ICANN72 | Virtual Annual General Meeting – DNSSEC and Security Workshop (2 of 3)
Wednesday, October 27, 2021 – 14:30 to 16:00 PDT

KATHY SCHNITT: Thank you. Hello and welcome to the DNSSEC and Security Workshop, Part Two of Three. My name is Kathy and I'm joined with my colleague, Andrew. 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. We will take questions and comments as we did for part one and we will post those instructions in the chat pod. And with that, I'm happy to hand the floor over to Steve Crocker.

STEVE CROCKER: Thank you, Kathy, and welcome, everybody. Shumon Huque and I have had the pleasure of running these panels for several sessions. This is the sixth episode and I think we're going to continue for probably a similar number. Our focus is on the automation of DNSSEC provisioning, specifically DS updates and multi-signer coordination. We have five panelists, whom I will introduce momentarily. Let me make sure that the slide mechanism is working. There we are.

So as I said, the focus is on two aspects that really arise from gaps in the original DNSSEC protocols, where there is no clear way to automate some of the changes that need to be made—the automation of DS updates on based on periodic key changes and coordination among multiple DNS signers, where they are independently signing the zone.

And the reason for having multiple DNS providers is two-fold. One is certain users—certain customers—may want the extra capacity and the extra reliability of having multiple DNS providers. But it also is the case that if one wants to switch from one DNS provider to another who is signing the zone and providing it independent service, that in order to make that transition work without a glitch—that means without loss of resolution and without loss of validation—then there's a certain amount of coordination that is necessary.

Coordination can be done manually but it is painful and subject to some human error because there's a lot of details to convey across. So that's been the focus of projects which you're going to hear about and you probably have heard about it in the past. So this is the update along these two dimensions.

Here's the agenda for this session. I will stop talking in a little while and then Jaromír … Oh, here I go. Jaromír [Tarish]—is that close enough? That's okay—talking about recent DNSSEC automation developments in the .CZ environment. Mats Dufberg—I'm in good shape today. Mats Dufberg from the Swedish Internet foundation on CDS and CDNS key verification in Zonemaster. Peter Thomassen from deSEC on authentication bootstrapping of DNSSEC delegations. Pouyan Fotouhi Tehrani from the Free University of Berlin, talking about a DNS Resolver Observatory. And then Ulrich Wisser from The Swedish Internet Foundation on an introduction to CSYNC.

It would be best if we hold questions to the end. Otherwise, I think the time will get out of control. And hopefully. There will be a sufficient

amount of time for your questions. Do feel free to put your questions in the chat.

Let me just briefly talk about DS updates. When you have a DNS provider who is signing the zone, and then there's a necessity to move a DS record up into the registry, the question is how best to do that. If you can get it to the registrar, the registrar will push it up using EPP. If the registrar is the one who's providing the DNS service, then that's easy because they have internal communications. But the cases that we're concerned about are where the DNS provider is external to the registrar.

The lines in red illustrate or designate the idea that the party on the top is pulling the data out of the zone that is created by the DNS provider. The blue lines suggest that the DNS provider is pushing the data upward. The fact that three of those four arrows are dotted means that those are theoretically possible but not evident in any known implementations that we've seen so far.

The solid line represents polling by the registry, directly to the registrant's DNS zone and pulling up CDS or CDNS keys. We have now—if you listened to Dan York's opening talk—now, on the maps, show those ccTLDs which do have DS automation.

And we are also hearing potential reports of some implementations that are underway but not yet complete, where the registrar might do the polling. In the past, I've reported that GoDaddy has said they're going to do that. I've checked repeatedly. And it's still the case that they're planning to do that be they've not yet announced a specific date and so forth. So we're holding our breath there. I've heard rumors from

others as well but I'd like to see more concrete data. This is as diagram that just shows the same idea, of the registrar polling the child zone to pull the information up.

ICANN has gotten burned a little bit about showing maps. So the maps that Dan York showed don't show country-level details but this is the country-level detail for Europe, taken from this week's maps from Dan York's system. We have, it must be Austria, and the Czech Republic, and Slovakia, and now Sweden—I believe that's correct—are not showing that DS automation exists there.

As I said, there's some rumors that are not yet at the level where one can depend on a date certain that things are going to happen but that more scanning is going on. We also have action within the Security Stability Advisory Committee to explore creating a formal report recommending that DS automation be supported. And there are some technical issues to be sorted out. Scanning is time-consuming and doesn't scale well but there are ways of improving on that.

The other half of what is of concern is the multi-provider coordination for both stable operation by multiple providers and glitch-free transition from one provider to another. This project has several people, some of whom are talking today and others who are participating on a regular basis from various organizations, and is moving along very nicely. It takes a while to put all the pieces together. One of the reasons it takes a while is that there are multiple moving parts.

The basic structure of what has to happen is that keys have to be moved between the zones, including from the new zone to the old zone so that there's cross-signing in both directions of the keys that they both have. So in order to make this all operational, there's software that's being prepared. There are changes to the standard packages that are underway. And then there will be multiple demonstrations that all of this works.

So here's a summary of where we stand in the big picture. And you can see from the number of checkmarks, namely two, that only parts of this are done. But you can also see from the square boxes that quite a bit is in progress.

Test beds are being configured and will be the basis for the demonstrations that I've talked about. The software components that have to be built are interfaces to the authoritative DNS servers. So those are outside of the control of this project. Those are changes to the standard packages, BIND and PowerDNS, etc. And then software that sequences the set of changes and has interfaces to all of the different parts and does checking of the process. So that's the gross picture of what's going on.

This is a slide that shows slightly more detail about what's done and what's in progress. There's a lot of design work that's in good shape and various documents that have been prepared. Some of the changes that are necessary are underway. I'll show you a couple more that flesh this out a little bit.

In terms of the standard name server packages, the ones that we're most familiar with, with respect to making progress that we need, are BIND, Knot, and PowerDNS. There are, in principle, three kinds of interfaces that could be used. One is a command line interface, another is Dynamic DNS, and the third is Restful API.

Command line interface is less useful because it requires full-scale access. You log into the system that's running it and that doesn't fit very well with trying to automate that without having more-than-necessary control. So what's highlighted here is the Dynamic DNS and/or the Restful API access. As you can see, BIND and PowerDNS are in good shape. Knot, as shown here, is coming along. And then there's room to add others. And we hope that, over a period of time, this will get fleshed out.

If we look at the DNS service providers, deSEC, NS1, and Neustar are the ones that we've been tracking. deSEC has everything implemented and you'll hear from Peter a bit about this. NS1 and Neustar are underway. And there is room for others. And we'd be happy to reconfigure this slide if they all want to move forward at once and pose the problem of how to fit this all in. And then, along the left edge, are the specific capabilities that are needed in terms of the capabilities of adding or removing various types of records.

I've been trying to build a library of these presentations. So this is a version of today's agenda with TinyURLs. This will be updated a little bit to be absolutely complete. And then, for each one of these so-called episodes, I have a list of the specific presentations that have been

accumulated over time. I'm just scanning quickly through this to show you that they exist and will be available for reference.

So with that, let me move on to Jaromír and turn the floor over to you and I'll continue to run the slides.

JAROMÍR TALÍŘ: Thank you, Steve, and thank you for the opportunity to share some information about our recent changes in DNSSEC automation in .CZ. Next slide.

So I'll be talking today about two different projects or things that we have been working on over the last couple of months. The first thing is that we have successfully migrated DNSSEC signer for .CZ TLD itself from Bind to KnotDNS. And the second is about development of our second-generation CDNSKEY processing tool. That is still in development but hopefully this will be deployed in production soon. Next slide. Okay. Next slide.

STEVE CROCKER: There we go.

JAROMÍR TALÍŘ: So for a long time, we started to do DNS signing, I guess, in 2006 or something like that. We always used BIND tools, dnssec-signzone and the others, for signing .CZ domain. And we've wrapped these tools by a set of shell scripts, handwritten by our sysadmins, that took care of the zone file checking, possible detection of issues, and preparing right

parameters for dnssec-signzone and running in regular interval. This was all driven by the cron jobs that were run every half an hour.

Even when we switched to offline KSK, where we have the KSK put into the safe and we do a little bit of similar procedure that ICANN is doing with the preparation of the ZSK for the next couple of months and signing those with the other team by KSK, we still used our shell scripts for doing this complicated procedure. Next slide.

So of course, we knew that since we have our own DNSSEC signing tool built into the KnotDNS, we should use it. The first motivation, of course, is, "Eat your own bread." That means that we should be the first one who is using that. This has never happened. The couple of countries started to use DNSSEC signer in Knot even earlier—I guess at least Germany, Denmark, and Costa Rica. But we finally managed to go this way.

And definitely, we took advantage of the features that KnotDNS provided. We now can use the online signing of KnotDNS so we can get rid of this cron-driven batch generation of the sign zone. But we just keep the DNSSEC signature generation on the Knot itself. In this way, the whole procedure is much more automated. And we, of course, can take advantage of the other features, like the zone checker that's integrated into Knot DNSSEC signer.

And as the advantage, when we have developers in house, let's say, we can easily contact them and ask for the more features. We can stop maintain the scripts that the people that created even doesn't work for

us anymore. And we can easily collaborate on the development of this tool. Next slide.

So the migration procedure was quite long. I would say it took several months. We prepared, of course, testing environment, where we tested all the procedures. And we continued from the less important domains, like the SLDs that are parked, our own SLDs that we have for our business processes. And at last step, we migrated our ENUM zone that we are still operating at the same level at .CZ TLD. And the last step, we migrated .CZ. Next slide. One back.

STEVE CROCKER: Oh. Sorry.

JAROMÍR TALÍŘ: Okay. So one specific thing for .CZ DNSSEC operation is offline KSK, which means that we have two separate teams—ZSK team that is responsible for only ZSKs and KSK team. It's the part of the [c-search] team, a security team that is responsible for holding the KSK.

And the way how it works is that the ZSK team that has the zone file under control, they pre-generate the ZSKs for six months and they create key signing requests that is sent to the KSK team, signed by the team to create the signed key response, and written back to ZSK team to check the signatures and import it to the hidden master and roll the zone.

So the process is still the same but this whole process was integrated into Knot. So we only changed the tools. And instead of calling the shell scripts, we used the Knot key manager with the command KSK sign KSR, pre-generate a couple of ZSKs, generate a key signing request and import the result—the signed key response. Next slide.

As a result, of course, there is a much easier management of the zones. Now, except offline KSK, the DNSSEC signing is fully automated. We managed to get rid of our homemade scripts and shrink the time of the zone generation from roughly 23 minutes to five minutes. We have now much better backup and restore procedure that was before used by some, I guess, rsync—something like that. And now we use the proper Knot backup and restore features for the high availability.

One interesting thing that we had to change, also monitoring because the previous version had … We checked whether there is some change in the zone of regularity. And there was a difference in the Knot and BIND, that Knot actually doesn't do any changes in the zone—that doesn't resign the zone and there are no changes, doesn't change the timestamp. So it [inaudible]. The sysadmins are woken up in the night if there are no changes. So there had to be some small updates in the monitoring. We are looking forward to some future development with this regard. Next slide.

So now, about the keyset management—how we call the CDNSKEY processing tool in the FRED registry. Next slide. So we started this project in 2017 as probably the first TLD that was doing that. And the features of our tool was that we were scanning the CDNSKEY from

whichever single site and we were waiting for the seven days. Every day checking and scans had to be identical before the DNS was bootstrapped.

We were sending the notification about all changes via e-mail and stored the result in a SQLite database. The whole system, we tried to make it dependent on the registry but we used for the link with the registry, the CORBA remote procedure call protocol, which is quite old.

Here are some statistics. Over a whole period, how many domains is managed by this procedure. You may see that it was growing for quite some time. There are two big increases when the big DNSSEC hosting companies started to use this automated DNSSEC administration. Surprisingly, over last year, the numbers are declining. I tried to do some exploration, why is that, but haven't found any particular reason for that. The overall number of DNSSEC signed domains is steady or growing really in low rate. So this may be just a normal situation that, right now, the DNSSEC space was saturated or something like that. Next slide.

So what are the issues? Of course, the seven days period is quite long. Anything may happen during this period. There may be some outage, for example. And then the scanning will stop and it will have to start again, actually, from the beginning. So we believe that scanning for multiple networks could make the procedure not even more secure. But also, we could potentially shrink this period to three days, like some other countries are doing.

We've experienced that that end users are quite confused by e-mail communication. So in the new version, we will stop sending e-mails and we will just put the information on the website about the results of the scans.

There is one specific issue, which is the conflict with the registry lock. There's always been a question whether we should ignore it or not—if the user will set that registry lock and then start to use this automation. What should have preference? This is yet not decided.

And as we started with SQLite database for storing the results, we found out that high availability is quite hard with that so we plan to switch to regular PostgreSQL that they are using for normal registry operations. Next slide.

So this is an overall diagram of the change to the system. In the first version, the core was the tool called cdnskey-scanner. It was a command line tool that expected the list of the domains on the standard input in the command line and scanned for CDNSKEYs and returned the result on the output.

So in the new architecture, we upgraded that to something that we call now, cdnskey-processor. It's not just the command line. It's a server-side component that has the API. And it has the master part that is spawning the workers—multiple workers—providing them with the batches of domains that they need to scan, getting back the result, and storing the result in the PostgreSQL database. And this whole component is accessible via API. So the registry itself just checks, via

API, the results and behaves accordingly to do the updates or something like that. Next slide.

This is a little bit more detail in the this whole cdnskey-processor. As I said, there is a master part and the workers. It's using RabbitMQ broker for communication between the master and the worker and scheduling the work for the workers. It has control interface that is used for loading the domains into the system and checking the results. It also has some diagnostic interface for monitoring the whole process. The whole server is accessible via GRPC remote procedure call that we use—that we replaced almost everywhere in the registry for the COBRA GRPC technology. Next slide.

This is the prototype how probably some result may look like—the detail of all scans that we have been running for the domains, for the name servers. But probably, we will have some better, shortened version for the general public. This is mostly for the name server operators that want to find out why, potentially the CDNSKEY was not scanned or what was the result of the procedure. Next slide.

So in summary, there are quite a few improvements. The architecture is now much more robust with multisite scanning. It has this batch processing possibility where not all the domains are loaded at once but it can be separated in the batch. And if there is an error on the scanning, only the batch may be started again. Hopefully this is a little bit better integration with registries. If some registry, for example, would like to use it, I believe that now it's much easier.

At the moment, we are running the version in parallel with the old version and we are checking the results, if they are the same for the previous version and the new version. And we are evaluating potential differences.

At the moment, the whole procedure to scan the .CZ one, 1.4 million domains, it takes 13 hours. It's not the fastest possible approach. We could make it faster but we wanted to spread it to as long a procedure as possible to not overload the name servers that we are scanning. But if we are comparing the same numbers, the same load, we see that there is a little bit—that the new version is running a little bit faster. I would say that in two or three months, we hope that we will run the new system in production. Next slide.

And that's probably all so thank you for listening. If there are any questions, I hope I will be able to answer them.

STEVE CROCKER: Thank you, Jaromír. Mats?

MATS DUFBERG: Thank you very much for giving me the opportunity to talk about Zonemaster tests for CDS and CDNSKEY. Next slide, please. We know, all of us, that the DNS is crucial. We also know that DNS errors can be interpreted as network issues. DNS is complex and DNSSEC makes it even harder. So we really need some way to check things. We need a tool. Next slide, please.

Zonemaster is such a tool that has been available since 2014. Zonemaster is targeted at the delegation of the zone and it verifies, for example, that all name servers give consistent responses. We know that if you get different responses from different name servers, it can cause a lot of trouble. It also verifies the responses on various queries and it verifies DNSSEC. Next slide, please.

So Zonemaster can be used to meet different needs or troubleshooting. You have a domain that you wonder why it's not working well. You can use it for monitoring, to make sure that your domain is working properly. You can use it for statistics and measurements. And you can verify a new domain that is ready to be used. Next slide, please.

So the Zonemaster is updated regularly and the latest release was in June this year. We will have a new release in the end of November. So we are working on that release. So in June, we added tests for CDS and CDNSKEY. Besides that, we also added translation into Finnish and Norwegian. So Zonemaster is a multi-language tool. And other improvements. And here, the focus is on CDS and CDNS key, naturally. Next slide, please.

So the tests that we have added to Zonemaster, we call them test cases. We have the IDs, DNSSEC15, 16, 17, and 18. So 15 focuses on the existence of CDS and CDNSKEY. So does this domain have any CDS or CDNSSEC record at all? Otherwise, there is nothing to test for CDS and CDNSKEY, of course. DNSSEC16 validates CDS to make sure that it's protocol valid. DNSSEC17 does the same thing with CDNSKEY. And DNSSEC18, which does not exist in the current version of Zonemaster

but will come in November, validates the trust from the current DS, if any, to CDS and CDNSKEY. Next slide, please.

And also, back up, please. I just want to comment that all tests are defined with written specification in GitHub. And you will get a link to where so you can check exactly what these tests do. Next slide, please.

So if we look at DNSSEC15, it would notify if the zone has CDS but not CDNSKEY or vice versa because that is what the standard says. You should have both. And it's error if not all name servers had the same set of CDS or CDNSKEY. It should be the same on all name servers. It makes an error if the zone had both CDS and CDNSKEY but they do not match. They must match. And it also makes an informational message if the zone has no CDNSKEY or if it has both. Next slide, please.

DNSSEC16 validates the CDS. So there will be an error if the CDS RRset is unsigned, if there is CDS without DNSKEY, if the RRSIG is invalid for the CDS RRset, if the CDS RRset is signed with an unknown DNSKEY, and if the CDS RRset is a mixture of "delete" CDS and normal CDS records. I don't know if you are familiar with the details of CDS but you can create a special CDS Key that actually removes the DS, not adds it, as the normal CDS record does. Next slide, please.

So it will emit a warning if the CDS record does not match any DNSKEY. That is the equivalent of a DS not matching a DNSKEY. If the DNSKEY RRset is not signed by all DNSKEY that the CDS records point at, that's also a warning because that's what you expect from the DS record and the CDS is the DS to be. Next slide, please. So we have DNSSEC17 that

does the same thing with CDNSKEY as 16 does with CDS. Next slide, please.

So the last test, DNSSEC18, that will come in November, will validate the chain of trust from current DS to CDS, or it should say, CDNSKEY RRset. It could be the case that the zone does not have any DS and that's okay, of course. Next slide, please.

So who is behind this Zonemaster. Since 2013, we at the Swedish Internet Foundation and AFNIC together develop and maintain Zonemaster. And we regularly publish new versions. Next slide, please.

And everything is available. So you can go to GitHub and find the full documentation and full specification. The license is permissive and you can install it as is or for other use. We have reference installation at Zonemaster.net, which is a working installation where you can check your zone. So if you have a zone with CDS or CDNSKEY, you can check it there. And there are more installations around the world, both public and private.

So I think that's the last slide. Or next slide, please. Yeah, features. Okay. So we based Zonemaster on standards and everything is documented in GitHub. Yeah. That's my last slide. Thank you for listening.

STEVE CROCKER: Thank you very much. Did I cut you off? Is there something you wanted to add?

MATS DUFBERG: No. If there are any questions, I'm happy to answer those.

STEVE CROCKER: Yeah. If there are questions—and we already have one question—please put them in the chat and we'll try to get to them at the end of the prepared talks here. So, Peter, now turning it over to you.

PETER THOMASSEN: Yes. Hi. This is Peter Thomassen from deSEC. I'm going to talk about authenticated bootstrapping of DNSSEC delegations today. [I will] talk about CDS, which is not authenticated for bootstrapping. This is a new effort in the IETF. Not yet. We're going to pose it to the DNSOps working group. The title of the draft is listed on the bottom of this slide and it's also clickable if you download the slides. Next slide, please.

Yeah. So as was already said in the first session today, the DNSSEC validation rate, globally, is around 28%. It varies by country but it's a significant deployment. Compared to that, the rate of delegations to be secure and to have DS records at the parent is only 5%, globally. It depends, also, by country. Some countries have financial incentive, for example. But overall, it's very low. And the DNS operator usually cannot do much about it. For example, at deSEC, we sign all the zones and we still observe that only less than 50% of our zone owners actually put DS records at the parent. Next slide.

So the big question is why is it that the DS prevalence is so low? Why isn't DNSSEC adoption taken on the secure delegation side? Next slide, please. So let's take a look how DNSSEC bootstrapping is done today.

Everybody knows here that to secure a delegation, you need to convey DS or DNSKEY records to the parent.

And there are several approaches to do that. In Germany, I know a few registrars who just query the DNSKEY record, not the CDNSKEY record—for good reasons, probably—from the target domain and assume it's not been tampered with, hope for the best and just set it up at the parent.

A very common case is for the registrant to somehow retrieve the key material of the public key from the DNS operator's web interface and put it in some other web interface at the registrar that's usually confusing—so many different form fields.

There used to be some approaches or ideas about REST interfaces, which appear to have died. And then, there's also the RFC 8087, that has been mentioned before, for deploying CDS/CDNSKEY from insecure but that requires somehow making it likely that what you retrieve is actually legitimate. So the common approach is to do monitoring and query that for three to seven days from different vantage points.

But in the end, you don't get rid of the downsides, which are it's still unauthenticated, or out of band, and/or slow or stateful at the parent for the monitoring, and error-prone, and lots of parties, and no automation and … Next slide. We think it's not the state it should be in. Next slide.

Yeah. So let's take a look at the DS signaling model to understand better why that is. This is a modification of the diagram that Steve showed

earlier. It shows how many parties are involved. One common way is that the registrant picks up the DNSSEC parameters from the provider, hands it over to the registrar, both using HTTPS. So it's secure, which is why the arrows are solid—or authenticated. And then it goes on via EPP.

An alternative approach, which is the non-authenticated CDS from insecure approach is that the parent—the registry or registrar—pull from the bottom, from the DNS provider. What we would like to do with our proposal is to secure the thing on the right. Next slide.

So if the two arrows on the right could be secured and authenticated, then essentially, you could still be doing the CDS stuff but not only for rollovers, also for bootstrapping. And it would be automated, in-band, immediate, and stateless. Next slide.

To do that, we propose to use an existing chain of trust that exists to the DNS operator—I'll explain what that means—and transfer trust from that chain onto the target domain. Next slide, please. To illustrate that, I think it's better to have a visual representation. So let's assume that we would like to secure the delegated—for example, .com—and we have the root zone here with two top-level domains submitted. Both of them are securely delegated also. Next slide.

Now let's say the DNS provider we're talking about has the domain, provider.net. ns1.provider.net is what their name server hostname is. And we will make the assumption now, or the precondition, that the name servers' hostnames themselves are securely delegated. Next slide. Now, the customer registers the example.com domain, which is not secure yet. And commonly, if CDS is supported, the DNS provider

would now add the CDS records to the example.com domain. Next slide.

Our proposal proposed to co-publish the same CDS and CDNSKEY records under a subdomain of the name server's hostname, which is securely delegated and to retrieve the CDS records also from the name server provider and to compare them against the ones in the target zone. Next slide.

So the first thing the registrar or registry would do is do the unauthenticated poll, as usual. Next slide. And then, instead of waiting for seven days, would instead do the validation against what's in the name server hostnames zone. Next slide. Once it's been verified, the registry or registrar can put in the DS records at the target zone, in the parent zone. Next slide.

So in this approach, we use an established chain of trust on the left and take a detour. And as a result, you get authenticated and immediate bootstrapping that is resilient against on on-wire attacker. Next slide.

There are some technical considerations. This looks more complicated than it is. One objection could be that this is overloading the use of CDS and CDNSKEY records. That is not the case because the existing use is only the apex and what we're proposing is at a subdomain of the name server hostname. So it's not a collision.

Second, it could be, if you just prefix the target domain name to the name server hostname, that you hit some length constraints or the number of label constraints in the DNS. To avoid that, we propose to

has everything in the target domain name, in the prefix, except for the first label. So in the case would be example .hash of .com .ns1.provider.net. We also propose to add an extra label between that has and the name server hostname. We propose that to be _boot. And that enables you to delegate all the bootstrapping stuff to a separate zone.

The advantages of these two things are that it removes the risk of accidentally touching your name server that—the name server's A records and all that—when you actually want to do bootstrapping stuff. You don't even have to have the same keys on the same servers. It reduces churn on the name server zone. It allows splitting off DNS operations so you can do a different key or you can host your .com bootstrapping stuff under a different infrastructure.

And most notably, the structure also allows the parent to discover bootstrappable domains specific to that parent, by looking only at things under the hash of its own name—for example, under the hash of .com. And if the bootstrapping domain, the name server domain, uses NSEC, then the parent can do a NSEC walk off the bootstrappable pending domains. So you would query NSEC at the hash of your suffix and it would point you to the first bootstrappable domain. Then you do bootstrapping and then you query the next NSEC record. Next slide, please.

Yeah. How about some numbers? So there are some experimental implementations at deSEC on our name servers. We deployed this under our name server host names for about 13,000 or 14,000 domains.

We also put out on implementation on GitHub that's linked in the draft tool that can fetch those records from the name server zones and also from the target domain itself, do all the validation and if everything's fine, output things that the parent could insert into its zone.

We also talked to some registrars, and registries, and DNS operators who are ready to do experimental implementations of this over the next few months. Still, it's interesting to look into what the practical readiness of the Internet ecosystem is for that. So, next slide, please.

The preconditions needed for deployment are that the name server targets are in securely-delegated zones. That may or may not be the case already. So it will be interesting to look at that. If it is the case, then the DNS operators don't have to do anything except inserting those extra records.

If it's not the case, we think it should be manageable for DNS operators to do that and secure their name server targets' delegations because, apparently, there are providers who are offering DNSSEC to their customers so they probably can do it on their own infrastructural domains, too.

Also, the method doesn't work if the name servers are in the same zone as the target. You get a catch-22. We figured out that this is not the case, for 99% at least, of name server targets in the .com zone, for example. So that should not be a common problem. And the target zone needs to be signed, of course, so that's a no-brainer.

Let's take a look at some more concrete numbers. Next slide, please. We investigated the top million domains from the Tranco dataset and for each domain, we extracted whether the domain itself is securely delegated, whether the zone contents are signed. We extracted our name server targets in the delegation, and for each target, we observed whether it is securely-delegated as well.

From that, you can compute, for each domain, whether it is bootstrappable, which means it is not yet securely delegated but all its name server targets are. So you could do this kind of signaling announcement stuff. And from that, you can look at what the fraction in the dataset it. So next slide, please.

We had about 3 or 4% timeouts and probably could have improved that. But for an overview, that's good enough. We looked into 960,000 domains and found that about 5% of them are secure and signed. That's consistent with the initial number that we got from other sources. And we found that 25% of those delegations have name server record sets where all contained name server targets are securely delegated.

That means that roughly 25% of these delegations are able to have things about domain announced securely by the name server operator. If you subtract from that the number of domains which already are securely delegated themselves, you arrive at 22% of the top million that can be, right away, bootstrapped. Next slide, please.

We broke this down by top-level domain and also by provider. I'm waiting for my slides to update. Steve, are we …? Oh, cool. Okay. So on

the left-hand side, you can see by top-level domain. You can see, for example, that in the .com zone, there is 490,000 delegations amongst the top 1 million from the Tranco dataset, 24% of which are bootstrappable—so not securely delegated but their name servers are.

So if the .com registry, which of course is not a country code domain so it's hard to do, would implement this method, then they would be able to bootstrap 160,000 domains. The fraction, 24%, is similar across other generic and country code top-level domains, like .org, .net, and others.

If we break it down by provider, the approach we took is we looked at the SOA record of the name server and extracted the RNAME field. And we found a quarter of the top million SOAs at Cloudflare and three quarters of these 250,000 domains are bootstrappable, name servers are secure, but the zone is not securely delegated itself. If Cloudflare, for example, implemented this, 190,000 domains could be bootstrapped right away. Next slide, please.

Let's take a quick recap. This the second-to-last slide. What are we having here? We have the signaling mechanism from the DNS operator to the public. For example, the parent can consume it and it allows the DNS operator to announce zone-specific information in a secure fashion, in-band, immediate, and without involving any third parties. An interesting thought could be what else could be done with that, that is not DNSSEC-bootstrapping-related. Maybe you want to announce something else that's interesting. Next slide, please.

Earlier in this session, it has been mentioned that there is multi-signer approaches, which is when you have two DNS operators which serve

and sign the domain with their own keys. In that case, each operator has to announce each other's keys and you have to do some kind of key exchange for that. So maybe that could be used as well. It's not part of our specification. It's just a brain teaser. So next slide, please, the final slide.

In a nutshell, multi-signer gives you redundancy in multi-homed zones without breaking validation. And it also gives you a smooth transition during provider transfer without going insecure because you can map that onto a multi-signer scenario where you have a brief period of multi-signing. It works by all involved operators, usually two, advertising each other's ZSKs within their own DNSKEY record sets and signing that with their own keys and also the parent advertising everybody's KSKs in the parent.

So you can do that key exchange, either manually, but you can also observe that the DNS operators usually know each other's NS record sets or NS target names because they know the common NS record set. And with that observation, if everybody's name server targets are securely delegated, you can use the same signaling mechanism for that kind of key exchange, or potentially for other stuff in the future. It's just something we wanted to put out there in case it is helpful to anyone. Next slide, please.

Yeah. So with that, I think there will be more contributions on multi-signer later in the session. Thank you for your attention. Thanks to our sponsors. And I think questions are for later.

STEVE CROCKER: Thank you very much, Peter. Pouyan, it's your turn.

POUYAN FOTOUHI TEHRANI: Hi, everyone. I'm Pouyan. I'm waiting for the slides. So I'm here on behalf of Eric, Thomas, and Matthias to introduce out latest project called DNS Resolver Observatory. We're not 100% sure on the naming and branding. Our website, the demo website, is dnssecviews.net. We're going to get back to that. And throughout the presentation, if you have any questions—something that you want to contact me—you can reach me under pft@acm.org.

So let's get to what we're doing. Next. So the motivation is easy. If you're using DNSSEC, you would change your keys. You would transition your keys at some point in time. Those changes can be observed instantaneously at the authoritative name servers.

But at the other end of the stick, users rely on recursive resolvers, which use different policies. There are number of factors—timing, caching, multiple signers, etc.—that would influence the propagation of those changes that you made to your keys. At the same time, infrastructure providers are interested to know how their services are being observed before, during and after the transitions. Next.

So we have been doing the first part—next, please—the first part of observing the changes at authoritative name servers. In the past 15 years, Eric has been maintaining SecSpider. We have also published some results and insights, recently, on their archive. You can take a look at that.

Next, what we want to do now is—next slide, please—to take exactly what the users are seeing. So if you remember the presentation from Steve, we want to put a checkmark at the real-time observation of what is happening at the client's end. Next slide, please.

Just a small teaser. We have run this project and we have already seen that even the very same host, which is multiple resolvers or has multiple interfaces, would get different responses for the same DNS query. So I'm going to get back to that. Next.

The use case that I have as also motivation of why we are doing this is the multi-signer DNSSEC. So there are two deployment models. I think it has been talked about. The first one is the owner has the KSK, signs the ZSKs of the providers, coordinates the DS through the parent zone. Next slide, please.

And the second deployment model of multi-signer is that each provider has its own set of KSKs and ZSKs and the owner coordinates the DS records with the parent. Next slide.

What we have in this case is the key transitions—the process models for those key transitions—are more complex that what we have in normal—I don't know—double-signature KSK transitions. So it means more coordination between multiple stakeholders. As such, it's interesting to see what actually happens at the client. Next.

So what we do is pretty simple for the time being, we are using the RIPE Atlas network, which has distributed probes all over the globe to do the legwork for us. On the righthand side. You see the infrastructure

operators who want to measure a zone, they register their zone through our front end, the dnssecviews.net. We schedule regular measurements over the RIPE Atlas network. Next slide. And we wait for the results. We persist the results. We aggregate them analyze them, and provide statistics to the infrastructure operators.

So how does it look like? Next slide, please. Before we get to the actual screenshots, the way we do it is easy. We find the zone apex. We schedule measurements for DNSKEY, DS, NS, SOA. DNSKEY resource records are the cornerstone. The rest of records that we fetch are for validation and to find out which ZSK is actually in use.

We set a number of random probes, currently only in the US, to execute the measurements. And we parse and serialize the data into the database if the response and valid and it's signed, which is not always the case. We also record which probe has observed which resource records and which signatures, at which point in time, from which of the resolvers it has been using. Next slide, please.

And finally, now that we have those raw data, we calculate different combinations of observed DNSKEY sets and the set of active keys. By active, I mean the keys that are being used to sign resource records—so for DNSKEY, which KSK or KSK set has been used to sign that resource record and for the other—as I mentioned, NS, SOA—which ZSKs have been used to sign them. We color-code each combination and calculate when each probe has seen which combination. Next slide.

So at the end, the providers can see which recursive resolver observed what combination of data at any point in time and space. By space, we

show it on a map so it's geographical space but it's actually the topological space. But they correlate together. You know that. So next slide, please.

The way it looks, we have a simple dashboard that you can enter your zone on the top. If it's not in the database, if it's not being observed, you're going to be asked if you want to monitor that zone. Next slide. Then you choose one observation point. The measurements are being done every hour so it takes at least one hour until the first set of data is there. You choose an observation point. And then, in the log section, you can see—like in this example, two probes have seen this specific DNSKEY set and they have seen that one ZSK and one KSK are in active use. Next. Next slide, please.

If you choose another observation point, you're going to see different results because everyone who has seen the same thing, we put them all together. So if you see different color codes, you can be sure that you're going to see different observed responses. In this case, it was interesting to see that multiple ZSKs and KSKs are active. Next slide.

It was not actually due to the fact that there was an ongoing double-signature key rollover but that these probes that we have observed here, they were having multiple interfaces—IPv4, IPv6—using different resolvers. So the systems registered that as observing different set of responses. I have changed that in the interface. So if you visit the dnssecviews.com, you're not going to get confused. But it was just an interesting observation that the same host can observe different responses for the same zone. Next slide, please.

So now, we go all the way back to our use case. Now that we have the data, we have developed some heuristics to actually detect multi-signer DNSSECs. For the first deployment model, where the owner owns the KSK and the providers have their own ZSK, we say that if we see that the same KSK is in active use by all of our vantage points, all of our probes, but different probes are seeing different active ZSKs, then we would say, "Okay. it's most probably multi-singer deployment model one." Next slide, please.

As an example, here we have two probes. Both see exactly the same DNSKEY set, which is the requirement for both multi-signer deployment models. So they see the same DNSKEY RRset. It has been signed by the same KSK, all good. But if we look at which ZSK is actively in use, we see that one probe sees 6178 KSK—that's its key tag—and the other one sees another one. So we would say, "Okay. This is most probably multi-signer deployment model one." Next, please.

For the second one, again, they must observe the same DNSKEY set, but this time, they see different set of ZSK and KSKs are in active use. But at the same time, the same set of KSK and ZSKs are being observed together. Next slide, please. So in this example, again, the DNSKEY resource record is the same but we see different KSKs have been used and different ZSKs. So we would say, "Most probably, this is a multi-signer deployment according the second model." Next slide, please.

So there are some caveats to these heuristics that we have developed. There might be standby keys. We wouldn't be able to assign that. If they are not actively in use, you wouldn't be able to figure which provider

actually owns which key. The question is, do we care at all? Another one is if there are actually ongoing transitions, are there going to be marginal cases that we would classify as multi-signer DNSSEC but actually, they're not? And what happens if people are using Anycast resolvers like the [quad eight]? Can they skew the results or not? This are questions that we need to answer. Everything is really fresh at this time. Next slide, please.

So to conclude, this is a known fact. There is a measurable discrepancy between records at that authoritative name servers and what recursive resolvers actually deliver and what clients see. The Resolver Observatory has been conceptualized to give the operators the opportunity to follow their DNSSEC deployment from the perspective of clients and in real time—to see if they turn a screw at their end, what happens at the end by the clients. And the aggregated data that we actually provide should be used to improve deployment practices and figure out acceptance criteria.

That's it from my side. Thanks. Looking forward for the questions. Steve, there are some backup slides that you can just skip right away. Thanks.

STEVE CROCKER: Thank you very much, Pouyan. Here's the backup slides. And Ulrich. Too far.

ULRICH WISSER: Yeah. Thank you, Steve. Thank you for giving me the opportunity to speak here. Yeah. So I was asked to present about the CSYNC record. Recently, at the Swedish Internet Foundation, we have deployed the CDNSKEY scaling. And we have been working with Steve and others on the multi-signer DNSSEC. And one part of automating multi-signer DNSSEC is this CSYNC record. We thought it would be a good idea to look at little bit closer at it. If you have any questions, I have seen that Wes is on the call. So he can explain this probably much better than I can. But okay. Next slide, please.

We actually plan on deploying CSYNC scanning for the .SE and .NU zone this year, hopefully. So let's see what this does. CDS and CDNSKEY is like the child—DS child, DNSKEY. We all have been talking about this at this workshop a lot of times. The child zone will publish the DS or CDNS key record and the parent will scan for it and then put the according record—the DS record—in the parent.

So if you then think that the parent might would like to look at other records than just the DS record, then we would have C-whatever records and we suddenly had to double or record types. And that might be not the best idea. That's why there is a CSYNC record. Next slide, please.

So the CSYNC record is obviously an RFC. It's published in the child zone. And it tells the parent which records to copy. So again, the parent scans for this record and then there is a bit map that explains which record types should be copied. So next slide, please.

Yes. So the bit map is the same bit map as we have in the NSEC record., Basically, the QTYPE one is bit one in the wire ordering and then QTYPE two is bit two, and so on, and so on. There is some optimizations where we don't always need to send the full list of possible records but we can just stop sending when there is only zeros following. Yes. So it can be specified which RR types the parent should copy but the parent decides which RR type to accept. And the parent might have policies around this, too. So next slide, please.

Of course, the most obvious use case for this is actually the name server synchronization. The name server synchronization has been a longstanding problem for the DNS. We all know lame delegations. It would be so much easier if we had an automated way of updating the parents. And well, we have. So in the case of the name server synchronization, the RFC specifies that if you set the NS bit in the bit map and you set the A and AAAA records, then A and AAAA actually mean, "Copy the glue records for my name server." That way, we can really update the parent with the necessary information for name servers. Next slide, please.

Yes. So here is an example record. There's a little bit more details to what this thing can do. Here's [SOA of] the example.com. Okay. I want the parent to copy. And then we have, actually, the SOA serial. It says the actual SOA serial from where you copy the data has to equal or higher to 66, in this case. So you can specify that the parent doesn't go back to an old version of the zone or anything. So it's a good idea.

And then you have two flags here, currently. And it says SOA minimum is like, "Look at this SOA serial that I sent you or don't look at it." In that case, it is recommended that you set the serial, actually, to zero.

And then you have the flags immediately. That says, "Okay. I really want the parents to update this now." So read the record, validate it, and then update the parent. The idea is that if you don't set it, the parent might have some out-of-band interface where you would have to agree to this update or validate the update somehow. But if you sent the immediate flag, then the parent knows, "Okay. I'm allowed to update this immediately in my zone."

And then, after this comes the bit map. And the bit map specifies what types should be copied. In this case, it would be an update of the name server records. So the NS set that the parent has gets replaced by the NS set of the child. So those are the glue records.

I think that is everything about the CSYNC record that I had to tell you. So I think we have some time left for questions. Thank you.

STEVE CROCKER: Indeed. Thank you very much, Ulrich. Yes. We have time for questions. And I am obliged to offer one comment. I misspoke at the very beginning when I said that Austria had implemented DS automation. That should have been Switzerland. And I'm sure that includes Lichtenstein as well. I can't read a map. And it's much later for you guys than it is for me but nonetheless.

All right. Any questions? I see there's two questions from Eugene. And Eugene, you've asked with respect to the question of the .CZ team but Mats answered you. I don't know if that answer satisfied you and then you had an additional question, which is still pending, about how do you handle DNSSEC offboarding. More specifically, when CDS zero is published and when DS is removed from .CZ, how do you communicate to the child zone owner that they need to wait for the DS TTL to expire before disabling DNSSEC on the child zone? I suspect that, Jaromír, this was really intended for you.

JAROMÍR TALÍŘ: Yeah. Of course. I think I can even get back to the first Eugene question about the SERVFAIL. Actually, there are two use cases—two different processes when we are doing CDNSKEY scanning. One is when we are doing the secure update of the DNSKEY, which requires to do the scanning via secure DNSSEC channel. So I this case, when we will get CDNSKEY and it's not signed, that means there is no way to answer. It could be SERVFAIL or the other. Then we treat it as invalid record. We will not do the update.

And in case of the bootstrapping issue when we are actually scanning the non-secured records, not-signed records, then I am almost 100% sure that we will treat a SERVFAIL as the error and we will have to start the scanning from scratch, like next 7-days period. This is definitely something that will change in the new version. So this is the answer for the first question.

And for the second question that's related to DNSSEC offboarding, I would say that this is not a question for us as the TLD registry, that when we remove the DNSKEY of a DS record from a zone as a result of the CDNS key zero, then definitely, it's the responsibility of the operator to poll for the status if this has been completed and to behave accordingly. I'm quite sure that this is the way how not to DNS signing works. There is some configuration of the resolver that's used for the checking—the result of the whole process of the changes in the parent.

And this is not just for the offboarding, for removing the DS, but this is also the case for the KSK rollover because they're not allowed to do automated KSK rollover via CDNSKEY. So the same situation where the child operator, not itself, must take care of what is the current state, whether the publishing is completed. And actually, the system can actually start to use a new key. So I hope this answered the question.

STEVE CROCKER: Thank you. Eugene, I've marked your questions as answered, if you're okay with that. Thank you. Wes, you have your hand up.

WES HARDAKER: I do. Thanks, Ulrich, very much for presenting about CSYNC. You freaked me out when you said that you might ask me questions because I wrote that RFC six years ago and that's about the last time that I've remembered the details from it.

But my question is actually so people back then were complaining that they didn't think people would implement CSYNC much because of

resolver-registry type of scanning and people' wouldn't necessarily do it. I didn't fully understand, then, what the difference in difficulty was between CSYNC and CDS. So did you find one of them more easy, more or less difficult to implement, or are there any lessons learned there?

ULRICH WISSER: At the registry, we are resistant to change records. We usually want to have a request from the registrant. But we decided that for CDS, this is an indicator by the registrant that they want to change. And the argument follows for CSYNC. We really think that we should make all this stuff a lot easier and this makes it easier for people to keep the data up-to-date.

We have been the first TLD to be DNSSEC signed and we really like this DNSSEC stuff. So we want to promote DNSSEC and we want to make DNSSEC easier. So multi-signer is the way to go but you can only automate multi-signer if you automate the NS changes. That was the motivation we had so that—make it easy for everybody. I had no complaints, not even from our registrars.

WES HARDAKER: Glad I could help.

STEVE CROCKER: Jaromír, were you about to offer something as well?

JAROMÍR TALÍŘ: We have been thinking about it in the past but so far, I think we didn't see quite a big use case for CSYNC. I guess the idea was it probably could be good for NS updates, for cleaning the lame delegations or something like that, which I don't think it's super interesting. With this multi-signer approach, probably it gets a new dimension that we all have to rethink the usefulness of this. Maybe we will implement it in the future as well.

STEVE CROCKER: I've sat in some discussions that were related in the following way—that with respect to how to update DS records, the two strategically-different ways are a push versus a pull. So the push would be that there would be an interface that a zone operator could call and push the change upward.

In those discussions, there was some consideration about, "Are you only allowed to change the DS record or can you change other records?" And those discussions didn't go anywhere except that the question was raised, "My goodness. How big a door are we opening and what kinds of bad things might happen if we allow automated changes from below without going through the usual process? It's not an answer. It's just a consideration that—a degree of caution about these things. The CSYNC doesn't have the same issues because it's pulling-based as opposed to push-based.

Any other questions? Yes. There's a hand. Mats?

MATS DUFBERG: I have a question to Jaromír. Some use CDS but you use CDNSKEY. Do you have some reason why you selected CDNSKEY instead of CDS?

JAROMÍR TALÍŘ: Yeah, definitely. There is a reason. The reason is that because of the architecture of the registry. For us, we don't collect the DS records from the registrars. We collect the DNSKEYs, and we count DS records ourselves. And we see the advantage on that, that we can use something we've already called for doing that. We can reuse the one key for multiple domains.

So the big DNS operator can have just one key and sign the key with all the domains. And he can just upload one DNSKEY via EPP with one update. That's all. Doesn't need to make thousands of updates, with the change of the key and the setting the key. So because of this architecture, we use DNSKEY. So we only scan CDNSKEY.

Actually, I have a question for you back because it attracted my attention that you mentioned in your presentation that the standard says that there must be both CDS and CDNSKEY. It's quite a long time ago when I was reading the standard but I don't see this I the case. I think that you can definitely just use CDNSKEY or just CDS.

MATS DUFBERG: The standard says that there should be both, not must. So it's clearly stated in the RFC.

JAROMÍR TALÍŘ: Okay.

STEVE CROCKER: Here's a question, which is completely different, from Andrey. How could they create an alternative DNS system in Russia? Who wants to speak to that? Silence. Let me ask the question in a different way. Where is a good place for that question to be discussed?

MATS DUFBERG: If I may.

STEVE CROCKER: Please.

MATS DUFBERG: I guess that this is DNSSEC-oriented so this is not the place. DNS-OARC could be a place to discuss something like that. Or in an ICANN meeting, it could be the Tech Day, which has already passed, that also covers the DNS questions. So next Tech Day could be a time to discuss something like that.

STEVE CROCKER: Thank you. Peter, you raised your hand and then took it back down.

PETER THOMASSEN: Yes. You were asking what the right forum would be for the question and I think the question is underspecified. I think before deciding where

to answer the question, the question needs to be clarified in the sense, whether it's a very technical question, like what would be an alternative system to the DNS, or whether it is a split-horizon DNS question, or whether it is a question of compliance and regulation, how it could be done by policy and all these. So I think these different types of questions would have different forums.

STEVE CROCKER: Thank you.

ULRICH WISSER: Steve?

STEVE CROCKER: Yeah. Go ahead.

ULRICH WISSER: I had a small remark to Pouyan, actually. Pouyan said that in a multi-signer setup, the servers would answer with the same DNSKEY set and that is actually not the case. I would say it's more likely that they answer with a full ZSK set but only specific KSK per signer because you don't need to cross-sign the KSK. You only need to cross-sign the ZSK.

POUYAN FOTOUHI TEHRANI: The signature might differ but the DNSKEY set, the RRset that you receive, should be the same.

ULRICH WISSER: No. Usually, DNS does that. Usually, DNS has the same set everywhere. But in the multi-signer case, you would have the signer one has KSK one, signer two has KSK two, and they exchange the ZSKs. But when you get the answer, you get only the answer from one of them and that doesn't include the KSK of the other signer.

SHUMON HUQUE: If I could make a quick comment there, Pouyan and Ulrich, I think Pouyan's presentation talks about both models. In model one, actually, the DNSKEY set is the same. In model two, the ZKS subset is the same but the KSKs differ between the two providers.

STEVE CROCKER: Thank you. Andrey has raised his hand. Are we able to give him the floor?

KATHY SCHNITT: Yes. Andrey, you just need to unmute your mic, please.

ANDREY SHCHERBOVICH: Excuse me. Sorry. My nontechnical background … I'm a lawyer by education and I am and ICANN 72follow, actually. [I asked] this question about Russian [security] because I think this is [the issue here]. In my opinion, [inaudible] DNS, [inaudible]

STEVE CROCKER: Andrey, you're fading in and out and we're out of time, in any case. Let me recommend that you send e-mail to any of us and we'll endeavor to engage with you on the questions that you're asking. I'm very glad that, as an ICANN Fellow, that you're jumping into this. And we'll try to be as helpful as we can. If you need help with finding any of our e-mails, almost anybody that you talk to will help you out. I'll be happy to put mine in the chat here. "In the chat," he said. Just a second. Feel free to send me e-mail, or any of the others, or elsewhere in ICANN. Many of us would be happy to engage with you. So thank you. I'm sorry. Go ahead.

ANDREY SHCHERBOVICH: Thank you very much, Mr. Crocker. It's a pleasure. That's [inaudible]. Thank you very much.

STEVE CROCKER: You're coming across muffled so that I'm having trouble. I think we're at that time that we need to bring this to a close. So thank you, everybody. An absolutely heavily-packed set of presentations and lots of activity. Let me turn things back over to Kathy to close out this session and lay the foundation for the third session—third part that's coming up.

KATHY SCHNITT: Thank you, Steve, and thank you panelists. This concludes Part Two of the DNSSEC and Security Workshop. Please join us back here, same link—I put it in the chat—for Part Three, which will begin at 23:30 UTC. And with that, please stop the recording.

[END OF TRANSCRIPTION]