ICANN75 | AGM – DNSSEC and Security Workshop (1 of 3)
Wednesday, September 21, 2022 – 13:15 to 14:30 KUL

KATHY SCHNITT: - the ICANN Expected Standards of Behavior. During this session, questions or comments will 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 and speak. All participants in this session may make comments in the chat. Please use the drop-down menu in the chat pod and select "respond to all panelists and attendees." This will allow everyone to view your comments.

Please note that private chats are only possible among panelists in the Zoom webinar format. Any message sent by a panelist or a standard attendee or to another standard attendee will only be seen by the session's hosts, cohosts, and panelists. You may access all available features for the session in the Zoom toolbar. And with that, I'm happy to hand the floor over to Jacques Latour.

JACQUES LATOUR: Hello. Thank you. All right. Welcome to the ICANN 75 DNSSEC and Security Workshop. Next slide. So this is our hybrid workshop for ICANN 75. Next slide. For the Program Committee, for the DNSSEC Workshop, there's a bunch of us. We meet on a regular basis once a

week to build the content of this workshop. So it's based on three sessions today. Next slide. The workshop is brought to you by SSAC. So there's a few of us in SSAC and then some help from the Internet Society to build the program. Next slide.

The agenda today is here. I think there should be a piece of paper everywhere with the agenda. So we have three parts to this session. I'll go quickly with the first part because I've got two sessions starting at the same time—the session one and the session 2.0. They're starting now. So we'll try to go quicker and we might have a shorter Q&A session. We have the DNS Provisioning Automation Session with Steve Crocker and Shumon. They're going to host that session. And then, at the end, we'll have an ICANN session. Next slide.

So we'll go around the world for DNSSEC, DANE, and RPKI development. I think this is the graph from APNIC on DNSSEC validation. The trends are upward, slowly, but upward. So I guess that's not too bad. Next slide. So the adoption by region. I think what we see here is we're between 20 to 70% but the average around 40%. So we still have a lot of work to do to get validation turned on in other region. Next slide.

One of the good thing here is DS record count. The line is going up. So the trends, you can go to this site, dnssec-tools.org. For .ca—that's where I'm with—we went in this range. We're at about 70,000 signed domains with .ca. And this line is following our trend closely so these are good news. Next slide.

DANE, MX. There's a use case to encrypt e-mail, SMTP, with DANE. So this trend is going up a little bit. Millions of domains are signed. So this is the news here. Next slide. RPKI, so the number of prefix v4 that are signed. I think the trends are going up a little bit here so that's a good sign. We still have a lot of work to do to sign the whole routing space but we're making progress here. Next slide. Same stats, longer trend. So next slide.

All right. So DNSSEC deployment status for ccTLD. If we look around, this is the stat generated by Dan York, the Internet Society. So we have a system to measure the deployment of DNSSEC with ccTLD. Next slide. There's a new status that we have with the measurements. We have seven ccTLDs today that support DS automation, so that CDS/CDNSKEY automation. We have a session later on today with Steve and Shumon on that kind of stuff so we have seven. So we've got a long way to go before all the ccTLDs support this but it's a new metric that we're measuring. Next.

And then we've got a couple of other status, which is experimental, announced, partial. So I guess the important thing here is if you see your ccTLD in this list and you are fully operational, if you're not experimental or announced, please let us know. Yes. That's right. Next slide.

These are all the ccTLDs that have DS in root. So the difference between DS in root and operational is if there's a registrar or a method for a registrant to be able to sign a domain by getting the key in. So if you are fully-operational, please contact us, the Program Committee, or Dan York. And next slide. So if you have questions or tools, these are

a couple of websites you can go to for DNSSEC or RPKI resources. So very quickly, that's the DNSSEC and security around the world. Wes?

WES HARDAKER: Yeah. Really briefly, thanks, Jacques. Always a good presentation. The graph that you showed in the beginning, from the stats.dnssec-tools.org's website is now actually much more interactive. You can go play with them, and hover over the graphs, and it'll give you details, and you can compare multiple TLDs against each other at once and stuff. So it's gotten a significant revamp lately. Actually , the collection system has also gotten a significant revamp lately. So it's even more usable than before, just FYI.

JACQUES LATOUR: Okay. Good. Thank you. So yeah. Sorry for going super quick on this. We're 15 minutes short on the agenda so just a quick high level. All right. So this is my presentation. A couple of months ago, I did a presentation at IETF on trust registries and digital identity. And there was a lot of confusion at the end. So the preamble, I didn't do well the first time. So the goal of today's session is to talk about digital identity trust registries. That's the wrong presentation, is it?

KATHY SCHNITT: Jacques, that's the one you sent me.

JACQUES LATOUR: Oops. Can I share mine, yeah?

KATHY SCHNITT: Yep. You can.

JACQUES LATOUR: That works, yeah?

KATHY SCHNITT: Yes. It works.

JACQUES LATOUR: All right. Good. So this is my second attempt at this. I'm Jacques Latour. I forgot to do that part. I'm Jacques Latour. I'm CTSO at CIRA.ca, top-level domain. For almost the last year, I've been working in Canada with a bunch of people on digital identity—with DIACC, with Trust Over IP, with a bunch of different organizations. We are trying to figure out what the role of trust registries are. So the outcome of this slide deck is, in my view, how do we bring together digital identity, the DNS namespace, and DNSSEC? To address one of my big concerns with all of this was the unique identifiers.

So verifiable credentials. Digital identity is based on verifiable credentials. That's the key thing here, is that if you have an entity that uses a verifiable credential and then it goes inside a wallet, a holder, they have a copy of that verifiable credential. So a digital driver's license, a digital version of a passport, a digital copy of a health card. So an individual would have a copy of that in their wallet, inside their phone. And then they have the ability to selectively present some or all

parts of those verifiable credentials to a verifier. So that's the framework for digital identity. You've got the issuance, you hold, and people can verify the credential.

From a realistic use case, at CIRA or .ca, we have a Canadian presence requirement process. So for some domain, we need the registrant to prove that they are Canadian. And today, the way to do that is they send us copy of passport, or a driver's license, or a birth certificate. And we don't want any of that. All we need to know is they are Canadian—either their nationality is Canadian or they live in Canada.

So with the digital identity system, we're building a proof of concept with this scenario where the Canadian government would issue digital verifiable credential for a driver's license, birth certificate, and all that. And then a .ca registrant would have a wallet. So they would have a process to grab the verifiable credential from the issuers. And then they would be able to present to us only passport, just to show they are Canadian, or that they live in Canada from their driver's license. It's all based on verifiable credential. That means it's a digital thing that is signed by someone.

This is a proof of concept that we're working on. The issues that come out of this all the time is how do you know that verifiable credential are authentic? Were they issued by the proper government entity or the issuer? Is the wallet trusted to have driver's license? Is CIRA trusted to view a passport from somebody? How do people handle all of this process?

And in this ecosystem, that's where the trust registry comes in. If you look at the Trust Over IP model or a bunch of these different models, trust registries, they're a thing. Supposed to take care of all of these things. From experience, what I've seen in the digital identity world, people focus on the issuer, wallet, verifier. That's where bulk of the work is done. Very little is done in the registry side up to now.

So if you look at trust registry, it's a directory, a hub, repository of all the entities that are part of a trust group. So a trust registry—for example, academia in Canada—it would have a trust registry for all the universities that can issue diplomas or certificates. The governing authority would be the Canadian University Association. The governance model would be that all accredited Canadian Universities that are allowed to issue certificates or diplomas would be listed in a trust registry. That's the concept of a trust registry—the group, its governing authority, scope, and model, all around.

So the challenges while working with the digital identity folks is we're not going to have one trust registry. We're going to end up having hundreds of trust registries in Canada. So Canadian fishing, Canadian Bar Association, academia. There's going to be a lot of different trust registries. And when I look at this, working in that environment, we're missing a unique identifier system for all of this. It was pretty hard to try to understand what the framework was for making sure that all the issuers had unique identifiers. And then the trust registries were all unique—that you'd be able to find each other. So there was a big gap in the unique identifier.

So if you look at the chain, you have the credential on top. That's a digital thing that is issued by an issuer. It's signed by an issuer. Then the yellow box is the issuer of the credential. That's the entity like a government or a university that would issue the credential. But what's missing in this scope is there needs to be some sort of national trust registries. So in Canada, we need to have a trust registry that has other trust registries in there. So we need country code national trust with different registries by organization, one for Canada, global, and a global trust anchor.

And the reason we started to look at this is this would ensure unique identifiers. All the verifiable credentials under this structure would be unique. So that would allow global interoperability, global unique identifier. And why not use the DNS for some of the credential issuers? Then we could have DNSSEC on top of it to make sure it's authentic. So if you split it, it looks like this. So you could have country code, the global structure. You could have country code site with trust registry and then commercial side with the industry-related trust registry.

So that's how the digital identity is forming today. There's a lot of small islands being formed with a like type of digital identity environment. But there's no way of a global framework. So if, in Canada, somebody comes in a bar and they present a driver's license from Poland, there's no way to date to figure out the chain of trust, and if there's global trust registry, and all of the relationships between issuers and the trust. So this platform, I think, meets the needs for the trust people.

In this example, many organizations, they are looking at having a domain name as a unique identifier. But they didn't understand how to build a chain of trust. They didn't know that there were mechanisms in the DNS to be able to do all of this—to find trust registries to see if an issuer is registered with a trust registry or to find digital identity documents. This is a generic term but the detail of an issuer.

So in this example, a verifiable credential—it's the box here. That's signed. That's a document that is signed by an issuer. It could be a diploma issued by a university—example.ca University. It's very famous. And it's got a public key that can be used to verify with. That makes up a verifiable credential. So this could be a cert with this or a PKI certificate. It could be , potentially, the bullet we see, DID, type of digital document. But the point is, it's signed. There's a public key. And then, in the DNS, we've built a bunch of stuff to be able to do validation.

So with a TLSA record with DNSSEC and DANE. The issuer, like vc.example.ca, that issued that credential on top, would have a TLSA record with a public key certificate. So in DANE, you could have a public key of the issuer. You could use that TLSA record to make sure the records are authentic.

So that's fairly simple. All of this, we do today. There's nothing new to invent here. It's just if vc.example.ca issues a bunch of university diplomas. It signs it with its key. It puts a key in TLSA record and boom. You can find, from the credential, from the domain name, the

public key and make sure it's authentic. That works. That's a really good use case for DNSSEC for that.

The new part is if you have the university diploma and you want to know what trust registry is associated with that issuer. So let's say you're an employer, like CIRA. We get a digital diploma presented to us. It's signed by vc.example.ca. We know to go look in the Academia trust registry for that university to see if it exists or not in that trust registry. If it is, then the issuer should have a TR or a trust registry RR type that says vc.example.ca is part of trustregistry.ca.

So that issuer could be part of two different trust registries in Canada. And itself, trustregistry.ca could be part of another trust registry—so global country code trust registry, whatever. So there's a hierarchy of trust registry, all the way down to the global trust anchor.

So with this, this is a simple DNS query. You can go look at a verifiable credential, find out what trust registry it belongs to. You know which one you need to trust for your application and then you can make sure it's affiliated with that. So this record plus a DNSSEC signature, RRSIG, it creates an authentic record that says, "This is valid."

The next step is you need to go look in the trust registry to make sure that the issuer is indeed part of that. So is example.ca part of the trustregistry.ca and the academia-trust-registry.ca. So there's two trust registry records here to say, "I'm part of two trust registries." Here, each one should have a label, _trustregistry. That's a new label. That's an easy part. And then it says vc.example is indeed registered in the trustregistry.ca with the TLSA (0 0 1). That's a hash of the public

key. So with this, a trust registry can say, "This entity is indeed registered in the trust registry with the DNS record, signed with DNSSEC."

So from a global point of view, if we had something like that, we could have a verifiable credential, a university diploma, this example. It's signed by vc.example.ca. Then you can have an affiliation to build your chain of trust with trust registry, all the way down to a global trust anchor in the root zone. So at a high level, this is the structure where a verifier can, in the DNS, check the full chain of trust if the verifier has the IANA … If they are DNSSEC trust anchor like everybody else, using this structure, they could figure out trust registry on a global basis.

And from experience working with the digital people, this addresses a trust issue with people in their ability to issue a verifiable credential and their ability to have the public, the people, trust that they are authentic in the way that they can understand. And the DNS platform, I think, is good for that.

So that's my deck. I guess I could do five minutes. Or I'll let the next presenter go, then. We'll do questions at the end if we have enough time.

KATHY SCHNITT: Thanks, Jacques. Can you stop sharing? And next up is Viktor.

VIKTOR DUKHOVNI: Okay. Let me share my window here, if I can find it. It's not offering me to share anything other then the whole screen so I guess I'll just share the whole screen. Let's see. Do you see my slides?

KATHY SCHNITT: Yes. We do.

VIKTOR DUKHOVNI: Okay. So I'm going to talk about something I've humorously called small bang DNSSEC. You'll see why shortly. For some reason, the slides … Ah. There we are. So I'm going to describe who are the key players in making DNSSEC work. I'm going to talk about the way DNSSEC works today where certain very dangerous and risky big events happen at some point during your DNSSEC deployment and you just hope that it all works out well. I'm going to talk about domains for which that's not entirely a satisfactory model because the risk is too high and so they hold back on DNSSEC deployments

Then I'm going to talk about some ideas where especially registries but also, in part, the rest of the DNS community can perhaps help us make progress and reduce the barriers to DNSSEC adoption. Then I'm going to ask for ongoing work in this area to see if we can actually make things work more smoothly.

All right. So as everybody knows, DNS is a complex ecosystem. We have registrants who treasure their domains, and work with registrars to get them enrolled, and then can make various changes through the

registrar, changing their NS records, and if they're doing DNSSEC, changing their DS records.

We have registrars who manage the relationship with the user—ideally, secure their login, and authenticate them properly, and all of that, and notify the users of important changes and other aspects of keeping the domains up and running, build them, and so on. And also mediate updates of the data related to the registrant into the registry via EPP. And finally, we have registries that are part of this model. They manage the relationship with the registrars. They operate databases. They serve the DNS and do all kinds of wonderful things for us.

And then lastly, left out largely in the cold, we have the DNS operator who actually operates the registrant's domain, often not the same party—somebody they contract to do it. And they really play no formal role in this model. They don't have established relationships with any of the three Rs, which we know can be a bit of a problem. So that's our ecosystem. In this ecosystem, we deploy DNSSEC, which involves all of these parties getting together and doing the things they need to do.

The child zone. At very low risk, the zone of the registrant can go ahead and sign all of their records, create keys, automate various rollovers, and so on. And all of this happens completely transparently and doesn't actually secure their zone until these DS or delegate signer records are actually communicated to the registry. This is where things begin to get interesting.

Sometimes, the DS records have to be entered into the registrar webform, which can be confusing and can make it easy to make mistakes when you're uploading your DS records. And then your zone goes dead because the DS records don't match reality. In any case, the mistakes can be discovered even later. But when the DS records do go to the registry from the registrar, the registry very rarely performs any validation of these. So they'll happily break your zone, make it be unavailable for anybody who performs validation.

Long TTLs on the DS records leave very little slack for errors. If something goes wrong, you face one or two days of downtime in many cases. And when you do try to back out, given that complex situation of the previous state hanging around for a long time, it's very easy to make mistakes that further exacerbate the problem. So that's big bang DNSSEC and it is risky.

I've already talked through most of the second slide. But basically, we have poor validation. We have no opportunity for timely rollback. And the zones that really care about having constant uptime are reluctant to play.

Here's an example of this. I don't particularly have anything against Ethiopia. But Ethiopia is a top-level domain that is internally signed as we see in the picture on the right from DNSViz. Algorithm eight, strong keys, everything's looking good. On the left, though, we see that the root zone has an NSEC record, which if one hovered over it in the real DNSViz website, one would see that this NSEC record proves the nonexistence of DS records for Ethiopia. Ethiopia is not delegated

signed even though it's been signed internally, if you look carefully at the dates, for already over a year.

So here we have a top-level domain that's been operating DNSSEC for well over a year, perhaps two or three. I didn't look that far back. And yet, it remains unsigned as far as the rest of the world is concerned. Something is holding them back. I don't know if it's quite the issues I listed but they probably play a role.

So critical zones. Critical zones are zones where the users and customers, or simply the huge number of people who depend on it, expect always-on service. For these zones, each minute of downtime carries substantial costs and reputational damage. If a top-level domain like .com went down for a day, that would be really bad. They probably don't want to do that. Of course, .com is signed and has been signed for some time so they manage the risk. But not everybody is willing to be quite so brave.

Critical zones disdain changes that can't be rolled out incrementally and regionally. One little bit at a time, and as you gain confidence, eventually roll it out globally. They also often have a culture of, "Sure. You can make changes but be prepared to very quickly roll them back." Before you even figure out what went wrong, at the first sign of trouble, you return to the previous. And then, at your leisure, you can start debugging, and figure out what went wrong, and try again later.

Big bang DNSSEC is not compatible with this operating model of incremental rollout because the DS record is rolled out globally. It happens everywhere at the same time. And it sticks around for about a

day because registries, by default, publish these DS records with very long TTLs. So as a result, at least in part, critical production zones are reluctant to deploy DNSSEC. We see that. In fact, most of the DNSSEC deployment is new registrations of domains. Domains that have been around for a long time and that are providing critical services to lots of users are sticking with plain old unauthenticated DNSSEC in large numbers.

So I want to suggest a new way of getting domains to sign up for DNSSEC, which for this talk at least, will be called small bang DNSSEC. In small bang DNSSEC, we do prepublication validation of DS records at the registry if this is something that we can get the community to agree to. We will publish DS records with short TTLs and we will assure registrants that if anything goes wrong, rollback to a prior working state or to no DS records at all are quick. So let's drill in on this a little bit.

Prepublication validation of DS records. This is something which both the registrar, and especially the registry, can be helpful in. It's not enough for the registrar to just provide a webform and whatever nonsense the user types in, that's what they send in EPP to the registrar.

Ideally, there'd be some validation of this data against the live nameservers of the zone at both steps, both in the registrar and the registry, to make sure that the data is correct and will not render the zone invalid. And of course, there are some details about multiple algorithms, and what happens if one of the servers is down, and so on. So this will need to be worked out.

Validation may perhaps need to be an opt-in for something that a high-value domain or a critical domain would choose to sign up for. Before they deploy DNSSEC, they would say, “Please validate my DS records, if at some point in the future I decide to have some.” The reason for this is that some domains may want to deliberately publish, for test purposes or for various other reasons, either algorithms that the registry doesn’t yet understand or deliberately broken records so they can perform tests.

So not every domain will have validated records. Perhaps some will not. But at least it should be possible to request that and to have that be a sticky property of a domain, that its DS record changes will be validated.

One might even propose that DS record changes should only be accepted if the zone happens to publish CDS records that match those DS records so that we have both an EPP request and a CDS record matching that EPP request before the DS changes are accepted, whether they be fresh, from-scratch new DS records or changes in existing DS records. This, again, would be too strict as a default. Not everybody wants to use a CDS model.

And there’s potentially issues to work out as to how this interacts with CDS probing. If you need CDS records for validation, does that immediately then cause them to be probed and published, even before you are ready to send the EPP message, take them live. So there are some issues to think about. But nevertheless, that’s something that we can consider.

And again, once we're talking about validating critical changes in a domain, NS records, if you publish the wrong ones, can also be quite damaging if mishandled. Perhaps those should be validated in much the same way. While we're doing DS records, do NS as well.

And lastly, in the registrar/registry/registrant model, by and large, there's very little interaction directly between registrants and registries. But we have registry lock is one example of at least something that skips the registrar and works directly between the two edges. Does validation play into that? Can you even sign a registry-locked domain or is it completely frozen? Some of these issues need to be understood.

So short TTLs. These are critical. The rollback on first sign of trouble culture requires changes that break something to be able to be rolled back in as little as a minute, or at most, two, or three, or five, or something. But certainly not hours or days.

So the proposal that I'd like to suggest—and this primarily would affect registries—is that DSRR sets would get a short initial TTL after any change, not only the initial creation of the DS records but also any future change that introduces new DS records, removes some, adds new ones, and so on. As soon as the DSRR set changes, its TTL drops. And then it gradually increases to the default value that a stable, long-term TTL gets after it's been around for some time.

This increase may be just a single step or it might happen in a few steps. This is all to be worked out in discussions with the community and registries as we figure out how this process ought to work. The

idea would be that each time a DS record is re-signed, a DSRR set is re-signed. And it hasn't changed since the previous time. It's an opportunity to extend its TTL to the next value if it's artificially deflated for being recent.

So again, should this be something that's applied to all child zones or should this be something that's opted in? In this case, I would think it might actually be a good global default, but again, subject to discussion with registries—.com, .org, ccTLDs, and so on. I'd love to hear your thoughts and certainly would encourage, at some point, experiments, and running code, and so on in this space.

And then, one might even contemplate signaling of the TTL the child zone wants. Maybe the TTL of the CDS record is one way to signal the desired TTL on the DS record in the parent—probably not the best idea but who knows. Let's be creative about what the correct ways of getting this to work that are sufficiently flexible might be.

Rollback. I want to be able to undo mistakes or discovered issues. If one deploys DNSSEC, and for some reason, it turns out to have been problematic—whether it got deployed correctly or not, something downstream broke—then we certainly want to be able to return to the previous working state as quickly as possible, which means minutes, not hours or days.

This , of course, presumes already a short TTL. It's no good being able to quickly yank your DS record if the previous one is stuck in caches for a long time. But the short TTL is not enough. One suddenly needs to be able to remove the incorrect DS record that will then quickly be

flashed from caches if it's reasonably new and replace it with one that carries the correct data or with none at all, so denial of existence of DS record, I guess.

So we want to better understand what are the SLAs on the timeliness of changes that a registrant would request from the registrar to undo DS changes. How quickly do registries effect these and allow the state of the world to return to something that hopefully will closely approximate prior working state?

This may be, in part, already covered by existing SLAs, at least for gTLDs. I don't know if there are any published SLAs for ccTLDs, whether they are uniform. Again, this ought to be at least something that a registrant can understand as to how quickly can they expect changes to be published if they need to be quickly performed to undo a problematic deployment.

Next steps. So this is just a set of half-baked ideas at the moment. But I think they're essential if we really want DNSSEC to be deployed, not just by small and brave individual domains but really by the large infrastructure operators who run the bulk of the Internet services and are very worried about deploying DNSSEC on their domains because of the very high cost of potential outages, however unlikely they might be.

So if anybody has additional ideas of what we can do to go beyond the ones I outlined in these slides, that'd be great. But in particular, I'm looking for feedback and ideas to work with registry operators—.com, .org, .net, .gov, whatever—to make this a reality.

There may need to be some involvement from ICANN if some of the contractual issues get in the way of some of the things we need to do to make this happen. The operators of authoritative zones may need to create and manage new signals, if signaling of some kind from the child zone is appropriate or needed. Critical zone registrants may have other ideas of things they need in order to reduce the barriers to entry. And of course, the DNS community at large may have useful ideas as to how to make all of this happen.

Thank you. I guess I can take questions now or maybe at the end of the session. But I'd like to hear some discussion. There was a related effort discussed at IETF 114 about dry-run DNSSEC. I'm not sure that one's entirely practical but it's certainly, at least, thinking along the same lines of trying to reduce the risk.

We've already mentioned the statistics page to see how DNSSEC is being rolled out. Again, as I mentioned, mostly it's being rolled out in terms of new deployments of domains that are introducing new small-scale services. We don't see much onboarding of DNSSEC for long-established domains, unfortunately. And that's what I want to address. And if you want to keep track of my various rants on DNSSEC and DANE, there's my Twitter feed. Thank you.

JACQUES LATOUR: Perfect. Thank you, Viktor. You're on time. So what we'll do is do Peter, his presentation on CDS consistency. And then that should leave us 10 minutes for Q&A for all of us.

KATHY SCHNITT: Viktor, you've got to stop sharing your screen. Thank you.

VIKTOR DUKHOVNI: Yeah. That's what I'm looking to do. I don't know. I think I'm done, right?

KATHY SCHNITT: Yes. You are. Thank you.

VIKTOR DUKHOVNI: Thank you.

PETER THOMASSEN: Hello. I'm Peter Thomassen, this time with the hat of SSE, Secure Systems Engineering, a small IT security consultancy from Berlin. I'm going to talk about the consistency requirements that CDS and CDNSKEY or CSYNC records should have because if you don't ensure a consistency of these records across nameservers, you will end up with failure modes that are probably not intended.

All right. So let's see what this is about. For everybody who doesn't know, CDS and CDNSKEY records are records that are placed in the child zone. The CDS records look like DS records and they indicate what the child wants the parent to public as DS records. The CDNSKEY is in DNSKEY format and it's also possible for the parent to query it and then compute the DS records themselves.

Yeah. So this is [code coffee], as the Germans would say. So it's been specified for a while. It's RFC 7344. Also, similarly, there is CSYNC type records, which indicate to the parent that something else needs updated in the delegation, most notably NS records or perhaps glue records, which are the IP addresses for nameservers under the same domain in the same subtree. This is also [code coffee], RFC 7477. And yeah. It's very good to use that if a DNS provider decides to change the nameserver hostnames or perhaps their nameserver IP addresses. It's also used to update the NS record set of the parent to the new record set that is needed after you do a provider change.

Yes. I just had a thought. Whatever. So the RFCs don't specify how exactly the parent should be doing the polling. And it's quite tempting for the parent to just fetch those records from just one authoritative server or perhaps user-validating DNSSEC resolver that they trust and effectively just retrieve these records from one server. In many cases, that may be fine. But in any case, it does not ensure that these records are consistent across the different authoritative servers. And that may lead to undesirable consequences.

So let's look at what can go wrong. There is two main failure scenarios. One is in multihoming, so when you have your domain hosted by several DNS operators. This affects both breakage in the DS record set and also in the NS record set. Let's first look at the DS record set. If you have that in a multihoming scenario, that's called multi-signer if you have several operators signing, and cross-publishing each others' public keys and all of that.

So let's say one of the two providers performs a key rollover and then accidentally publishes only their own CDS or CDNSKEY record set in the child without taking into account that they should also be publishing the other providers' CDSKEYs. Then, when that is consumed by the parent, that will lead to the DS record set being replaced with a set of records that only refer to the rolled keys of the one provider and the other provider's keys are gone. So some queries, of course, will end up at the other provider and then the responses can be validated. So the domain, at least for a significant fraction of queries, is effectively broken as far as validation is concerned.

Similarly, you can have NS breakage. This is not so much about cryptographic stuff. But if one of the providers publishes an incomplete nameserver record set—for example, after they changed their hostnames for their name servers—and again, forgets to include the other provider's name server records, and then requests an update via CSYNC, the parent scans for that and initiates an NS record update in authentication. That leads to the other providers' name server records to be removed from the delegation's nameserver record set.

And again, it's broken. It's not as badly broken because you can still resolve the domain and all queries will be answered by just one provider. But you lose the properties of a multihoming setup, silently, and you reduce to a single-provider setup which probably would be unexpected. And probably, one provider shouldn't be in the position to affect that.

Then the second failure scenario is during provider change, which is actually a special case that's on the first slide. But they deserve extra consideration because in the future, if at some point multi-signer automation should be available, then this may end up being a common case. So the provider change, if you want to leave the delegation secure without turning off DNSSEC, it requires a brief multi-signer period where you start with the old provider. Then you onboard the new provider into the multi-signer scenario. Then you have both. Then you offboard the old one.

That works by the old provider importing the new provider's DNSSEC parameters and related records and vice versa. And when you have done that and the zones are synchronized, announcing each others' DNSSEC parameters on the authoritative servers, then you can update the trust anchor in the parent domain, the DS records. And once that has propagated, then you can update the nameserver records and include the new provider in them.

The problem is what if the new provider fails to sync the CDS and CDNSKEY records properly, specifically doesn't include the old provider ones and only the new ones, then when that is queried by the parent and consumed, the old provider's trust anchor will be removed from the DS record set in the delegation but not yet from the nameserver record set in the delegation but not yet from the nameserver record set.

So some queries would still be directed to the old provider but the trust anchor is removed so validation, again, is broken. That shouldn't happen. And a single provider should not be in the position to remove

other providers' trust anchors that would mess with the separation of duties in an appropriate way. I don't know if there's a consensus on that but I would assume there is consensus on that. If there isn't, please speak up.

Fortunately, this can be remediated by ensuring consistency before you process or when you process these C-type records. As I said, validation breaks down if a single provider makes a mistake, undermines the multihoming guarantees, like operator independence. But it can be solved if the parent is careful.

There is a general strategy that can solve this, which is pretty simple. Everybody who does CDS, or CDNSKEY, or CSYNC scanning, I would suggest should adopt this strategy. The strategy is you do the querying and scanning as before but you do it from all authoritative nameservers. Then you ignore those which don't respond or which tell you that they don't support this type of query.

It's possible, for example, that the nameserver provider has two nameservers. One of them serves CDS and the other just doesn't, for whatever reason. Maybe it's old software. But if the answer is proof of nonexistence of CDS, that's also fine. But all the responses that you do get, they must be consistent because otherwise you end up with the failure scenarios earlier.

So ensure that those are consistent and refer to the same set of keys for CDS or CDNSKEY, or that for CSYNC, that they refer to the same set of name server records. When you have that, you can continue

deploying the parent records. But if you find an inconsistency, it is important to abort and to avoid the breakage explained earlier.

I wrote a draft on this, which is going to clarify the related RFCs that I mentioned two slides earlier. So this is in the process. I proposed it to the IETF DNSOP Working Group and let's see where it goes. But whether that is going to go anywhere and getting accepted or not, I wanted to make all the ccTLDs specifically that are doing the CDNS scanning today aware of this kind of issue. I don't know how far consistency checks are done today. I am afraid that probably they aren't usually done. And I just wanted to bring that to everybody's attention.

You can find the details in this document. And perhaps one day it will proceed somewhere. Until then, I'm happy to take any questions you might have.

JACQUES LATOUR: And you are done, right?

PETER THOMASSEN: Yes.

JACQUES LATOUR: So two minutes before your scheduled time, which was supposed to be 10 minutes later. So we now have 15-minute Q&A.

PETER THOMASSEN: Back to normal schedule.

JACQUES LATOUR: Thank you. Yes. So any questions?

KATHY SCHNITT: Jacques, we have a comment that was left in chat from Mats. “Instead of set short TTL on DS it is possible to sign the zone and have tools that test the zone using the planned DS.” So that was just a comment.

JACQUES LATOUR: To Viktor, right? Viktor’s slides?

KATHY SCHNITT: That is correct. Yes.

VIKTOR DUKHOVNI: Right. My response is that the planned DS isn’t always the DS you end up with because of potential mishandling of the DS en route from you to the registry. For example, via webform, somebody might typo it. Or if you send an e-mail to your registrar, in some cases, or otherwise there’s a manual process that can go wrong.

And in any case, issues can happen, even if your domain is signed properly. Yet, at some point, you discovered some critical application is just unable to handle DNSSEC for some reason, even though it’s all valid and you need to go back. You wanted to do it. But shucks. DNSSEC is breaking something important and you have to go back

and figure out what that is and try again at a later date. So the short TTLs address all kinds of concerns, ones that you can't necessarily test until things actually go live.

JACQUES LATOUR: Peter?

PETER THOMASSEN: Yes. Maybe to expand on that comment, I believe it was last October that Slack tried to roll out DNSSEC and they encountered a problem, I believe with some wildcard signing at route 53. Some problem happened. And then resolution failed for I don't know how many, if all, but at least a significant fraction of slack.com queries. And the issue there was not with the DS record set. So if you had had some software testing, the appropriateness of the DS record beforehand, which they perhaps even did, they wouldn't have caught the problem. Regardless of how that incident was then managed, I do think it is a good thing if, in such situations, rollback can be as prompt as possible.

VIKTOR DUKHOVNI: Yes. Some of you may have notice the word "slack," used just as an adjective on one of my slides, kind of sneaked in as a subtle hint. But anyway.

PETER THOMASSEN: That's a good one.

JACQUES LATOUR: So there's a question in the chat. No, in the Q&A pod.

KATHY SCHNITT: "For the average user, setting up certificates, digital signatures, and other security-related setups are far too complex. Are there any designs in progress that make it all as simply as clicking a button that says, 'Secure and certify my computer?'"

JACQUES LATOUR: Good question? Anybody.

VIKTOR DUKHOVNI: Securing a given computer isn't really securing a domain. So in some sense, the question is really a completely separate set of topics. But to the extent that one wants to operate a domain, certainly the tools for signing a domain, if you're operating it yourself, are getting a lot better. They still could use ongoing improvement but there's real investment and effort going into that space.

And certainly, if you outsource your domain to an operator, often they will just take care of all the details. You can, in fact, just check a checkbox, "Do DNSSEC for my domain." And many an operator, especially also the registrar, will do all the work for you. This is increasingly the case. The issues I'm concerned about are all the domains that are much more complex and have been around for a long time. But yes. That's my feedback anyway.

KATHY SCHNITT: Peter, did you want to add something?

PETER THOMASSEN: Yes. I agree that for security DNS domains, automation is not ubiquitous but is getting better. So I don't know if it's fair to say let's wait for a year or two but I expect things to improve.

But regarding certificates that you mentioned, I think it's important to distinguish between machine certificates and personal certificates. For machine certificates, you have things like Let's Encrypt, which already had considerable automation, and of course, in many cases, one still has to run some tools. But here is also server software that has that already integrated. For example, for web servers, there's the Caddy Web Server. And you can just turn on HTTPS on the Caddy Web Server and it will obtain a Let's Encrypt certificate without you doing anything at all.

So I think this situation is getting significantly better. I don't know when Apache, or Ngnix, or some people like that are going to implement such functionality. Perhaps it already exists. I don't know. But also, if you look at all the cloud providers that offer hosting, they usually come with certificates automatically. So that already has also improved by a lot.

So I would, at least as far as machine certificates are concerned, disagree slightly with the assessment that currently it's very complicated. It has been much more complicated and it's on the way to get better.

Regarding personal certificates and securing your computer, I think that is not defined strictly enough. It depends a lot on what use case you have and each of those has to be automated individually.

JACQUES LATOUR: There's a comment in the chat, "Clearly this needs a blockchain." Anybody care to …? You want to elaborate on that?

VIKTOR DUKHOVNI: Jacques, I think that was actually for you on your trust registries. And in a way, I actually agree though because if I get a university diploma and I want to present it to somebody 30 years later, the TLSA record, 30 years later, isn't going to match the key that they signed my diploma with. So in fact, something like a university issuing really long-term credentials probably do need to publish some sort of Merkle tree of all the diplomas they've issued and whatever. So designs get more complicated.

In fact, there may even be room for blockchains in this space or something resembling them, anyway. But that was, I think, a comment on your slides.

JACQUES LATOUR: There's also a trust issue that people are seeing, having too many different, isolated blockchain ledger environments and you don't know where to go to find that.

VIKTOR DUKHOVNI: So those need to be registered. Yeah. The key is for that currently sign the root note of the ledger need to be in DNSSEC or something, right? But the ledger needs to carry historical documents and that gets complicated. You'll think all of that through. It was too much for your talk.

For Peter, actually, I have a question if I may unless there are other people ahead of me. No? So, Peter, you said that perhaps some of the operators might publish a CDS RR set and some might not. To me, that sounds like potentially a consistency problem in the zone data. They really should all be able to support it. If somebody denies the existence and somebody doesn't, that feels wrong. I'm not sure that use case needs to be handled.

PETER THOMASSEN: Yeah. I'm not fully clear on what's best here. I wasn't thinking that one operator would publish CDS records and another wouldn't. I was thinking that within one operator, they may be using a diverse set of softwares. One of them may be incompatible with CDS records.

VIKTOR DUKHOVNI: [Permission] people to load the zone because the zone shouldn't—

PETER THOMASSEN: Yes. I think you have a point. I'm not sure how I wrote it in the draft. I'm going to look at that again and that's good feedback. Thank you.

VIKTOR DUKHOVNI: Right. Okay.

JACQUES LATOUR: All right. Sorry.

VIKTOR DUKHOVNI: Actually, if there are people from registries in the room, I was asking for what are your thoughts and comments, not so must questions. Does anybody have anything to say about my crazy ideas about validating a DS before they're published, or small TTLs? Am I ranting or is this something that might happen?

JACQUES LATOUR: I can answer that. From a .ca point of view, I like the idea that when the domain is being registered, that we do all the checks to see if the CDS—on the name server, that are present, on the other side if there's a legitimate registration. It's authentic. Everything works out with CDS to bootstrap that domain right away. So that's what you're meaning, right?

VIKTOR DUKHOVNI: No. I'm talking about an EPP request comes in with DS records and they don't actually match the live zone when you probe it right then when the EPP request happens. And you say, "The DS record change will break the domain. Sorry. I will not publish it until it's corrected."

So the idea is if I, as a .ca registrant, ask my registry to publish a stupid DS record—sorry—and I opted in for validation, perhaps, then you will

say, “This registrant wants validation. You told me earlier that’s what they want. Now you’re giving me bad data. No.” So I would want you to live query the domain during the EPP request before you publish and reject the change. That’s validation. And then, when you do publish it, publish it with a low TTL initially and ramp it up gradually in some number of steps. Maybe just one.

JACQUES LATOUR: I need to validate but I don’t think we will allow to put the DS record that would break the delegation. So we do check.

VIKTOR DUKHOVNI: That’s good. I know .de does that. But many registries don’t. So that’s where the problem is.

JACQUES LATOUR: What about a newly-registered domain that has a CDS. It’s all signed properly. And then, at the registration when it’s created, if there’s a CDS, to automatically upload the DS record.

VIKTOR DUKHOVNI: Again, that’s fine. No problem. Again, I’m concerned about existing domains that want to sign but are very worried about the potential consequences of unexpected situations. So they want to, a, reduce the risk of anything bad happening. And b, if something still happens, be able to back out quickly. That’s really the agenda, is risk reduction and fast rollback. I guess nobody else has comments.

JACQUES LATOUR: I support that. If that would have been … We had a high-profile .ca domain that had a failure and the DS with a one-day TTL caused a lot of pain. That high-profile domain is not signed anymore with DNSSEC because of that. So having this mechanism would have probably helped them definitively reduce the pain.

VIKTOR DUKHOVNI: Okay. Good. So hopefully you'll help me flesh out these ideas, and we'll write some drafts, and maybe get action to actually make some of this into code on various registry backends.

JACQUES LATOUR: That's all the time we have.

KATHY SCHNITT: I do need to make a request, please, that it is ICANN's policy that folks wear their masks indoors. It's been brought to my attention. So please remember to wear your mask. Thank you. I'm sorry, Jacques. We have someone that has a question in the room. Go ahead.

RICK WILHELM: Yeah. I had my hand up in the Zoom Room but it didn't get seen, apparently. No worries. Thanks, Viktor, for the presentation. It looks like that ideas have evolved since we and some others had coffee at IETF.

Regarding the concept of validation, that would be something that would be a bit of a stretch for us to get to because then the registry would be taking on some sort of a proof of both verifying that something was going to work and then also claiming that something didn't'. So that's a different kettle of fish than a metaphorical kettle of fish. There's no actual fish involved. I think it's all vegetarian. That's a different set of scope than a quick rollback, which is something that's a different kind of a discussion.

But just on the concept of validation, within an existing product definition, that would not be something that would be very likely. If it's a different product definition, that'd be something that would be maybe a different discussion. Thank you.

VIKTOR DUKHOVNI: Just one response to that. Speaking of big and small banks, not all of the things I'm looking for need to happen at the same time. So if some things are easier to do sooner and others take longer thought and whatever, that's fine. I'm not sure what a product definition is. I mentioned the idea that there's some sort of opt-in, something maybe resembling registry lock or whatever but not registry lock. But you buy a validated domain rather than a regular domain. I don't know. But I'm looking for creative thoughts as to how to make this work—basically, your input as to what is possible is, in fact, part of the goal of this.

JACQUES LATOUR: Viktor, Mats added a question. It says, "Short TTL requires that the TLD update the zone file with short interval." So there's a relationship between the minimum TTL and—

VIKTOR DUKHOVNI: No. There's no need to update the zone file frequently. The zone file can still serve the same record. I'm not asking for a short RRSIG lifetime. I'm asking for a short TTL. What will happen is that resolvers will query the zone more frequently, initially, because they will expire the DS record from their caches, let's say initially after five minutes, for some number of hours or days before it goes up. And then, when it's later re-signed, it gets a longer TTL and so on. The re-signing time needn't equal the TTL. Even already today … Sorry. Perhaps that's a long enough answer.

JACQUES LATOUR: We're out of time, by the way. So last Peter and then—

PETER THOMASSEN: Yeah. Just one sentence on what Viktor said. I think what he said is generally true. But let's say you are a registry and you have an hourly schedule where you publish your zone file. Then, of course, you can stick to that if you have one- or five-minute DS TTLs. But your prompt rollback requirement requires the zone file, in fact, to be published any time within whatever the rollback period should be. So you would perhaps not need more frequent deployments of the zone file but you definitely would need ad hoc publication.

VIKTOR DUKHOVNI: Right. Yeah. Incremental changes rather than holds on publication or whatever. Yeah.

JACQUES LATOUR: All right. Thank you. That's it. The next session is at 15:00, in half an hour.

KATHY SCHNITT: That's correct, Jacques. Thank you for being up in the middle of the night. We appreciate it. And we'll meet back in here in a half an hour. And it will be the same Zoom link for you folks that are remote. We may stop the recording.

[END OF TRANSCRIPTION]