ICANN69 | Virtual Annual General – DNSSEC and Security Workshop (2 of 3)
Wednesday, October 21, 2020 – 14:30 to 16:00 CEST

KATHY SCHNITT: Hello, and welcome to the DNSSEC and Security Workshop, Part 2. My name is Kathy Schnitt, and I'm here with my fabulous colleague, Kimberly Carlson. We are the remote participation managers for this session.

Please note that this session is being recorded and follows the ICANN expected standards of behavior.

During the question and answers portion of today's call, please feel free to type your questions in the Q&A pod. We will read questions and comments aloud during the time set by our chair or moderator of this session. If you would like to ask your question or your comment verbally, please raise your hand. When called upon, you'll be given permission to unmute your microphone. Kindly unmute your phone and speak at that time.

With that, I'm going to hand the floor over to Steve Crocker. Steve, go ahead.

STEVE CROCKER: Thank you very much. Well, welcome to the panel on DS updates on multi-signer coordination. This will be a very fast-paced set of presentations. Some of the details, if you want to dig into them, are available on the slides. Shumon Huque and I have been working on this in the past, and we will continue to work on it in the future. So this is a continuing series. Let's see. I run the slides for everybody.

The focus is on completing, if you will, the DNSSEC protocol details driven by an understanding that has emerged relatively late in this very long, multi-decade process, [so] there are a couple of gaps in the protocol specs relating to the automation of provisioning issues. One aspect is the automation of DS updates, and the other is, how do you coordinate cross-signing when there are more than one DNS operators involved, each of which has its own keys. This latter is important for two different reasons that actually share the same solution. One is, if you are a large registrant and you want to have the reliability and capacity of having multiple DNS operators serving your zone, all of which are signed but using their own keys, there has to be some coordination across them. We'll get into all those details. The other is that, even if you are much smaller and just using a single DNS operator but you want to move to a different DNS operator, the transition process is essentially exactly the same as running two. You just break it into two steps: bringing the new operator on board and then, in a relatively short time, letting the other one go.

Today's agenda has five parts. The bulk of it is in the three middle parts: a couple of presentations on the status of DNSSEC deployment, and then a series of presentations related to the DS automation process, and then a couple of presentations that go into some depth about the multiple-signer and transfer problem and solutions with that. I will be running the slides for everybody and, as I said, we're going to proceed briskly to stay within our time.

With that—oh, let me take you through just briefly the DS update issue. The issue arises when you have a separate DNS provider who rolls the key and then that creates the need for updating the DS record up in the registry. There's a multiplicity of ways in which that can be done. The arrows on the left (the blue arrows) represent the idea that the DNS operator pushes the new information upward. It could be upward to the registry directly or to the registrar or to some software that's operating on behalf of the registrant. Alternatively, it could be that the registry or the registrar or the registrant's software is polling, looking into the DNS provider's operation, to see when the new keys have been created and pulling it upward from there. We have discovered, after some period of time, that, in the gTLD space, the registries are, I'll say, disinclined, or, from their perspective, not able to participate in this because they're supposed to stay out of the direct interaction with the registrants.

So here's a collection of all of the different possibilities for how to do these things. Then I've listed 1 through 6 and eliminated 1 in the left side there to show all these different options. Very quickly, just pushing through each of these, you can have the DNS provider push it upward. You could have it push it upward to the registrant. You could have the registry poll. The solid line indicates that that actually is in use among several ccTLDs. Or you could have a registry poll the [trial] zone, or you could have registrant's software do the same thing.

With respect to the cross-signing, you have multiple DNS operators. Two are shown here, each of whom is doing key signing and then publication of the zone. One possibility is to have software operating on behalf of the registrant that is doing the coordination across moving keys back and forth. Or you could have, at last notionally, the DNS providers cooperate with each other. In practice, the first is the most likely, and the latter doesn't seem very likely at all, although there's plenty of room for surprises in this area.

With that, I'm going to move right along and turn the floor over to Ulrich Wisser, who will talk about what's happening in the Swedish domain. All yours, Ulrich.

ULRICH WISSER: Hello, everybody. Today I will present the history, present, and future of DNSSEC in the .se zone.

Next slide, please. My name is Ulrich Wisser, and I work for the Swedish Internet Foundation. I have been working on our registry system for many years. Now in the DNS Labs group. I'm Co-Chair of the CENTR-Tech Working Group and member of the DNS-OARC Program Committee.

Next slide, please. So the history of DNSSEC in Sweden starts very early.

Next slide, please. In 2005, we were the first TLD in the world to be signed. At that point, of course, the root wasn't signed, but we had some success with it anyway.

Next slide, please. We signed with RSA-SHA1 because it was the most secure option at the time.

Next slide, please. Fast forward some years. In 2017, we did actually roll to RSA-SHA256. A year after that, we actually rolled the .nu zone to ECDSA.

With that, we come to the present. In the present, we actually have 100% of our registrars supporting to DNSSEC. This has been a process for us, but actually nowadays we have a contractual requirement for our registrars to support DNSSEC.

Next slide, please. Approximately 50% of our domains in .se are signed. We were the first TLD to give incentives to our registrants. So, basically, the renewal for a domain that is DNSSEC-signed is cheaper than an unsigned domain.

Next slide, please. In Sweden, we have a high penetration of validation. More than 90% of all DNS queries in Sweden go through a validating resolver and actually achieved this because we've had, for a long time, a good collection to ISPs/DNS folks in Sweden. We have a group where we collect people that run DNS in Sweden, and we were able to even switch all DNSSEC validation before it was in the root. So they would [have to] import our own trust anchor to this.

Next slide, please. So what is the future of DNSSEC in Sweden?

Next slide, please. We want to go to 100% DNSSEC. If you believe this is impossible, I would have just one question for you. That is, is it reasonable for the DNS to be insecure? I hope that your answer is like mine: no. From that, it just follows that we have to find a way to do it. "How" becomes the question instead. I think today's session is one of the key points in making it work because the automation of nameserver changes [is] a big step to make 100% DNSSEC possible. We at the registry want to work hard in the following years to make this possible.

With that, I'm done. Thank you for your time. If you have questions, I'm happy to answer them, now or later.

STEVE CROCKER: Later, I think, but let me just offer a note of congratulations. You guys have been leaders from the earliest days, and the results are exemplary. So this is very uplifting. Thank you.

The next speaker is Han Zhang from Salesforce.

HAN ZHANG: Thank you, Steve. Hello, everybody. My name is Han Zhang from Salesforce, and today I'm going to briefly talk about deploying DNSSEC in a large enterprise.

Next slide, please. [Which covers—] it's an introduction and hybrid DNSSEC architecture and also some takeaways.

Next slide, please. Actually, I'm going to skip this one because the audience has a lot of knowledge of the DNSSEC. But I do want to cover some characteristics of our enterprise DNS service. At Salesforce, we use third-party DNS vendors to get a sufficient authoritative footprint. We use multiple DNS providers for resilience. So, if one of them is down, we can still survive. Some of our zones are very dynamic. That means we have about one million changes per day, and these changes cause some update propagation delays. Some our customer-facing zones are very large and have millions of records even before signing.

Next slide, please.

Next slide, please. Because we have multiple DNS providers, when we deploy DNSSEC, we need to deploy DNSSEC on these multiple DNS providers. The ideal model to do this is to use a multi-signer DNSSEC model, introduced by Shumon. When we deployed DNSSEC at that time, not enough DNS providers supported these models. So we had to deploy the following two DNSSEC models. The reason why we deployed two models is … So we have many zones at Salesforce, and they have different characteristics. So we categorized them into two groups based on their sizes, how critical these zones are to our business, and how dynamic the zones are.

Next slide, please. The first DNSSEC model we deployed is the … Can we go back one slide? Yeah. Thanks. So the first one we applied is the hidden signing primary model. So the left side is the application code—the zone update originators—which sign the DNS updates to the REST API service. So, in the middle, our team built a REST API service and also the hidden primary nameservers. From there, we sign the zones, and then we do the zone transfer of the signed copy of those to the different DNS providers.

Next slide, please. The second DNSSEC model we applied is a third-party singing vendor model. In this case, the zone update originators still sent DNS updates to a single DNS vendor's REST API service. Then it sends the DNS update to two different infrastructures. These two different structures do the online signing. Because they're using different infrastructures, we still consider them as two separate DNS vendors. So we still have the resilience.

Next slide, please. Because we have many zones and we have to use multiple DNS providers, we had to do some zone migrations from one DNS vendor to another.

Next slide, please. And this sounds very simple because we only need to do some code change to let the application talk to the new vendor's REST API service. Then, at the same time, we can do the NS change for the zone.

Next slide, please. But, if we do this simple zone migration, it would be a disaster. It would break both the provisioning and also the resolution. The reason is that the code change to talk to the new DNS provider's REST API service and the NS change cannot happen at exactly the same time because of the [code release cadency] and also the TTL for the NS records. So our solution is to do a multiple [multiple-step] migration. With this, we were able to migrate zones without any downtime or impact to our customers.

Next slide, please. One thing—yeah, next slide, please. Thank you. So one thing we observed from the zone migration is the delegation between the parent and child zone. For example, if both the parent and child zone are on the same DNS provider, and only the child zone is migrated, then the old provider still serves the child zone, even after we change the NS record for the parent zone. Our solution is to migrate both the parent zone and also the child zone at the same time because it introduces no downtime and there's no impact on provisioning. The shortcoming is just in case we have to roll back the change, we have to change NS records for both the parent zone and also the child zone.

Next slide, please. I'm going to share some takeaways. There are some challenges and even surprises when we deployed DNSSEC, but this can be done with preparation, migrations, monitoring, and cleanup, on top of the DNSSEC-specific task. We also have some DNS camel's burden from old standards. For example, the delegation of child zone and parent zone. But this one is different from the burden from the new standards.

Last—next slide, please—this is teamwork, so everybody on our team is also for this talk. We would love to thank our providers, engineers, and also the product managers.

That concludes my talk. Thank you very much.

STEVE CROCKER: Thank you. It's pretty interesting to see what the action is for a large customer, like Salesforce, wrestling with how to use these facilities. So I think that's very instructive. Thank you very much.

HAN ZHANG: Thank you.

STEVE CROCKER: We now move on to the DNS automation portion of the session. We start with Shumon, who's also from Salesforce. So this is looking ahead at some future things. Take it away, Shumon.

SHUMON HUQUE: Thank you, Steve. It's good to be here.

Next slide, please. I wanted to start this section of the panel by reviewing what we think is the state of DS automation, specifically CDS and CDNSKEY support in the industry today. The high-level summary is that a small set of ccTLDs have adopted it. By contrast, there's no detectable movement towards adoption in the generic TLD space so far.

Can we move to the next slide? So let's review support in the ccTLDs. For .ca, CIRA (the Canadian Internet Registration Authority) spoke about their plans and support a while back at ICANN54 in 2015. I have listed here, next, .se in Sweden. I was under the impression that Sweden has done some work with CDS and CDNSKEY, but I didn't see Ulrich mention that in his slides, so maybe he can correct afterwards or later on in this panel. However, on .za (South Africa), we'll be hearing from Mark Elkins a little bit later about their use of CDS and CDNSKEY. If you attended the previous ICANN workshop, there were presentations by CZ.NIC regarding .cz, of course—that's the Czech Republic—and by SWITCH regarding the use of CDS/CDNSKEY in .ch, Switzerland, and .li (Lichtenstein). So I left some summary technical details on this slide, just for reference, but if you want a more detailed treatment, I'd suggest looking to the archived video and full presentation slides for those. It's possible that there are ccTLDs besides those mentioned on these slides. If I've missed you, please get in touch with Steve Crocker and myself, and we'll make sure you're mentioned next time.

Next slide, please. Registrar support. This is a little bit of a new area where not much has happened to date, but you will be hearing a very interesting presentation, I think, from Brian Dickson of GoDaddy shortly about plans that they have. So I'll just leave it at that.

Let's move on to the next slide. DNS software support. Almost all major open source DNS implementation support publication of CDS and CDNSKEY records and have done so for a while. Where more work is likely needed is tools and machinery to automate DS publication based on these records [and books] into registry/registrar APIs and systems to the extent that's available and possible, etc.

Next slide. What about support in commercial managed DNS providers? Well, to date, there unfortunately isn't much support, but we have started speaking with several of the big players. What we can report is that several of them are actually interested and receptive, but prioritizing implementation likely depends on hearing more customer demand. I'll mention that Neustar and NS1 are both open to supporting this—and GoDaddy, too, as you'll find out soon from Brian. And we're continuing to talk to others.

I think that's basically all I had, so, Steve, why don't we move on directly to our next speaker?

STEVE CROCKER: Thank you very much. Excellent. Jim? Jim Galvin from Afilias.

JIM GALVIN: Thanks very much, Steve.

Move to the next slide, please. As Shumon so eloquently recapped for us here, there's been a lot of discussion about the availability of technology to do DS automation. It's been around for a while, and there certainly has been some success. So I've been talking about what the issues with DS automation are on the gTLD side for some time—quite a number of years. I've even had presentations in the past here in the DNSSEC Workshop. I'm going to focus here on two primary considerations that are most applicable in the gTLD space and factor into why there hasn't been as much broad deployment as one might like in this space. I think there are two real issues. The first one is one that one shares quite normally with security. You have this catch-22. Nobody really wants to make the investment in doing something [security-]wise and add that because a lack of a market for it. Nobody asks for it. DNSSEC is not special in this case. It's been around for more than 25 years, and yet there is just this bump, this wall almost, in getting it done in investment needed. But I want to point specifically, because I think there's a real gap of authority here in the gTLD space … I think this is a primary problem.

Next slide, please. I'm going to point out two things in particular about what actually exists. In the gTLD space, registries and registrars are quite driven by what's in their contracts. Those, of course, are borne out of consensus policies. So you'll see here on this slide what exists with respect to registries. Registry operators are obviously required to find their gTLD zone file, and they are actually required to be able to take in public key information in a secure manner according to industry best practices, which is the key phrase there: "secure according to industry best practices." What exactly is that?

Next slide, please. You'll see, on the registrar slide, that what's interesting is that they are required to allow customers to use DNSSEC upon request in order to move the DNSSEC key information around. But, again, it's only according to industry best practices. So what exactly does that mean, and where are we?

Moving on down to the next slide—this is my last slide here in this space—I think that one of the key things to point out here is that DNS service providers in particular do not have a defined role in the DNS registration ecosystem. Registries' industry practice is just EPP. That's what's done and that's what's accepted and that's what's defined. In the case of registrars, the fact that they support EPP and they can make those transactions does not mean that they necessarily have to make it available to customers, just that they have to be able to do that, even it's manual, and done via SneakerNet. In fact, you can call registrars up and ask them to do this. You can go through that whole SneakerNet kind of process. It's also worth pointing out again that DNS service providers just don't have a role here. Even registrars are not required to provide DNS services. They're only required to move DNSKEY information around. So technology is not the problem here.

We've seen that from the past and you're going to hear more about DS automation going forward. There really are some issues in what people feel like they are allowed to do or have to do. I think it's important to keep that in mind as we're evaluating these issues.

Thanks.

STEVE CROCKER: Thank you very much. Now we segue to Brian Dickson from GoDaddy. You've heard him mentioned in previous talks. Brian, take it away.

BRIAN DICKSON: Okay. So we support DNSSEC/DS currently.

Next slide. What we currently have is for our managed DNS customers is that we do that automatically for DNSSEC customers. So we manage all the keys. We submit changes when the key-signing key rolls through EPP to the registry. What we have decided to do is to extend that out through doing polling for our registrar customers who are not using us for managed DNS. That is the Scenario 3 here.

Go to the next slide. It's basically third-party DNS. What we're planning on doing is still requiring the initial DS setup, which lets us take advantage of the existing processes and procedures for authenticating the registrant. So it maintains the same level of security. That's to get them locked into the ecosystem for authorizing and to initiate polling, so we know it's them that's doing it and that it's all set up properly. Once that's done, [we] poll the CDS and CDNSKEY periodically and send the updates to the registry. So we don't have an ETA but we see this as being the most expeditious and scalable solution or getting around all of these non-technical hurdles. So it may be a technical problem, but we see this as being a simple-to-implement technical solution. Basically, if we can do the polling scale-ably and send the updates in, and that doesn't require any changes by the registry and it avoids having to work on any kind of APIs, it just works.

I'm not sure there's any questions about this. I think its pretty straightforward. And we looked at other alternatives and decided that this was the simplest possible solution. It's actually going to be very simple for us to implement.

STEVE CROCKER: I'm going to use my privilege to just engage a tiny bit here, but we'll do questions mostly at the end. In the prior interactions you and I had about this, you said this is definite; you're going to go do this.

BRIAN DICKSON: Yes.

STEVE CROCKER: You didn't say what the date is. So, as I said at the beginning, this panel is a continuing operation, so consider yourself invited and expected to show up at the next panel. Maybe there will be announcements before then, but certainly, by then, we'll want to hear more about this at the next session.

BRIAN DICKSON: Absolutely. I look forward to it.

STEVE CROCKER: Great. All right, moving right along, Mark Elkins from South Africa is talking about gathering the children's DS records.

MARK ELKINS: Good afternoon, good evening, good morning, everyone. I presume you can hear me, yes?

STEVE CROCKER: Yes.

MARK ELKINS: So I'm also a member of the committee that helps with the DNSSEC meetings at ICANN. Our host was suggesting that maybe this would be a good idea: to go and look at CDS records.

If we move on to the next slide, please. My theory was that, as I'm running or managing the edu.za zone … Just a bit of context around that. It's closed. It's for the non-university-type educational people of after-school age. It's been DNSSEC-signed. It's also IDN-enabled as well. It's a bit of a play toy. There's 150 domains that are active in there as well. Of those, eight are signed. The whole system is totally web-based, including the WHOIS system.

Next slide. So, like I said, I'm into DNS. I'm into DNSSEC. So registrars and registrants can send updates by the web interface, but it needs a manual intervention. They can also update information by using a TXT record in their zone. But I was looking for something perhaps a little bit more simple, such as looking for CDS records.

Next slide, please. Now registrants/registrars can simply sign their domain and include obviously the correct CDS records in their zone, which means, now with a script that has been written just to do this, newly signed domains … One waits for actually four days before including that information into the zone, whereas existing signed domains, which are just looking for CDS or DS updates, happen periodically from a day-to-day basis.

The next slide, please. So the theory is fine.

We can move to the next slide. The theory is fine. PHP script written over a period of about three days at 3:00 A.M. It goes to use the dig command. I was using a system call for dig at that time, and it looks for everything. And it works. And it worked quite nicely. Then I thought that the CDS Quad0 would be really simple to do. I never realized initially that the domain still has to be signed rather than just dropping that record in, but never mind. We learn lessons as we go. So a Quad0 CDS will remove all DS records if there are any, or it simply won't add anything if there is. Otherwise, essentially, a CDS in a child zone will simply mirror a DS record in the whole of the edu.za zone. New CDS records are recorded and, if no changes are made after three days, they're deemed to be valid, which is how a new signing record would get in. Those seemed very, very simple. Then I changed to using NET_DNS2. Things become a little bit more interesting.

Can we have the next slide, please? So, like I said, I initially used dig, and then I moved to NET_DNS2. So, dig by itself—this is an example of one particular error. One of the nameservers for a particular name didn't work. This is where I realized that doing this wasn't just going to be bringing CDS records and converting them into DS records, but there's also an opportunity for other things as well. For example, with using NET_DNS2, you have to supply IP addresses for the lookup. So now you're looking at nameservers, checking to see if they work. And the script now eventually does things like unpublishing unworking domains after ten days. There's another script that looks for domains that are bad, and then readds them and things. So the process did get a bit more complicated.

If we can go on to the next slide, there's some very, very brief logs. The log just simply shows how many domains are in the zone, when something was started, and all the domains that are currently signed—mostly my own domains. Then the two highlighted ones in the middle was me adding nic.edu.za and then quickly transferring it over to Cloudflare and then telling Cloudflare to automate it to sign it. Automatically, what would happen is, after a short time, the status for nic.edu.za changed to "I've seen you for four times in succession, so I'm signing you." That's what that was all about. 150 domains takes about 93 seconds or 90 seconds. If I'm going to go a little bit more, where would I go from here? Well, this seems to work and it works quite nicely. It's scalable. You just have multiple versions of the same software looking for different ranges of domain names. I'm also a registrar—not an ICANN registrar but a registrar for the local co.za system. So I could use this as a [mythology] for examining and extracting CDS records also from my clients if they run their own DNS.

Next slide. Thank you very much for your attention.

STEVE CROCKER: Thank you very much. Excellent work. Dan York is now going to address the question of, how do we keep track of the progress?

DAN YORK: Hi, everyone.

Next slide. So—go on to the next slide, too. Some of you may be aware we've been generating these maps for a good number of years now—a long time. Steve, actually, and his Shinkuro were the ones that originated these maps as part of the DNSSEC Deployment Initiative[. In] maintaining those for many years, we in the Internet Society took that on in 2014 and continued to generate these maps every Monday out of a database that we maintain.

But the challenges you'll see, if you look at the maps—you can go on to the next slide—is that there's a lot of green. If you know the map thing, basically DS in root is the bright green, and then operational is the dark green. Quite honestly, I need to do some little updating to … But a lot of this will be dark green, except for parts of Africa and a few parts of LAC and Asia-Pacific. But a lot of map is there. So it's starting to use its usefulness because, at some point, it's pretty much widely deployed across all the ccTLDs.

Next slide. The current thinking that Steve and I have been discussing is … Right now, there are five deployment states in the map, starting with experimental. We learned somebody is playing around with the DNSSEC, that they've announced, that they've done some things but it's not really there, that there's a DS in root, and that they're finally doing that.

The proposal that we're playing with right now—next slide—is that we would add a sixth state color, not yet decided, or something. But it would basically be the next level once you've gone beyond truly operational. It would then do this. The idea would be that it would allow us then to evolve these maps in a way that we'd see something more in terms of how the state of DNSSEC deployment is happening in the ccTLDs.

Next slide, please. So, right now we're just exploring what the changes are that will need to be made to the database and the code to go and do this. For people who subscribe, there's a mailing list where you can get the maps every Monday. I will be going out to that group, just to get their feedback around this[: are] people fine with that? We're not always sure how people are using all the maps in different ways. Our goal is to begin this work this quarter and see where we can go with that.

So that's it: my quick, short, simple thing. If you got ideas or feedback or things, please ping me at york@isoc.org. Thanks.

STEVE CROCKER: Are you going to run a public process on choosing the color?

DAN YORK: No. [It'd never] [inaudible]. We will choose something that works appropriately. Thank you very much.

STEVE CROCKER: We now move into the next portion of the program, focusing on the coordination of multiple signers and transfers. Eric Osterweil from George Mason University has done some excellent work looking into what the history has been of past transfers. Eric?

ERIC OSTERWEIL: Thanks a lot, Steve. Hey, everyone. I'm Eric Osterweil, professor at George Mason University. This is some joint work with Steve and some students. Basically what we're looking at is something we call, are DNSKEY transitions working?

Next slide, please. I'm going to move through pretty quickly, even though there's a lot of words. So apologies if I go too fast. I'm certainly happy to answer questions. We have a whole bunch of data. This analysis focuses on watching DNSKEYs and DS records. Looking at the guidance that we've had more or less since the beginning, stemming from RFC 5011 through various other RFCs that have emerged to eventually be 7583, etc., looking at how keys are transitioned … In fact, the root key just rolled, which I think we all know. But I think, in this work, what we're doing is we're asking a question: what do key rollovers actually look like, and are they "working"? So in this talk what I'm going to do is I'm going to present a framework that we've developed that actually quantifies and demonstrates this. But first, just being a little pedantic, we've talked about rollovers in the community for well over a decade. I just want to revisit that real quickly to hang the rest of this talk on.

What exactly is a rollover? Is a rollover when one key that you're using goes to the next new key that you're using, which kind of makes sense? So a one-to-one transition? If that's the case, then what happens when your zone as n keys and, after a transition, there's m keys, where m and n are not one? Did all the disappearing keys roll over to each or all of the remaining keys? If some of the keys that are still there aren't being used to sign data, did they get rolled over to or not get rolled over to? So that's what we coined the term "key transition" about. So our perspective is that a key transition is a superset. It says the keys in your zone are changing and maybe a degenerate case of a key transition is a one-to-one key rollover.

Next slide, please. Like that the movie is not just a summary of a bunch of a snapshots, when we poll things, when we measure, when we monitor, we get a snapshot of things, you have to stitch it together into something more continuous. So we came up with a methodology that I don't have time to get into but I'm happy to chat about at length that we call bridging, busting, and binding. The long and short of it is, when we take a look at a zone, when we monitor it, when we poll it, we get a key, we stitch that into a continuous lifecycle. So what I have here is a sideways candlestick that shows the very first inception of a key was seen and when we actually first saw it. You can imagine all the little pieces of monitoring that we saw a long the way. It gets stitched together into "Here's the key's lifetime." Then we overlay that with when we saw the key actually getting used. So that let's us basically say, "Well, here's the timescale of one key, and then, later on, another key showed up," and we can stack them side by side. We can look at timing gaps and say, "What's the gap between when the first key showed up and the second key?" and a whole bunch of other things.

Next slide, please. So all of that is to say, "Here's some views of some actual key transitions in the wild over the course of years." Apologies—the text is so small you can't see the dates. But these are what we could call orderly transitions. You can see just inherently a lot of periodicity. In the first case, you can see that keys were provisioned and deployed before they actually got used, which is good hygiene in a lot of ways. The upper graph is arin.net. The lower graph is .com. You can see they behave a little differently, but nevertheless you can see a lot of order in their transitions. The big long bars are KSKs and the bright color, and the little ones are ZSKs.

Next slide, please. And here's the DNS root. You can see the very earliest key transitions that the DNS root did. You can see some little red bars. Those are just where the revocation was set. So you can see we now have a visual representation of what key transitions look like historically based on [inaudible] data.

Next slide. You can also see that not every zone follows the exact same patter. The upper one is [inaudible]hotels.se. You can see, at one point, there were, I believe, 40 or 41 keys that were still valid according to their signatures that basically would be allowed to be used to sign data, which is quite different. Down below is an ARPA zone, where you can see they have a bunch of keys that are in active use at the same time. There's maybe nothing exactly wrong with that. It's just a little bit different than what we've seen in other places. There's a whole bunch of other anomalies that you can see by actually plotting these out and looking at them.

Next slide, please. So why I am going into pictures? Well, pictures are really cool and I love pictures and blah, blah, blah. But what we've actually done is we've actually looked at the inter-gap timing of all these things: DS records, DNSKEYs. And we've come up with a topography of what's really important to measure in order to quantify these suckers. So what we've proposed is a DNSKEY transition anatomy. What are the time gaps that you should measure in order to say something meaningful about key transitions. And what does "meaningful" mean? Well, we took, as a starting point, RFC guidance and we said, "Well, RFCs say you should do these various things, which means we can overlay that into our anatomy and say, "Well, then did it work? Are you doing something different? If you're doing something different, is it good? Is it bad? It is maybe a clever inspiration that isn't codified in RFC? Or is there a problem there?" So down below I have a summary table that says, how did we use this anatomy to codify a process definition of various guidance from RFCs?

Next slide, please. Just a real quick aside. All this data is taken from secspider. It's been running since 2005, collecting data, etc. The graphs I showed you are a snapshot we've started analyzing from the first ten years, so it's continuing to go since then. But that corpus that we used for these analyses and these graphs was about 3.45 billion rows, and we have about half-a-million zones that had key transitions that we measured in there. It was over two million keys that it wound up being. But secspider still runs today. We've got now closer to 30.1 billion rows and 7.7 million DNSSEC-enabled zones that we're tracking. Actually, there's about two million in the hopper. So it's growing, growing, growing.

Next slide, please. I really wish we had time to get into some cool results. We have tons of them. But I'll just tease you all with a little bit. So here's some graphs of, again, just the first ten years. What we did is we looked at ZSKs and we said, "Okay, well how many of the ZSKs are following guidance." What we can see is that a lot of the key transitions, as specified in the RFCs, are doing something a little bit different than the RFCs say. So, okay. Then, we look at KSKs, where there's a lot more that has to go into a transition but then again there's a lot more models they could follow, again we see a lot of heterogeneity. What's interesting is that, before the chain of trust really started to form before the hierarchy was really there, obviously everyone was doing a lot of 5011. But then, as the hierarchy started to form around 2008, you started to see a lot more experimentation, a lot more different models of key transitions.

Next slide, please. One high-order bit was that, of those that did employ key transitions methodology from the RFCs, we saw a lot of warnings and errors. We were really liberal about that we don't want to call something an error unless we think it really affects something. Mostly we'll call deviations from the specification a warning unless there's a reason that it should be definitely not done a certain way. And we see there's a lot of interesting behavior that is maybe non-standard. So the correctness may not actually be there. But that doesn't mean there's a problem. It could mean that there's an interesting optimization there that works just fine.

Next slide. I know I'm a little bit over, so I'll wrap up with one more slide. There is a ton more data, and I'd love to chat with anybody who's interested about that. We have a tech report that should be released on archive pretty soon. The title is there. But basically what we've done is we said, "Hey, look, just like when deer walk through a path and they eventually wear a trail out (they call those desire lines), what are the desire lines of key transitions?" So we basically [disparatize] all of our measurements so that we can actually plot this out in a topography and say, what are people doing? Sometimes is it good? Sometimes is it bad? Can we learn something good or bad, etc.? What we're planning to do with this—our future work—is to basically build a real-time analysis engine that can go off and start looking at key transitions as they're happening, report out operationally. It looks like it's going well from a validator's perspective. A lot of this is only half the picture because we're looking at the authoritative data, but we also want to start adding in, "Hey, validating resolvers are having the following experience tracking and using a key transition."

Yeah, I think I'm super over, so I'll stop there. But I look forward to chatting with anybody.

Next slide.

STEVE CROCKER: Thank you. You have to come back up a slide, which I'm just going to power through here to get to Shumon's talk. Shumon, it's your turn now.

SHUMON HUQUE: Great. Thank you again, Steve. So I speak about multi-signer DNSSEC at the last ICANN DNSSEC Workshop. This is a configuration which multiple DNS providers cooperatively sign and serve the same DNS zone.

Next slide, Steve. Today I'll give a status update and talking about continuing developments. RFC 8901, the specification document for multi-signer DNSSEC, has been very recently published. You have a link to it on this slide. To date, NS1 has a preliminary implementation, and Neustar has a work in progress. So we're hoping to invite those guys back to a future workshop to talk about their stuff in more detail.

Next slide, please. I'm going to talk about some testbeds that we've developed. If you've followed this area, you probably know that two multi-signer models have been developed. In Model 1, which I show in this diagram, the zone owner holds a common key-signing key, and each provider has their own zone-signing key. The zone owner uses each provider's API to obtain their respective ZSKs, builds and signs the resulting DNS keyset, pushes the signed DNS set back to each provider. And the DS set in the parent—that's the thing on then top left of this diagram—references the common zone owner KSK.

Next slide, please. So the zone multisigner1.com is a testbed implementation with two providers. One is NS1 and the other is a set of BIND DNS service, operated by me at this time.

Next slide, please. This slide is just for reference in case folks want to poke around with the configuration later. It lists the DNSKEY RRset configuration for this zone.

Next. This slide is [inaudible] for reference. It lists the high-level steps needed to set this configuration up. Now, BIND doesn't naturally support this mode of operation, so you need a few not-too-elaborate hacks to make it work involving manually signing the DNSKEY RRset and stitching it into the zone. But basically it can be done without too much trouble.

The more interesting part might be the NS1 slide, which we can see on the next slide. There are three API functions. These are essentially rest HTTPS API functions which allow you to obtain the current DNSKEY configuration, push a new DNSKEY RRset configuration into the zone, and lastly to push a DNSKEY RRset signature into the zone.

The next slide, please. I ran this configuration through some DNSSEC diagnostic tools like dnsviz, and looks entirely clean, as I expected.

Let's move on to Model 2 on the next slide. Here, each provider has their own KSK and ZSK, and the zone owner's task here is to coordinate the cross-sharing of ZSKs between the providers.

Next slide. Then, finally, the DS set in the parent zone, on the top left of this diagram, needs to reference each provider's KSK. Now, there's an interesting question here about which entity does the DS publication? It can of course be the zone owner, but mechanisms that may allow the DNS operators to play a direct role here are of course very much on topic for this panel, as you might guess.

Next slide. Here again I'm just showing some configuration details if you want to poke around in it later just for reference.

Let's move on to the next slide. This is Provider's A's DNSKEY RRset, and the next slide is Provider B's.

Next slide, please.

And let's move on to the next slide. So that's multisigner2.com. That's the second test bed. Now, some set of details here, again, for reference with BIND. We can actually make this configuration work pretty clearly with its auto DNSSEC mode where the nameserver itself is managing the signed zone. There's a tool called DNSSEC-importkey which allows you to pretty easily set up this configuration.

Next slide, please. Running the model2 zone through dnsviz actually elicits a couple of warnings. So the zone validates fine, but dnsviz does not expect to see a configuration such as this, where you have different sets of servers for the same zone presenting different DNSKEY RRset and doesn't realize this is an intentional configuration and not actually an operator error. So I'm actually speaking with [Casey DC], the author of this tool, on ways to fix that, probably with an optional configuration knob telling dnsviz that, yeah, this is a multi-signer configuration. This should not be treated as an error.

Let's move on to the next slide.

And the next slide. So the use of CDS and CDNSKEY can in theory work fine for multi-signer configurations, too. There are a few subtleties involved in this configuration. For Model 1, the zone owner needs to push these records down to the providers. For Model 2, we need to cross-share these records across the providers in the same way we do for the zone-signing keys. It should work fine. I'll admit that this is not a configuration that I actually tested, but that's one of the points of setting up the testbed. So we're going to try and roll through some examples and make sure we validate that this configuration works fine.

Next slide please, Steve. We hope to expand the testbeds and do more work on them—things like recruiting more implementations and vendors, doing key rollover tests and continuous validate-ability tests, better automation tools. I was also thinking of applying some of Eric's key transitions analysis to the multi-signer configurations, which would be a very interesting research area. So, if anyone is interested in helping out with some of this work, please get in touch with me.

Next slide, please. Open source DNS software probably needs some better support to make these multi-signer configurations work. So some of the implementations that I've looked at partially support one or the other configuration, so I think there's a little bit more work that could be done in this area. We'll be chatting with those guys a little bit more.

Next slide, which I think is my last slide. We plan to look into extending existing multi-provider DNS toolkits that are already out there to support the two areas where they are deficient. One is support for multi-signer operation, and the second one is DS automation.

So I think that's it from me, so, Steve, I will turn it back to you now.

STEVE CROCKER: Thank you very much. We're moving right along. I get to wrap up quickly here. So, taking everything that's been said and distilling out a few action items—let me make it very clear that these are my personal opinions; nobody else has to be charged with being blamed with these—I see a sequence of actions that will help us move forward. One is that I think the multi-signer testbeds are extremely important and will provide some real experience with actually making these protocols work and raise the visibility and attention level to this.

Brian Dickson's announcement that GoDaddy is going forward with supporting scanning for the CDS and CDNSKEY records is, in my view, a gamechanger and opens the door for registrars, particularly in the g space, to make some serious progress here. I think that's a very, very important development. We'll be tracking that closely.

DNS providers—I'm not sure what the preferred term is; sometimes I say third-party DNS providers, and people use other terminology … In general, there has to be interfaces so that ZSKs and KSKs can be injected into their systems to support both transfers and the multi-signer activity.

A political issue is whether or not third-party DNS operators should become recognized within the ICANN ecosystem. They are not at the moment, so they are invisible.

There's several options that could become … For example, a new stakeholder group within the Contracted Parties House. I suspect that that is not what anybody desires because that implies contractual relationships, and I suspect that's more than is necessary. They could become active within the ISP constituency, or they could become a new constituency within the Non-Contracted Party House's Commercial Stakeholder Group. I'm sure there's other possibilities that I haven't listed there.

DNS toolkit vendors have a role to play here in terms of adding interfaces for moving the keys around. This interfaces with the DNS providers.

Then, as Dan York mentioned, perhaps it's a good time to begin keeping track of support within the TLDs for automation of the update of the DS records. I think we should begin to see some interesting statistics there, although, since one of the ways to implement DNS automation is through the registrars, it may not be just as simple as adding a new state to the TLD records.

So that's my quick summary of recommended actions. As I've said a couple of times along the way, this is a continuing series, and what you've just seen on the previous slide is the beginnings of the agenda for the next meeting/first ICANN meeting of next year.

With that, do we have any time? We do. We have two minutes, according to my watch, for general question and answer. Happy to engage with anybody. I have not been able to see the Q&A pod because I think I'm obscuring it, but I will try to kill the sharing. That's—how do I do that?

KATHY SCHNITT: Click Stop Share.

STEVE CROCKER: Yeah, but I don't have the button on my … I'll do it this way. All right. Now I can see the Q&A pod. Well, I hope other people have been watching this.

Let me just ask for anybody who verbally wants to ask questions. So a lot of the questions are being asked and answered by others, which is very good.

Jaromir says it should include Costa Rica and Slovakia. I think that the slides will get updated and, as we get the census added, this will come out in the wash. It's one of the reasons to move forward with adding a new state to the maps.

[Shiva] asked, "Perhaps ICANN could also interact with open source proprietary or free groups with document lists attached to identify existing open source DNS or DNSSEC software [gaps where fixes would] be needed." Well, that is, in essence, what we're doing here at a pretty high level. How to engage all of the vendors is a more open question.

Anything else? Anybody else want to raise any questions or speak?

Oh, Vittorio says in the chat that he looked at ISP constituent charter a few years ago to see if a DNS vendor or developer could join but it didn't look so. Interesting.

Well, I don't see any further action. I don't hear anything further. And we have now reached the bottom of the hour. So, with that, Kathy, I turn the floor back over you, and we move on to the next portion. Everybody, particularly all the panelists on this very fast-paced panel, thank you very much for your excellent presentations and your very, very smooth cooperation. Talk to you all next year.

KATHY SCHNITT: Thank you, Steve. That's fabulous. Are you able to stop sharing now?

STEVE CROCKER: I thought I had implicitly done that.

KATHY SCHNITT: Oh, you did, and I think … Matthijs, were you able to pull up? Is that your screen? Yes.

MATTHIJS MEKKING: This is my screen.

KATHY SCHNITT: Beautiful. All right, it's all on you.

MATTHIJS MEKKING: It cost some time to get the screenshare working, but I finally did it. Can you still hear me?

STEVE CROCKER: Yes.

KATHY SCHNITT: Yes, we can hear you.

MATTHIJS MEKKING: All right, great. Thanks for having me. I'm going to talk a little bit about the new key-signing policy in BIND 9.16. ISC has this as a current stable version. It's eight months now. It will be the next [ESV] after 9.11. One of the significant changes here is this: the introduction of the new policy with the goal of making DNSSEC-signing even simpler than it was before.

But first let's look at how it was before. In 9.11/9.14, we had automated DNSSEC-signing. It was not hard but had some peculiarities. The configuration was quite simple: just turn on auto-DNSSEC-maintain. But then you had to know if that zone has to be dynamic or we have to accept this other option, like inline-sign. Also keys you had to create in advance, so there was [no] generation. And the keys that you could generate had some meta data. That meta data: you set when the key needs to be published, when it needs to be used for signing, when you need to remove it from the zone again. This meta data can be set up with another tool like DNSSEC-settime. All this can be managed with another tool, DNSSEC-keymgr, which can you set some sort of policy already on and manage your rollovers. So these configuration parameters in DNSSEC-policy.conf that the DNSSEC-keymgr reads, together with some options in named.conf actually determine meta data values and then the rollovers. Okay, actually that sounded quite complicated.

In 9.16, this is the configuration. And we got rid of all the peculiarities. You put this in your options configuration and will sign with the default DNSSEC-policy. If you don't care about the [data] for DNSSEC, then this is all you need to know.

Pablo mentioned earlier that one of the barriers to using DNSSEC is that operators are afraid to make a misconfiguration and that then has some impact. So I hope, with this configuration, it makes it easier for them. At least my opinion is that it makes it more intuitive to use. And there's more automation going on. So there's less stuff that you can actually touch and make it go wrong.

I also think that this new approach makes it easier in the multi-signer model or any other complex signing system because you have an easier reference to a policy—[how am I] going to sign the zone—and you can use the same policy on the different side. Also, if you're using more or different software vendors, there's other vendors that use this policy model. [inaudible] DNS uses this. Open DNSSEC uses this. So it makes it easier to actually have multi-diverse software. The new model is more robust because it no longer relies on that meta data from the key files perhaps. Instead it relies on the timing state machine. The state machine is used to determine when the keys needs to be published, which keys need to be used for signing, it's mathematical model actually [to prevent yourself from going rogue]. Its interesting that Eric mentioned that rollovers are sometimes not done as set in the RFCs or in the standards. Basically, using a state machine can actually give a little bit of different behavior in rollovers. I've seen that. But still the state machine is there to make sure that the correctness is there.

We use this as the default policy: a single CSK, ECDSA algorithm, NSEC. All these things are maybe perhaps not traditional, but we do think that this is the most [sane] thing for people that don't want to look into the details. There's no key rollover, so you don't have to worry about key ceremonies. All these other parameters will ensure shorter, smaller responses.

If you want, you can easily set up your own policy. I have a few examples here. I have a cats policy for my kitten example, which I'm going to use for my demo, if that's going to work. For example, if you do want to have the traditional split, this is how you would set it up: a key [inaudible] configuration [clause] with has a KSK line. One line represents one key you're going to use in your zone. So we're going to use one KSK here and one ZSK with different lifetimes, both RSA-SHA256 algorithm. This is just an example.

Here I have some more examples, just to show that you can set different timings in different formats. You can use the [ISO 8601] format or the BIND TTL format or just plain seconds. Not yet supported is actually NSEC3, but it will be released in the next release. You would most likely configure it [like this].

So, previously, we had all these options to turn the knobs on how you would sign your DNSSEC. All of these options will be obsolete eventually because they will become redundant with the new policy or they no longer make sense or at least everything you were able to [tune] with the new DNSSEC-policy. So it will take a while, but these options will go away sometime.

What else can you expect? I said that this new policy is able to generate keys for you. It does that on a just-in-time basis. You can actually trade keys in advance if you want to and [name data daemon] will be able to pick that up. Otherwise, it will just create the keys when they need to be pre-published.

It does CDS/CDSKEY. That was actually already in 9.11, so that's nothing new here. It works on multiple zones, so it no longer matters if you're a dynamic or a static zone. Just setting the DNSSEC-policy it'll work. It also works on secondary zones, and then it acts as a bump-in-the-wire signer.

What else? Algorithm rollover. To us, this is just a policy reconfiguration changing your algorithm in that DNS policy statement and run an RNDC [inaudible] reconfig. I hope making this easier will help in [inaudible] where we can see a faster transition to a newer algorithm. We have some RDNC commands that you can see the status of the DNSKEY manager.

I think it would be best if I am able to do a demo to actually show how easy it can be. So now I'm going to share a different screen.

Okay. I have a couple of windows here. I hope that is visible.

KATHY SCHNITT: It is. We can see it.

MATTHIJS MEKKING: Okay. Cool. Is it readable? Is it large enough?

KATHY SCHNITT: I'm able to make it out.

MATTHIJS MEKKING: Great. So I locally set up just a couple of zones here. This is my configuration. I have actually two lines of DNSSEC configuration here. I want my keys in a specific directory. This would be just a zone I can set up right now. I've left out the configuration for secondaries here, but this is all I need for getting my zone signed but it inherits that default policy from the options statement. This is zone where I'm going to show how things work because this has a different DNSSEC-policy. I'm going to show that it in a bit, but basically it is a policy which faster times so we can actually see rollover happen within a few minutes. I've just one more zone, but nothing with DNSSEC. You can override it if you have zones that you don't want secure. I don't know why, but it's an option.

So what else do we have here? I'm going to, in that same window that I showed you here, watch the output of the DNSSEC status with RNDC on that zone that we're going to sign. It says "connection refused" because we haven't actually started [inaudible] yet. Then, in these two windows, I'm going to show the dig output for the CDS records when that is actually going to be introduced in the zone. This is for the DNSSEC key.

So let's start Name B. We can actually see two keys being created. Here are the key files for Siamese and kittens. We'll see that everything has actually started to be published. There are in rumored or omnipresent state. Those are two different states for things to be published in the zone. We can see that the DS is still in the hidden stage, and the CDS record here has not been published yet because first those other things have to be in the zone long enough so that that validators know about it, and this is the only present state.

Let's see. Now, I don't actually have a parent here, so I'm going to mimic that we have seen the DS record in the parent in just a bit.

There we go. The DS moved to the rumored state. That means that the CDS record can be published, which is happening right here. So it is available for polling for a script to be run. So we can actually say, "Push it to the parent." Normally, you have to wait a little bit before that is in the parent. But, for this demo purpose, we'll do as it is instant. So I'm going to say this is published. It's marked as published. Now I have to tickle BIND a little bit to say, "Load the keys and run another key manager run." Because of the short time, this soon will go into a omnipresent. If that happens, we're actually done with the signing.

So let's try a rollover. You can see that the next rollover is scheduled next year for the KSK. This one is set to 30 days, as derived from the policy. But, if you want to do it faster because of some event, you can run a rollover command. This can all be scripted, of course. So a rollover is scheduled. I've tickled BIND again. We'll see that there's more keys. There's another KSK here. You can also see that in the status. Basically it publishes everything except for DS, again, because these things have to be long enough in the zone so that validators know about. That happened right now. So this is in rumored state, and this is in unretentive state. At this point, we all have to interact with the parent again, and then, after that has happened, we'll say, "Okay. This key has been withdrawn and another key has been published.

Yes, actually, I have a little more to tell about it. I can actually say that, if you paid attention here, that [DS—CDS] record has changed. We've seen things happening here, that the old key has been removed right now.

The only thing I want to show you right now is that an algorithm rollover is just a reconfig. I had a different configuration called Cheshire, which is another cat. That basically has a different algorithm. We're going to move to [ZSK] with lifetime unlimited so we can use it forever, basically, in a different algorithm.

So there's a lot of things happening here. There's three keys with two different algorithms. The CDS remains the same for a while because the new algorithm has not been in the zone long enough. You can see the new algorithm here. What I can show you eventually is that we have to run the check DS command on both keys again to make sure that the rollover continues. We can do that not yet because the is in hidden. We have to wait for this CDS to change. But I think you would get the gist of it.

So the new DNSSEC-policy, in my opinion, is 90% ready, but we still have some improvements on the way. In the shorter term, as I mentioned, we want to add the NSEC3 and we want to make check DS RDNC command actually internal so you don't actually have to run those commands. BIND is capable of doing queries and can actually check the parent nameserver itself to see if the DS has been published or if it has been withdrawn. We also want to derive the TTL from zones, so it makes configuration shorter. For example, the max zone TTL is something you have to configure right now but can also easily be derived from the zone.

In the longer term, we still have to add 5011, offline KSK, which might help in the multi-signing model even more and things like standby keys.

But, overall, this DNSSEC-policy, I think, is the foundation that makes it easier to implement all these things but also to maintain [inaudible].

Going back to the demo for a bit. Yes, the DS has been rumored right now, so we can [withdrawn] and we can publish this one. We can actually see that this new CDS in the zone is this one. And things are happening right now. We have done an algorithm rollover.

And that's it from me, actually. I don't know if there are questions.

KATHY SCHNITT: Again, if anyone has a question, you can please type it into the Q&A pod. And since we still have a few more moments, if you want to ask a question verbally, please raise your hand. We've had one up for a while, but I think we lost it. Viktor, please go ahead.

VIKTOR DUKHOVNI: Can you hear me now?

KATHY SCHNITT: You're still faint, Viktor.

VIKTOR DUKHOVNI: [This is as good as] I can do. Is that better?

KATHY SCHNITT: I can barely hear you. Can anyone else hear him?

MATTHIJS MEKKING: Barely. We can try.

VIKTOR DUKHOVNI: Okay, I'll try. So you showed a setting expiration time in [zone] keys. One of the things that I've always been worried about is that a key will expire from the zone before really any suitable replacement is visible in terms of DS in the parent. Is there any kind of safety margin there to prevent keys from [going] away too early?

MATTHIJS MEKKING: Yes. The lifetime or the expiration time in that DNSSEC-policy is actually a way to do key rollovers. So, if I set a lifetime of one here, BIND will make sure that a new key is pre-publish. It will generate a key and pre-publish it well in advance before the other key is retired. That means—

VIKTOR DUKHOVNI: But what about KSK and if there's no DS in the parent?

MATTHIJS MEKKING: There is this thing indeed that we still haven't completely automated, which is the publishing of the DS in the parent. Now, if you don't issue that check-DS command, or, if later, internally, BIND will check for the DS and it doesn't see it, it will stop the rollover. So the timing that is used in DS status is for—how do you say it?—an estimation. But if you forget about it and don't publish the DS, then it will stay in the correct state at least.

Does that make sense?

VIKTOR DUKHOVNI: Okay, so it is safe now, right?

MATTHIJS MEKKING: It is safe.

VIKTOR DUKHOVNI: Once you remove your active key.

MATTHIJS MEKKING: Yes. Previously, with the meta data, it was not safe because it relied on the operator knowing to set the right meta data.

VIKTOR DUKHOVNI: This is important. That's very good. The last thing I ask: do you have a tutorial URL that walks people through how to do all of this?

MATTHIJS MEKKING: I could make this demo into a tutorial. There's not something right now, but it would be a good thing for maybe … What we have is KB article on ISC. So that's a good suggestion.

KATHY SCHNITT: There's a question in the Q&A pod. It says, "Great work. What's the format of the on-disk data for keys? I read that the meta data key files will be gone, but it would be great to have some human readable way to see what's going on. I can envision some helper scripts to gather stats or external monitoring things." That's from Hugo.

MATTHIJS MEKKING: Thanks for the question. The format will be the same, actually. There will be one extra key file, which ends in .state. So the public key file, .key, and the private key file, .private, will stay. The meta data will also stay in those files. So they're actually not changing. Just know that it's now an estimation of those events rather than the actual time. We'll try to set those meta data to be most precise, but, for example, if something happens [inaudible] that you [inaudible] DS submission later than you anticipated, those times will deviate.

KATHY SCHNITT: [Thanks for that]. I don't see any more question in the pod or hands raised. Is there any other panelist that wants to add anything before we close the session?

We do have another question. This is from Sarah. "Can you talk a little about the migration process from an existing set of [inaudible] 9.14 with manually generated keys to 9.16 with the policy?"

MATTHIJS MEKKING: Yes. So you can create keys in advance. So you can also use existing keys and then do reconfiguration to DNSSEC-policy. If you want to use the same keys, then you may have to make sure that those characteristics match the keys and what's in the policy. If you're going to change to a policy, BIND will use those keys and introduce the new keys in a safe manner. Think of it as an implicit import statement. BIND will be able to detect those keys that do not yet have a state file, will read them, will determine what state they're in, and create a state file for them. At that point, they are the keys that are in use. And, if they match the policy, they will keep being used. If they don't match the policy, they will actually be out-roduced. Or I don't know a better word for that.

KATHY SCHNITT: Thank you. Again, any of our panelists or anything want to make any further comments before we close this session?

Sarah responded, "If you write a tutorial, can you include a section on this?"

MATTHIJS MEKKING: Yes, I will because it's a good question and I think it's useful information.

KATHY SCHNITT: Thank you, Sarah. Thank you for the presentation.

With that said, it looks like we've come to the end of [inaudible]. Part 3 will begin at half-past the hour. We will be in the same Zoom room. So you can disconnect and come back, or you can just hang out and listen to us panelists chitchat. Otherwise, I want to again thank all our panelists, as well as our planning committee. We're going to close this session until Part 3. Please stop the recording.

[END OF TRANSCRIPTION]