thing about the way that I look for CDS records and gather those children is that I end up doing a whole bunch of digging, which means I can check on things like bad name server records. So I'm glad to say at this point, as a side to signing some stuff, having some stuff signed, is that I don't have any bad name server records in this at all the moment. Then go to the next slide, please.

So the interesting thing, then, is that there are basically two organizations which do have DNSSEC signing. And that is my own organization. When a customer chooses to use me as their registrar as well as the registry, then their domain will automatically be signed. So there are eight domains like that, and obviously all are signed.

And then 10 domains happened to use Cloudflare. And as I presume people are aware, a person can register a domain, or rather transfer a domain, to Cloudflare from a DNS point of view and allow Cloudflare to run their DNS. And there's a button somewhere there that you can click on to get DNSSECs signed up, etc.

And then the nice thing about finding the CDS records is that I can pick them up and put them into the zone file all automatically. So of the 10, 4 are assigned. What this sort of suggests to me, and I'm very well aware of this, is that the people who are using the edu.za system for domain names are not necessarily tech-savvy type people. And then we have another talk, if we can go to the next slide.

The next slide is me working as a registrar. So "Registrar DS Management for (essentially) DNS Providers". Strangely enough, over the last few months I've actually gone down in the number of domains that I look after from a registrar point of view. And I suspect that COVID actually has been taking its victims. The social system has been taking its victims. And I personally lost a friend which also lost me six domains as well because of his passing.

So anyway, literally all the domains where I host the DNS locally are signed when possible wherever it makes sense. So that is something that I can state. I also, obviously, am a registrar for domains that I don't run the DNS for, and we'll look at that in a second.

Again, a very similar bunch of software which means, essentially, I can see who has bad name server records and I can proactively do something about that. If we go to the next slide, please.

And a little bit of—I should turn this into two slides, really—some context. So I'm a non-ICANN accredited registrar, but accredited with ZA and a couple of other ccTLDs. I'm a reseller for DNS Africa. And that's a fully automated EPP system, so when I do get DS records the way I upload it is by EPP. And I'm a reseller with others. And that's kind of nice. I've got access to about 720 possible TLDs that I can sell to.

I am a small organization, 1,514 domains. So of the 521 domains that I manage with internal DNS, 366 domains are signed. So that's 70% DNS signed. And that is something that I've been pursuing. The ones that are missing would be ones that are in transition, or I'm actually doing the

DNS but I'm not the registrar, and other strange things like that. 43 have no zones because they're basically subdomains of existing domains.

And then the bulk of my domain is where I am the registrar but someone else is the DNS operator. And right now, 27% of those are signed which I'm very, very happy about. Of those 949, 163 domains are using Cloudflare. And my success rate is that out of that 163, I have 154 that are actually signed with Cloudflare. And so that's all being brought in. And then there's another 106 other domains which are signed.

What I've been doing is telling people about the ability of BIND, for example, to run a domain—run it and get it signed, etc. And a lot of people seem to be quite happy doing that. But then again, in South Africa something like over 50% of all recursive resolvers are also DNSSEC-aware. I'm sure someone else can confirm that number.

So out of a potential of 1,470 potential signable domains, 635 are actually signed. Which means we have a success ratio of 43% signed domains which I'm very, very happy about.

If we go on to the next slide and look at some more observations. What is this thing that I run? Well, I have two very, very similar scripts, and they both run at around about the same time. They go and dig every remote domain for CDS records. Sometimes you find that the DNS at the far end doesn't know what a CDS record is, but more often than not, we get back some form of answer.

If it's an existing signed record—and so I'm checking for updates, etc.—then that would be via a DNSSEC-aware resolver. Otherwise, for the

non-signed ones I'm doing a TCP dig to one of the name servers that's listed and looking for CDS records there.

If I come across a CDS Null record, then I remove the DS records on my side. Otherwise, the CDS is mirrored to the DS record and installed into the parent. And the way I also do that is that new CDS records are recorded on my side and I keep going back for up to three more times. And if there are no changes, they're deemed to be valid. So that's how the authentication part works. And if we move on to the next slide.

Cloudflare is very, very interesting. As far as I know, they are the only large-scale organization where you can give them your DNS and they will sign it for you and no charge, which is always very, very favorable with people. What's nice is that they sign with Algorithm 13, elliptic curve. They only add the CDS record, Digest Type 2, to the zone. However, they always use Key Tag 2371.

And that's a bit of a shame because that short numerical value which is meant to be able to quickly help us identify the referenced DNSKEY record, well, it's always the same. And I only realized this about two or three weeks ago, as well. I presume it is the same around the world and it's not a southern African issue.

They also support the addition of a CDS Null record to signify that the DNS has been withdrawn. So that also needs to be built into the code algorithm that I was talking about before.

I would be interested to know whether many other people that provide free DNS services also automatically have the ability to—DNSSEC—sign the zones as well. Carry on.

And that really is the end of my presentation. So if there are any questions either now or later, I'd be happy to obviously go through that. Thank you very much for this opportunity.

JACQUES LATOUR: Thanks, Mark. We're about 10 minutes ahead, so if there are questions now, we can take one or two. And then we'll move to the next presentation. If not, we'll get Ash and JSON to talk slow—

KATHY SCHNITT: Oops.

JACQUES LATOUR: Oops. Okay, so if you think of a question, at the end we'll have time. There's a question in the chat. It should be in the Q&A pod. "Does ZA support initial CDS discovery?"

MARK ELKINS: No, it doesn't. There are very, very few ccTLDs or SLDs that do that. And remembering, I'm actually just running an SLD rather than a ccTLD here on my side.

JACQUES LATOUR: Then the next question, we can handle at the Q&A. "Geoff Huston, can you explain further on the weakness of the common key ID used by Cloudflare in DNSSEC [signing] hosted domains?"

MARK ELKINS: I would love to ask Jeff that question. Yes, definitely. Good question. [inaudible].

JACQUES LATOUR: So Geoff, you can answer that question at the end of our panel unless you have answer right away. Oh, he can't talk.

Okay, so we'll go to the next presentation with … Okay. He has no answer. So the next presentation from Ash and Jason, zero-touch universal IoT device identity. You've got the floor.

ASH WILSON: Okay. Is this coming through clear for everyone?

JACQUES LATOUR: Yep. It's good.

ASH WILSON: All right. Thanks for the intro, Jacques. And thank you everyone for taking the time to attend our session. Jason and I are going to talk about zero-touch universal IoT device identity.

Before we dive in, I'd like to like to thank all of the organizations involved in this. This was a joint effort across three organizations—TELUS, CIRA, and Valimail. And TELUS was gracious enough to provide us with the hardware which was a Raspberry Pi HAT that supports the IoT SAFE SIM cards. And so that's been hugely helpful.

So when we talk about IoT security, that's a really broad topic. And so kind of narrowing that down and looking at security aspects that have to do with identity, it's generally agreed that using a PKI-based device identity is one of the best things you can do for your identity strategy. It enables a lot of security features.

And so a few of those are that you can implement end-to-end message security. And that's where the message originator, say in a decoupled application, attaches a message signature that's verifiable by the message recipient. This is very similar to how S/MIME works, but sort of tuned for constrained networks like IoT, dealing with message brokers, things like that.

And then the sending entity can also use the recipient's public key to encrypt that message. So this is a really good way to mitigate against middleware compromise which is something that, if you've watched the news, this still happens.

Transport authentication is using that device certificate as a TLS client certificate in the TLS handshake. And this is easier in some ways to protect than just using shared secrets. And so you can use a hardware secure element to wrap the cryptographic functionality around that

private key so that you can use it for establishing a TLS connection without having to expose the private key to the operating system.

And network authentication is very closely coupled with transport authentication because, well, in this case we're talking about EAP-TLS. So this uses the TLS protocol to authenticate a network client before allowing that client to fully join a network.

The really nice thing about this, again, it gets us away from pre-shared keys which, within the context of Wi-Fi, if you have to rotate your home Wi-Fi shared secret, then that's cumbersome enough having to rotate for 10 or 20 devices. But on an enterprise scale, that's really, really difficult to do. It's much nicer to allow access based on PKI-type identities so that you can admit or revoke based on that certificate instead of blanketly sharing the same secret with everything that needs to access to the network.

And so why don't we just do this everywhere right now? As great as that is, there are interoperability and cost barriers that keep this from being adopted, especially in small IT shops. And so certification authority is not … There are costs associated with that. To set it up, to operate it, to maintain it and appropriately protect it—there are costs involved. Either you're running the infrastructure on your own or you are consuming that as a SAS-delivered service. But it's not a zero cost thing. And your on-site IT staff need to understand that well enough in order to troubleshoot. And that's not something that you kind of pick up in school. That's the kind of thing you learn on the job. So the skillset to support that is not as common as we would like it to be.

And also, when you introduce and CA or a PKI into an application, you're introducing a new namespace as well. And so when you have multiple namespaces that don't have a way of enforcing identifiers across …

For instance, one certification authority can internally enforce consistency among its own namespace, but it has no way of preventing another CA from signing a certificate against the same entity names. And so the risk of identifier collision presents a risk of impersonation. So that's why very often you end up, if you're doing PKI-based client identity, then your onboarding everything to the same CA. And so that costs time and it's just a lot of overhead. And if we could get away from that overhead for the operator, it would be great.

The message security aspect. So talking about "decouple applications," the actual certificate discovery aspect of this kind of difficult. You very oftentimes end up with a proprietary integration between whatever you're consuming application is and your certification authority. And so maybe that's a restful PKI integration, but there's no standardized way to access those certificates.

So either you're creating a custom integration or you're jamming the certificate into the message itself. And especially in IoT applications, this kind of a waste of bandwidth.

Transport and network authentication. In order to make that happen, you have to share trust anchors. With the way that TLS currently works, you have to make sure that you have all of your trust anchors in place.

And you also should operate an OCSP responder or distributor publish a CRL.

It's not hard to imagine how this is sort of a daunting task for, especially small organizations. But taking all of the complaining I just did and setting that aside for a minute, what would we really like a universal identity to look like? An ideal state for a universal client identity would be where a user or a consumer could purchase a device or purchase a secure element with a pre-provisioned identity that just works.

Think of this like a passport as opposed to a library card and allowing the CA to be involved earlier in the supply chain so that the small IT shop doesn't have to run their own CA. And then having that universal identity would enable roaming not just across different network access methods like Wi-Fi or 5G, or even hard wired ethernet, but also across network providers.

And so this would be using the same universal device identity, that digital passport, to access your home network or to access a municipal network or take a device into the office for "bring your own device" and being able to classify that and get that attribution. But a client identifier that is also useful for not just accounting for access, but even for accounting for a chargeback.

And so being able to do this, especially the chargeback aspect, allows the … We think would maybe even incentivize network operators to support such a protocol. And of course the accounting side to actually attribute bad behavior to an organization that owns the identity. And

then also making revocation simpler to signal than operating and OCSP responder or generating and distributing a certificate revocation list.

These are all things that we'd really like to be able to do with a universal device identity. but right now support just doesn't exist in the ecosystem. So digging into the problem, what we've kind of suggested is that the lack of a binding namespace is sort of problematic. And the Cloud Security Alliance, in their summary guidance for Identity and Access Management for the Internet of Things, had a good long list of "these are things that you should be concerned with."

Step 1a. "You should define a common namespace for IoT devices." And I think that everybody can probably agree with this. But the challenge is that in this first step of the journey, very often application architects will create a whole new namespace. And this might have some sort of taxonomic representation that "this the device from this manufacturer that does this thing, and here's the serial number at the end." It can be as sophisticated as that or as simple as just the hash of a public key. And there's a lot of nuance in that.

But let's, just for a second, back up and say what if we didn't create new namespaces for clients for every single application. What if we just used a namespace that we had been using for server identities for decades? Because here we have this perfectly good DNS and everybody agrees on this namespace. What happens if we just use DNS? And so the challenges that get in the way of doing really good PKI-based client identity sort of evaporate. And that's thanks to DANE.

So the absence of a universal namespace. That problem goes away when we all agree on DNS and DANE, which is built on DNS/DNSSEC. And using DNS records to present certificates or information about certificates gives us a discoverability. So being able to do that lookup and to pull that certificate is super, super, super useful. So this is how DANE actually addresses the challenges with IoT device identity.

So looking at the problem and where we would like to be, what's the next step as we kind of look at what would be a really good first step into getting into this without having to, for instance, change the way that the TLS handshake happens in order to support this new protocol? Can we get to a DNS-bound client identity and show success with that? Or network access?

And so the minimally invasive approach that we came up with is, well, let's look at EAP-TLS and let's see if there's some way that we can dynamically manage CA certificates while still enforcing the DNS name bindings. And so this was an interesting challenge. The best common practice for implementing EAP-TLS on a RADIUS server is that you should only use one CA certificate. And the concern is conflict or identifier collisions between the identities that are trying to access your network.

Fortunately, there are some sort of hooks in the process where you can do validation beyond the CLS handshake. And we'll dig into that in just a second.

This what our proof of concept environment looks like. On the left, you have TELUS who is shipping these SIM cards with pre-provisioned

identities. Those identities are represented in CIRA as a hash. And we use that because CIRA is also, via the IoT Registry, functioning as a certification authority.

We also present the certificate under the organizations like the identity owning organization's namespace. And we present the entire certificate there. We present the entire certificate to enable the object security use case. So the signing and encryption for end-to-end message security. We have that there. And we also present the trust anchors via a web server. And we'll kind of go into detail of why that's there and the way that it is in just a second.

And also there's sort of an optional component here. We have a policy server that we use just for ease of managing network clients, but in the code that we're releasing this is something that you can configure in a static file, actually, in the RADIUS server itself. And there are a couple of automation components that also exist on the RADIUS side that we'll dig into, as well as some adaptations of the supplicant or the wireless client software that Jason will go into detail on in a few minutes.

So first I'm going to give a brief overview of the protocol by which we manage and make sure that those identities are bound to the correct DNS names. Beginning from the policy, the policy represents, say, all of the DNS names of clients allowed to authenticate to a network. And so for each of those DNS names, we go out and we grab the certificate from DNS. And from that certificate, we extract the authority key ID which is a hash of the public key of the CA certificate that's next in the chain for authenticating that identity.

So we grab that certificate. We pull the authority key ID out of it. And from the name of the device and the authority key ID, we can pose the well-known URL. And that's the box in the upper right-hand corner of the screen. We're using a web server as content-addressable storage, essentially, using that public key hash. And so we're able to pull the trust anchor using that pattern.

And the trust anchors for all allowed identities are stored in the ca.pem file. And this is to support the TLS handshake on the server side. Remember, we didn't want to alter the way that the TLS handshake works just yet, and so this enabling existing infrastructure via automation to use DNS-bound climate entities.

And then we also store the hash of the certificates that we pull from DNS in connection with the name use to retrieve it. And that's to minimize the DNS interactions in the authentication process.

On the server side, when authentication happens, the TLS handshake and EAP-TLS happens just as it always has. But we have this pre-populated ca.pem file that contains all valid trust anchors for all clients that might want to connect. And so what we do right after the TLS handshake happens is that we have a second step that’s sort of like an authorization step.

And what this does is compares that certificate that's presented against the mapping of the identity to the hash of the certificate to make sure that we don't have some sort of cross-domain signing issue because we are dealing with some PKI complexity.

And then as a final step before network admittance, we check that the IoT Registry still presents that identity is valid. So this is how we can get revocation at the speed of the TTL because if the IoT Registry pulls that record from DNS, that machine will not authenticate to the network. And so it just goes silent.

So talking about a process is all fine and good. However, what's even better than that is a demo. So what I'm going to show is part of this protocol in action.

The part of this that I'm going to show you before I jump right into the guts of it is … I'm going to show the logs from a RADIUS server that I have running on a Raspberry Pi. And I'm using a free account with a service called Balena. And what Balena allows us to do is just contain our orchestration. So this is just to make things easy to look at.

This is my RADIUS server logs for my home network, and this is my iPhone. I'm about to turn on my Wi-Fi. And you can see as it tries to join. And I'm just now allowed onto my network. This is all driven via a policy that could be statically configured on the RADIUS server, but I'm actually using an external system with a web interface to drive it.

But the idea here is that in the background, every minute or so, it does a synchronization of the policy to known good hashes, and then also manages that CA certificate file. That's driven off of this integration right here. There are two pieces. There's the trust management piece which manages the CA certificates and the trust mapping which is what we're referring to as the mapping between the DNS name and the

certificate hash. And that's just so we're not making so many DNS requests every single time a device connects.

And then there's the verification. This part right here is called by the RADIUS server after TLS authentication successfully completes. Then this is called, that examines the certificate. And it also performs the check against the IoT Registry.

So now I'm going to get back in here. I'm going to turn it over to Jason who's going to talk us through the client and supplicant side of this.

JASON BLAKEY: Thanks, Ash. I'm Jason Blakey. I'm a Senior Dev at CIRA, working in CIRA labs. What I'm going to be talking about today … Ash has kind of given the very everything-kind-of-view, but I've been working on a smaller part of the system. The part I've been working on is interacting with the IoT Registry.

But we've talked about the IoT Registry in ICANN before, so if you'd like to look up those previous presentations that are available, one's from ICANN69 and one's from ICANN71. And I'll have to apologize. I'm fighting a cold, or what I hope is a cold. It seems to be a cold so far. So I may sound a little rough or I may take a cough break now and then.

So what I'll be talking about today is mostly wpa_supplicant and the IoT Registry and IoT SAFE. Could you go to the next slide, please, Ash?

So what I have been working on for the past few months is a proof of concept as part of Ash's overall system. So we have an IoT Registry in

place right now that allows for device provisioning on the fly for device provisioning, zero-touch provisioning other than the power cycle, as well as secure identification and validation of a certificate on the backend.

So we put this together and it's up and working. We've been using it on a couple of POCs, but we were looking for another POC to use it on. And this seemed like a good POC to use this system on, a way to identify and confirm identity from the IoT's point of view. Next slide, please.

So to get the overall flow, you should probably know a little bit about how provisioning works on the IoT SAFE applet, if you haven't heard about the IoT SAFE applet before. So when I have this applet on a SIM in a cell phone or in a modem card on a Raspberry Pi—I'll power on that device—the IoT SAFE applet will startup. It will generate a public and private key pair if it hasn't already done that before. And the magic of the IoT SAFE is that private key remains on that applet. It is unreachable just like an HSM.

It then sends out a request using TELUS to get a certificate based on its public key. I get a certificate back. It stores that into a slot in the IoT SAFE. And on the backend, when that request for that certificate is received in our CA, it will also store a DNSSEC record for that public key—either a [CERT] record or a TLSA record, depending on the use case.

So at this point on the IoT SAFE, we've got a public/private key pair, a certificate, and whatever provisioning data, whatever credentials we wanted to send to that device. Maybe there's an application that's

running on the device that needs to log into an MQTT server somewhere, and it needs to know how to do that. So we can provision that remotely as much as we would like to. Next slide, please.

So at this point, we've got a fully provisioned device with a known and verifiable identity, thanks to its public key being stored in the Domain Name System. So we're using DANE. DANE is giving us this, and what can we do with this system? Next slide, please.

So the POC, the part of the POC I've been working on, a lot of different parts on this thing. But we thought, wouldn't it be interesting if you could have a device that instead of the user having to go through some kind of interface on their phone and Bluetooth, and tell it the Wi-Fi connection and all the rest, we could remotely provision that device with Wi-Fi credentials so that it could log in by itself? As soon as it was provisioned and powered on, it could connect to Wi-Fi.

So we mixed up a POC using these parts. And the most important part here, really, I would say is .. Well, the hardest part to get a working, I'll say, was between wpa_supplicant and WolfSSL. Everything else, pretty easy, pretty standard. But WolfSSL is the first provider to market, I think, with IoT SAFE support for their TLS stack. And getting it working with wpa_supplicant is a bit of a dance right now, but it's getting better. It's getting better. Is there anything else I wanted to say here? I don't think so. Next slide, please.

So this is the overall flow of the Wi-Fi login process. If you start in the lower left-hand side, the IoT SAFE applet will power up. The IoT SAFE will be there listening for calls from wpa_supplicant. Wpa_supplicant,

using WolfSSL as its TLS stack, will interact with the IoT SAFE, retrieve certificates, access indirectly the private key, and talk enough EAP-TLS to the OpenWrt router to get a connection started.

Now that OpenWrt router, in my case, is statically provisioned. But in Ash's case, it will be dynamically provisioned with all the policy decisions that you might want to put into the system. Once that's in place, you can have yes or no connectivity completely hands-free for that device. The device is provisioned indirectly or directly through—hands-off is what I'm trying to say. And it can be disconnected hands-off as well. So it's very nice.

I think that's all I wanted to say on that slide. Next slide, please. I think that's back to you, Ash.

ASH WILSON: Great, thanks. So while this kind of represents a first step into using DNS-bound client identity … And there are a few reasons that we couldn't go full-DANE across the stack. Probably the biggest thing is that we haven't yet defined what the TLS handshake looks like for a DANE-represented TLS client certificate.

So the future, like the overall vision for this … What we're showing you is, we think, something that … It works now, and it's a really good stepping stone to getting to that DANE-everywhere. And once we have that TLS handshake defined—how that's supposed to work—and once that is worked into the EAP-TLS process, I think right now we're still trying to … Or maybe it's been adopted by now. TLS 1.3 support for

EAP-TLS has been kind of a while coming, but once we have DANE as sort of a first-class protocol supported inside that TLS handshake, then even a lot of the stuff …

Even though this represents a step forward in consolidating to the DNS namespace, the actual supporting infrastructure to make all this happen, a lot of that supporting infrastructure just evaporates when we can go DANE deep into that TLS handshake inside the EAP-TLS protocol. So we're really excited about this.

IoT SAFE is a great way to provision a very secure identity in a way that makes it accessible to, we think, small shops, small IoT groups or even IT support inside an organization that maybe doesn't have the ability to run their own CA or the budget to consume that from somewhere else. They can buy the identity from a provider and just use it on their network. And that's [part of] the big win with IoT SAFE.

The call to action, the reason that we're here, is that we would really like, if this is something that's near and dear to you and you'd like to get involved, we have a working group that was just established at IETF called DANCE. And that stands DANE Authentication for Network Clients Everywhere. And of course we're really excited about it, and we hope that this talk has gotten you charged up as well. So, yeah, now we can kick over to the Q&A.

JACQUES LATOUR: Thank you, Ash and Jason. We're seven minutes ahead, so if you have any question, please go ahead.

ASH WILSON Sorry, I had too much coffee and I talked too fast.

JACQUES LATOUR: Yeah, I know. But we were 15 minutes, so now we're down to 7. So that's pretty good. I see a question from [Sharkawi] in the chat. Do you want to ask your question live? Just unmute.

ASH WILSON: Okay. So I think, starting at the top, we have a question. "Users like to install their security solutions instead to purchase pre-installed IoT solutions." I'm not sure of the question in that.

JACQUES LATOUR: So the question is, "Users like to install their security solutions instead to purchase pre-installed IT solutions." So I'm not sure where the question is there.

ASH WILSON: Yeah. I think that, kind of speaking to that, we would like the user to have a choice. This is not incompatible with an organization that wants to run their own certification authority. So if an organization already has a CA, then this is just as simple as tying an automation in between the CA and DNS in order to publish those records.

But it also opens it up to organizations consuming universal identities from another provider if that organization is not able to operate their own CA, for whatever reason.

JACQUES LATOUR: Okay. Next question. "Could the IoT device be equipped with cellular network and eSIM? Could the IoT device be equipped only with the network interface and eSIM?"

JASON BLAKEY: Yeah, it certainly could. Right now we're using a Cat-M1 network to get our provisioning data. And also in a previous POC, we were using Cat-M1 for [outward] data as well. And if the eSIM supports IoT SAFE, it should be all happy about it. So, yes.

JACQUES LATOUR: I think [inaudible] you did a presentation yesterday or the day before—I can't—[inaudible] on identity management. So maybe you can reach out—there are synergies there—and integrate in a little bit of both together.

JASON BLAKEY: Great. Thank you.

ASH WILSON: And the last question we have. “I understand that there's a problem with DANE and DNSSEC on IoT devices—TimeSync and Bootstrap and root key rollover. And how do we plan to solve it in this case?”

So we very specifically left out DANE for server identity for the network access use case. And that's partly the reason that we have the DANCE Working Group formed because there are some challenges, especially with …

The way that EAP-TLS works is that the TLS handshake happens without the benefit of IP connectivity. So the client actually can't reach out to the DNS server to pull the authentication part for DANE. The client can't do [it] until it's admitted to the network. And so there are other related proposals for delivering the chain. Like the DNSSEC authentication chain being delivered in the TLS handshake would allow that non-IP connected device to still do DANE validation of the server. But that protocol is still not adopted, and it's something we're hoping to discuss in the working group.

And also, making sure that you have the root key on both ends of that [inaudible] so you can do server and client validation for DNSSEC, that is a challenge. And you know, there are different ways to look at it. If you leave an IoT device on the shelf for many years and you have a root key rollover and then it's put directly into service, in my mind it would be a matter of good hygiene or best common practice to make sure that you're running updated software on those IoT devices.

One of the challenges that we've had in the past, especially with botnets is old software all over the place and not having security

patches to mitigate some of those holes. So you could argue that if best common practices are being followed and you're running those device updates before you put it in service, then that challenge can be mitigated. But this is definitely something that bears further discussion.

So thanks for the question, Hugo.

JACQUES LATOUR: Any other questions? If not, we'll go to Duane. Thank you. Virtual clap. That's what we're missing in our virtual meetings.

ASH WILSON: Thanks.

JACQUES LATOUR: Duane, the floor is yours.

DUANE WESSELS: All right. Let me share my screen here. Look good?

JACQUES LATOUR: [Yes, sir].

KATHY SCHNITT: It looks good.

DUANE WESSELS: Okay. So thanks Jacques and to the Program Committee for inviting me here to this presentation today. This is about message digests for DNS zones and a little bit about some plans for adding the message digest to the root zone.

So this first few slides are sort of very introductory to ZONEMD. I apologize if you've seen this before. But if you haven't, a DNS zone digest is a cryptographic digest, or a hash, of all the data in a DNS zone. And one of the nice properties about it is that it's embedded into the zone data itself as a ZONEMD record. So it travels with the zone and stays with it. The digest is computed by the publishers of a zone, and it's verified by the recipients of a zone.

So a lot of us have probably seen something like this. This is a screenshot from an Ubuntu website where there are these ISO files for download. And there's also this SHA256SUMS file which is a checksum of all these other files listed here.

So the ZONEMD record is very similar to that. It's a checksum [inaudible] digest. But unlike in this case where that information is in a separate file, again, it goes with the zone and stays with it.

So the ZONEMD protocol and record format are defined in RFC 8976. Very briefly, the way that it works is that all of the zone data is given as input to a selected digest function. And in order for this to work between all parties, there must be a well-defined and consistent ordering of the data as it's given as input. And it must be in a well-defined and consistent format. So in this case, it's basically the DNS wireformat for all the data.

The calculation of the digest excludes the ZONEMD record itself as well as its signatures. So the digest is included into the zone and then, ideally, for signed zones, that adds extra assurances about the provenance of the ZONEMD record.

So, some reasons why this is useful. The ZONEMD digest protects the zone data at rest. So this is data security versus channel security. It means that if somebody gives you a zone that includes the digest, if they give it to you via e-mail or via thumb drive or via FTP server or whatever, you can verify that the data is authentic.

It's expected to be useful in distributing zone data between primary and secondary name servers, especially in today's sort of modern and complex environments where there are lots of any cast sites. Or maybe an organization is using multiple DNS providers and that sort of thing.

Of course, we're also interested in applying ZONEMD to the root zone. One of the reasons is because there's a lot of interest recently in serving the root zone data locally, which is defined in RFC 8806, or also talked about as a hyperlocal root sometimes.

And there are a couple of sort of anticipated uses of ZONEMD that don't really follow the traditional use of zone files from like primary to secondary name servers. One of those, for example, is the Centralized Zone Data Service. This, of course, and ICANN service where you can go and download zone files. And it may be useful, when downloading such zone files, for you to know that you got the data as published by the zone operator.

Another one is something called the response policy zones which are a feature of certain recursive name servers. And one of the reasons it's interesting here is because response policy zones, although they are sort of zones in format, they don't really fit within the DNS namespace and they aren't signed with DNSSEC in the same way that more traditional zones are.

So some more details about the ZONEMD record. Here at the top is an example record. You can see that, in this case, this for the zone called example. There's a TTL. There the IN class. And then the ZONEMD record type.

And then there are four fields. The first field which is highlighted here is the serial field, the serial number. And in the ZONEMD record, there's a serial number that must match the serial number from the SOA record. If they don't match, then verification using this ZONEMD record is failed.

The next field is what we call the scheme field, and this sort of specifies how these own data is processed as input to the digest function. The RFC at this point only defines one scheme which is called the SIMPLE scheme. There are some others that are reserved for private use, but it’s possible that in the future there will be more schemes defined, probably to support larger or more complex zones or zones that have lots more frequent updates.

The third field is called the Hash Algorithm field. This is essentially the digest function, and the RFC defines two choices here. SHA384 and SHA512, and then reserves the number of private-use bullet points as well.

Then of course the fourth field is this large, hexadecimal blob which is the digest value, the output of the Hash Algorithm.

For the SHA384 protocol, this is always 48 octets long. And it's always 64 octets long for SHA512.

For other algorithms, that may be a different length. There is a requirement in the RFC that for any Hash Algorithm, it must be at least 12 octets in length.

This table shows sort of the recent state of known implementations of the zone digest protocol. It's maybe a little bit out of date. I think things are evolving sort of quickly and some of these may be a little bit more farther along than represented here. So as were working on the RFC, there was an implementation that we called ldns-zone-digest which you can find. We used that for a lot of testing and whatnot. That's available on GitHub.

The Unbound recursive name server implements ZONEMD. Unbound is not designed to publish zones, so it doesn't calculate the digest as a publisher. But it does implement verification as a zone recipient.

Similarly, the LDNS library can do both adding ZONEMD record to zones that should be published and verifying zones that are received. Those are done in the ldns-signzone and ldns-verify-zone programs.

There's a set of tools from NIC Chile Labs called DNS-tools which can do both functions. PowerDNS Resolver. I believe they are working on and may be very close to release of a version of the software that implements verification similarly for Knot Resolver. I know the folks at

ISC for BIND are working on this as well, but in the release versions they can only parse the records and not actually verify them. And similarly for the Net::DNS Perl module. It knows how to parse the records.

If anyone knows of other implementations, I'd be happy to be told about those and add these to the slides.

So this graph shows some benchmark results that were done a while ago now, maybe a year ago. This is really taking all the that we could get from CZDS and adding the root zone as well, and then calculating the digest for all those and recording the time and then graphing it here.

So you can see at the very top, there's the .net zone. The .com zone, we actually didn't [attempt it]. The hardware that we had access to wasn't able to load it all in memory. The zone was a little bit too big for that. So the .net zone is the largest one there. And you can see that it's pretty predictable in terms of performance.

For the root zone, to calculate the digest on this hardware that uses the time, it was about 100 milliseconds.

This slide shows output from enabling, or the use of the feature in Unbound version 1.13.2. So if you have the configuration line sort of shown here, if you add an auth-zone stanza and if Unbound finds the presence of the ZONEMD record, you'll see these lines in bold down at the bottom where it says that the hash was correct and that verification was successful.

So I'll talk a little bit about some tentative plans for adding ZONEMD to the root zone. There's a committee within ICANN called the Root Zone

Evolution Review Committee. And last year RZERC published a document with some recommendations to the Board for adding the ZONEMD record to the root zone with sort of these four key steps.

Number one is that the Root Zone Maintainer and Root Server Operators need to verify that the addition of the record will in no way negatively impact the distribution of root zone data within the Root Server System. The DNS Internet community should be made aware of plans to use the ZONEMD record for the root zone and given opportunity for feedback.

Developers of server software are encouraged to implement the feature and consider enabling it by default. especially when the software is configured to locally serve root zone data. And then PTI and the Root Zone Maintainers should jointly develop a plan for deploying the ZONEMD record in the root zone and make it available to RZERC for comment.

So a lot of these things are underway already. Some of the root zone operators have already been able to confirm their readiness for ZONEMD. As we've seen, some of the implementers are making good progress on this, although that is not strictly a requirement. And Verisign and PTI are in the process of developing this plan and hope to give that back to RZERC pretty soon.

So tentatively we're thinking along these lines, that there will be a single ZONEMD record in the root zone. Initially, there would be included with it one of the private use algorithms. This is similar to what was called the DURZ. When the root zone was initially signed, DURZ was

Deliberately-Unvalidatable Root Zone. And so the idea here is to publish the ZONEMD record with a Hash Algorithm number that doesn't really enable validation and only tests whether or not the presence of the record leads to any issues.

So after that initial period, maybe lasting a month or two, then the record would be published as a SHA384 digest. This could possibly start as early as the second quarter of next year.

One tricky part about adding the ZONEMD record is that for situations where the zone is processed in its presentation format or its text format, whether or not that should be as a native ZONEMD record or as a generic/unknown RR format. So the two examples here are equivalent. The first is, of course, the native record format. And the second is the generic format.

When the zone is distributed within the Root Server System, this is not really going to be a problem because that always occurs via zone transfer and in wireformat, not in in the presentation format. However, we think that there's probably a number of folks that get the zone file on a regular basis from, for example, www.internic.net or some other source. And they may be using tools to process the zone data. And those tools may not understand the ZONEMD record format. So probably out of caution, it would be better to use the generic format to reduce the chance of disrupting anyone's existing processes or use of the zone files in that way.

But this is something that I would definitely like to have feedback on if folks have opinions one way or the other about this issue.

That's it. Happy to take questions. I think I'm probably early. Right, Jacques.

JACQUES LATOUR: Yeah. We're five minutes ahead. So 15 minutes left for Q&A.

DUANE WESSELS: All right.

KATHY SCHNITT: Jack, we've got someone with their hand raised. It looks like it's Brett. [inaudible] ask your question.

BRETT: Yep. Sorry, I had to change myself to Panelists [inaudible]. Can you hear me all Right?

JACQUES LATOUR: Yes.

BRETT: Okay, good. Duane, thank you for the presentation. Very interesting. It looks like the slide where you had implementations, it looks the implementation are fairly in their infancy at the moment. And they're obviously moving forward. But, for instance, BIND is listed as "parse only" extension.

Does it not need to be the case that all of the Root Server Operators will need some kind of implementation to be able to check the ZONEMD record when the zones are transferred. Or am I misunderstanding something?

DUANE WESSELS: So in my communication with the Root Server Operators, I try to make it clear that the thing that they need to assure is just that the presence of the record doesn't cause problems.

BRETT: Right.

DUANE WESSELS: We're not really asking them to turn on validation of the ZONEMD record. They're certainly welcome to, but it's not a requirement at least at this time in the initial stages.

BRETT: Okay. You'd like them to do that eventually, but it doesn't need to be done at the first step. And obviously, as implementations [grow], then they will be able to do it.

DUANE WESSELS: Yes. I would definitely like, eventually, to get to the point where all of the Root Server Operators have enabled this and can verify that they're getting the correct data to all of their systems. But certainly understand

that it may take a while to get there and get comfortable with this before they all turn that on. But I would like to see that as an eventual point.

BRETT: I think it would be useful if you and all the Root Zone Operators came back and did further presentations on how well lots of gone for the information for other people.

DUANE WESSELS: Yeah. I'd be happy to do that. And I assume others would as well. Yeah.

BRETT: Thank you.

JACQUES LATOUR: So we have a question in the pod, first, from Andres. Want me to read it?

DUANE WESSELS: So, Andres' question is, "Is the ZSK used to sign the ZONEMD record?" Yes, that is true, Andres. The ZONEMD record will be signed using the same processes that all the other records—of course, other than the DNSKEY—are signed by the Root Zone Maintainer. So, yeah, it will be signed by the ZSK.

JACQUES LATOUR: And Warren had his hand up for three seconds. So let's see what he wanted to say. He's gone. Oh, no. He's still there.

DUANE WESSELS: Warren said he cannot unmute.

KATHY SCHNITT: One second [more,] Warren. Try now.

WARREN KUMARI: Okay. Yeah, I was just going to follow on from what Duane was saying initially. Oh, sorry. I should mention that I'm a co-author on this draft, although, as per usual, Duane did almost all of the work.

DUANE WESSELS: Oh, come on.

WARREN KUMARI: Yeah. Just mentioning that, initially, this is just a resource record like any others. And root operators or whoever would just serve it like any other.

Eventually, root operators might want to validate the record before deciding to serve the zone. But it's still unclear, if they determined that it wasn't valid, what they would actually do. Perhaps they would still want to sign the zone or try and fetch it again or something. but at least they would know that something is potentially not right.

And it would be very sad if, for example, the ZONEMD record got corrupt and all of the Root Server Operators decided not to serve the zone because of that. So the exact behavior, when it doesn't validate, is still to be determined for the root operator. [inaudible].

DUANE WESSELS: Yes. And I will also add that today within the Root Server System, the distribution of the root zone from the Root Zone Maintainer to the Root Server Operators is sort of protected by the use of TSIG keys. So that offers some production today from corruption and whatnot.

KATHY SCHNITT: Jacques, [inaudible]?

DUANE WESSELS: He's muted.

JACQUES LATOUR: I'm muted. Any other presentations either on Mark, Ash, Jason, or Duane? Brett still as his hand up.

KATHY SCHNITT: Brett, did you have another question? Just unmute.

BRETT: Yeah. It was just a comment, really, on the first presentation, Mark's. There was a question from Geoff about the risks around Cloudflare using the same key ID everywhere. And the thing I just wanted to mention was that they're not just using the same key ID everywhere. They're actually using the same key everywhere. So it's the same DNSKEY for every signed zone as far as I can tell. Obviously, Geoff's question was "what risk does that present." I feel proud to be able to answer one of Geoff's questions or a change. It's usually the other way around.

Obviously, that presents the risk that if that key is compromised anywhere, then everybody's zones are compromised, I would guess. But I have no information how Cloudflare protects their keys or how their network runs. But that's just what my view is from the outside.

No answer required. It was just a comment.

JACQUES LATOUR: Okay. Well, this concludes this panel. So we have a 37-minute break, and we gather back here again at 21:30 UTC?

KATHY SCHNITT: That is correct. I put the link in the chat. It's the same link. You can either stay on or jump off and come back. And we will see you for Part 2 at 21:30 UTC.

And with that, please stop the recording. Thank you, Jacque. Thank you, panelists.

[END OF TRANSCRIPTION]