SAN JUAN – DNSSEC Workshop, Part 3
Wednesday, March 14, 2018 – 13:30 to 15:00 AST
ICANN61 | San Juan, Puerto Rico

JACQUES LATOUR: The quiz at ICANN59 2017, so that's the sheet. So, there should be enough around. So, if you don't have one, let me know. We can bring some. So if you need one, let me know. All right. All right, oh yeah. You can do multiple. Put Warren's name on [inaudible].

UNIDENTIFIED MALE: [inaudible] one of them.

UNIDENTIFIED FEMALE: I've got three. I'm going to [inaudible] different answers.

JACQUES LATOUR: Excellent. All right. So, from what I learned from Roy Arends is you can create your own rules. So, we have six good answers per question, six and a half point for good answer. There's eight and a half questions and 784 points. So, this should be fairly easy. We got about few hours to do this and I can change the rules. So, one point per answer, one good answer, so we could put simple. That's it. Maximum of 10 points, and then we do correction and we go from there.

Question one. There's only one answer per question. It's one point, one answer, you cannot write ABCD. That's multiple answer. You have one answer per question. You get no points. You get no points. You can try it out. Fill one with multiple and we'll see. Yeah. Let's do an experiment.

Question one. So, which ccTLD is the most recent to be signed using DNSSEC? Which is the most recent signed using DNSSEC, to be as something French. So, Guinea-Bissau AX [inaudible] island, Haiti, Bhutan, Italy. So, which one was most recently signed? One answer, A, B, C, or D.

Question two. In what year did Puerto Rico and Brazil first sign their TLD? So, we had a presentation this morning on this. A, 2007, 2009, 2011, 2013.

Question three. According to APNIC stats, which country has the highest per capita deployment of user capable of DNSSEC, including ECDSE and RSE validation? So, which country has the highest per capita deployment of DNSSEC? A, Sweden, B, Kiribati, C, Netherlands, G, Greenland. Oh, D. Sorry. I'm just dyslexic a little bit, as you can see. Just a little bit.

Question four. According to RFC 4509, how should validating resolver and [until] the presence of both SHA1 and SHA256 digest in a DS or RSSAC? A, you should in your both DS. You should in your SHA1 DS. You should in your SHA256 DS, or you

must support both DS. Question four. So, we talked about this at the DNS-OARC.

Question five. I've learned [inaudible] by now but. Question five. In what year was the first KSK signing ceremony held by ICANN for the zone in Culpepper, Virginia, USA. So, 2000. We don't know. 2004. 2007. C, 2010. D, 2012.

Question six. This is a tricky one. What does the H stand for in the software package SoftHSM? Homogenized, A. B, heuristic. C, harden. D, hardware. E, shut up. But it only applies for one question. It's going to go on after the next one.

Question seven. Which TLD has the highest recorded number of DS record since the root was signed? So, the most DS ever. .se, A. B, .nl. C, .br. D, .us. E, .bank. This came from Roy's slides somewhere. It is. That was the slide deck open from DNS-OARC, so that's where it came from.

Question eight. Which of the following DNSSEC-related terms is not an acronym? DANE, B, ENAM, C, [inaudible], D, DNS. That's Jake [Zach] help me do the quiz and there's a lot of quizzes, so that's an interesting one. Only one good answer. A or B or C or D.

Question nine. What percentage of all TLD in the root are signed? A, 97 to 100. B, 94 to 96. C, 90 to 94. D, 86 to 89. And E, 81 to 85.

And the last one. Which of the following TLD is not in the root zone? Pretty interesting, eh? A, .aaa. B, .abc. C, .aco. D, .aeg. And E, .ant. I didn't even need to scroll down multiple pages to do this thing. I just went, "Whoa, this is interesting." That's was hard, actually. All right. All done?

Correction time. So, you pass your sheet. Make sure you put your name, pass it to your neighbor, and then we'll go through the correction. And I'm always right, so I'm the authoritative source of answer here.

So, question one. Bhutan, December 2, so most recent. Most recently signed. C.

Question two. 2007, Puerto Rico. Answer is A. I hope I'm not drawing anywhere, please. Eh? When they first signed the TLD, I don't know what they do with the DS. I don't care. That's when they signed it. If we look at the stats, there are Verisign named, DNSSEC. Next, next.

According to APNIC, which per capita, it's answer is D, Greenland with 77. Yeah, exactly. All three people have – yeah. One person doesn't have it, right?

Question four. So, we all learned that the DNSSEC workshop, you should…

UNIDENTIFIED MALE: [inaudible]

JACQUES LATOUR: So, the answer is B, you should in your SHA1 if it's there, which in turn makes a KSK like if you do a rollover invalid or something like that.

Question five. In 2010, we need a few easy answers to get a ticket for lunch, right? But that's too late. Yeah. What does the H stand for in the software package? Hardware. So it's soft hardware, unlike the [inaudible] router, this one you can probably flex.

Question seven. So, .us had the most DS record ever. I think it was ADS record, so you can find Roy's slides somewhere and it's there. Proven guaranteed. But there was eight with…

UNIDENTIFIED MALE: [inaudible]

JACQUES LATOUR: E-mail Roy. I am always right. This is the number of DS record in the root zone for a TLD. Which single TLD has the highest record number of DS since the root was signed?

UNIDENTIFIED MALE: [inaudible]

JACQUES LATOUR: Oh. So, they had two KSK with four DS of different type per key.

UNIDENTIFIED MALE: He means he's the quiz narrator and the quiz narrator is always right.

JACQUES LATOUR: Exactly. Sorry. That's my bad. NUM, NUM. It's not ENUM you don't say D-A-N-E. You don't say E-N-U-M. I'm right. No. [KIM], NUM, DANE. DNS. It's right there. Oh, DNS. Okay, all right, I missed that. Next.

So, percentage, 90% still. Yeah. 90.6. But that falls between 90 and 94.

Last one. .ant, which is the only one I would know what it's. All right, so now we correct. A. C. All right, so get your sheet back with your scores, so whoever has five or more, raise your hand. Five or more. Now six or more. Six or more, just one hand. Seven, eight, nine. That's it. So, you're the grand winner.

RUSS MUNDY: Okay. Thanks, everybody. And we always have a fun time with that, and so we're now, we're going to have squeeze just a little bit because that was planned for before lunch. So, let me just

jump right quickly to Viktor and his presentation, and take it away Viktor. You have the clicker. Okay.

VIKTOR DUKHOVNI: All right. Is the mic over there? I think I'm more comfortable standing, if you don't mind. All right. We'll see how that goes.

All right. My name is Victor Dukhovni. I've been working in this space of e-mail security for some time now, since 2001. Some of you know me as a [postfix] maintainer, now also on the OpenSSL Team, and I've written a few RFCs related to this space. All right. Next slide.

So, I'm going to talk to you about a few things. The first thing I'm going to cover a little bit background for anybody who doesn't know about DANE. Yes, anybody who doesn't know about DANE, here I'm going to cover that briefly. I'm going to talk about what to do even if you're not implementing DANE yourself but so long as your zone is DNSSEC signed, you're potentially at risk of not receiving e-mail from people who do implement DANE. If your DNS lacks the appropriate hygiene, so I'll just cover the issues there.

I'm going to talk about how to implement DANE reliably. Some of the enthusiastic early adopters jumped in feet first but haven't really planned how to keep it working properly, so I'm

going to help you plan and monitor and automate DANE. And I'm going to talk a little bit about a survey that I'm running, which tracks DANE adoption and helps people who accidentally mess up their DANE to correct it by e-mailing them, all that kind of stuff.

And then I have an appendix. I have way too much material to talk about, not enough time, so please find a copy of the slide deck on the website for ICANN for this talk. There are many slides I will whiz through quickly or not cover at all. I have 50-some slides and 45 minutes, which is not going to happen.

All right. So, quick overview of e-mail security. E-mail is fully secure, right? We have the sender can use authenticated TLS to submit e-mail messages to his mail submission agent, your mail server, and the recipient can read his e-mail securely over TLS authenticated IMAP, and so e-mail is magically secure because both the sender and the recipient are okay, right?

Well, there's a little bit of a problem. The mail actually has to get from the sender to the recipient, and so this is goop in the middle that most users aren't aware of called MTAs or mail transfer agents that move mail around between organizations. And, of course, those are secure, too, miraculously, the e-mail just arrives from place A to place B, fully authenticated and encrypted, well not exactly.

And so I'm going to describe in a little bit more detail what actually happens in that step two between the sender and the recipient, which is the space where software like Postfix operates and where I focused my attention on security.

All right. So, what actually does happen between organizations when e-mail moves in the background from one visitor to another? We have something called STARTTLS or Opportunistic STARTTLS and that is that mail servers will try and determine whether the destination they're sending mail to support encryption of the e-mail while it's in transit.

And this technology is quite useful. It resists passive monitoring. If somebody is wiretapping your line but isn't doing anything to interfere with the communications, then your traffic remains confidential while one e-mail server sends to another.

But this technology is vulnerable to active attacks. There are a number of papers that you can read that show that these actually take place from time to time and in various places, and the active attacker who's willing to interfere with the traffic can do so via BGP hijacking, causing the traffic to let's say go through Ukraine unexpectedly for a few hours, even if you're communicating between one U.S. organization and another, or they can do so by doing DNS cache poisoning, if DNSSEC isn't in place, or if they're on path, they can just strip the STARTTLS

advertisement from one mail server to another, and then appear to the sender that the recipient is not capable of encryption, and the e-mail will be sent to Anyclear.

All right. Only despite all of that, Optimistic STARTTLS is an amazing success. Back in 2014 – I don't know if you can see on the left. That's when this graph starts in January 2014, there was about 25% of the traffic that Gmail was seeing was TLS encrypted. I remember even longer ago it was 5% or 10% when I was postmaster as Morgan Stanley way back.

But now, on the right even though it's a little hard to see how it fits into the scale, about 90% of the e-mail coming and going into Gmail is STARTTLS encrypted, so this Opportunistic TLS stuff is actually working pretty well as intended. And there's a link you can actually check that periodically and see how the statistics are changing.

But we can do better. I'd like to get [inaudible] to a state where at least for those people who wanted to, we can resist active attacks. We're not vulnerable to the BGP hijacks and the DNS cache poisoning and the like. So, to do that, while existing in an ecosystem that's largely still Opportunistic Optional STARTTLS, we need a signal that tells the sender this recipient can do security, and the signal needs to be downgrade resistant, otherwise the man in the middle will downgrade it and strip that

signal just like they do stripping STARTTLS. And the convenient signal mechanism in this case happens to be DNSSEC. It's by far the most suitable for doing this in e-mail. All right.

So, one message you should take away is that lots of people like to say, "Well, NTP uses TLS, that's just like http. Surely, all the things that work well for http will work well in e-mail. Why isn't e-mail more like http?"

And the answer is radically different. You can read about it at the link off the top of the slide, but mostly, the first thing that's radically different is indirection. E-mail determines where to send mail to based on MX records in DNS. Http doesn't do that and in order for e-mail security to work, you essentially have to trust the content of that MX record, so you're fundamentally relying on the DNS already for part of your security.

And the other thing is that in my view, at least, the web PKI, the certification authorities, the thousands of them that there are, are way too many to actually build any kind of secure system, especially because when there is an exception and you maybe don't trust a particular SEA, unlike with http, where you can click through and say you know what? I want to view that page anyway, it's not my bank. I don't give a damn. Show me the insecure page. With e-mail, there's no user to say show me the

insecure page, so you kind of have to trust as many CAs as you can lay your hands on, which is not exactly a good security goal.

Okay. So in comes DANE, and DANE is very well suited to SMTP. DANE stands for DNS-based Authentication of Named Entities. Quite a mouthful. I wish it were something a little bit simpler, but anyway. So, in SMTP, the way it's defined in the RFC 76 that Wes Hardaker and I put together, is that you indicate that you can secure or receive e-mail securely by publishing a record in DNS. A record in a DNSSEC-signed zone.

The record that you publish in DNS starts with a port number. For SMTP, it happens to be 25. It continues with the protocol tag and SMTP runs over TCP, so we have TCP there, and then it names the MX host, for which we're going to do security, and then it lists some parameters that indicate how to secure the traffic.

The presence of such a TLSA record for e-mail, as defined by the RFC, says I will absolutely always do STARTTLS. If you don't see STARTTLS from me, there must be an attack or a major operational error at least in my zone where I accidentally forgot to enable STARTTLS, even though I promised to.

So, this is important. This signal is conveyed over DNSSEC, so it's downgrade-resistant. If somebody tries to say, "No, no, there's no such record in the zone," because DNSSEC does

authenticated denial of existence, the validating resolver won't believe the attacker that there are no TLSA recorders. They'll see a failure and they'll move on to the next MX host.

So, the first thing is we have a contract to do STARTTLS but beyond that, we also have a contract to not only do STARTTLS, but to in fact have a certificate chain, which matches those TLSA records. That's what those public key SHA256 things are about. I'll talk about those a little bit more in a minute.

So, DANE authenticates domain control via DNSSEC, so you don't have extraneous CAs. Let's encrypt and, perhaps, you do trust many others you might not trust. This system is self-contained within the DNS both for secure publishing of the records and authenticating the [peer], and it's downgrade-resistant. Okay.

So, I promised that I will talk a little bit about how to get along with DANE. Domains are starting to implement it. Comcast – I don't know if the gentleman is still here – are doing DANE validation outbound. So, if you want to receive e-mail from Comcast, please pay attention here. If you want to receive e-mail from gmx.d or web.d or a number of other organizations, and even IETF.org now does DANE validation outbound, I believe, they certainly do it inbound, at least.

So, if you want to coexist with DANE, we need to understand the following. The DANE senders will not talk to MX hosts that for which the TLSA lookup fails. The TLSA lookup can return a valid record. It can prove that the record doesn't exist. Nonexistence is not a failure. Failure is that incorrect data is returned, bad signatures, something wrong about the DNS packet, or expired signatures or things of that sort, or simply not answering at all. If any of those things happen and especially if they happen across all of your MX hosts, you're not going to receive an e-mail from DANE-enabled senders, whose number I hope will continue to increase over the next two years.

So, for domains without TLSA record, if you're not doing DANE yourself, you at the very least need to make sure that the denial of existence of your TLSA records, the fact that they're not there, is delivered reliably. And this DANE is the first protocol for which this is important. Lots of other things have relied on DNSSEC to protect records that are there and make sure that the records that are there and are delivered are secure.

DANE needs records that aren't there to be proved to be correct, and so not surprisingly, early on, 2014 when I started doing work on DANE, there were lots of name servers that weren't getting this right because nobody cared, nobody relied on it. Now we do and over the last few years, I've managed to get most of the operators with the broken name servers to fix it, so it's a largely

solved problem, but you need to be aware of it and test your own domain.

So, the key thing for coexisting with DANE, therefore, is DNSSEC hygiene. You need to make sure that all of the things on this slide, EDNS(0), IP fragments, correct responses of no data versus MX domain and so on, all work for your domain. I'm not going to read every bullet on every slide here, take that away and test with your domain. Make sure it works correctly.

Monitoring is, of course, important. One thing I want to stress is that there have been the number of firewalls, metal boxes, things that think it's a good idea to protect your name server against hostile traffic by blocking queries for certain record types. So, if you see a query for an A record or MX record, that's surely fine, but if you see one of these weird queries for TLSA records, CA records, CDS records, new DNS record types, surely, that's a good thing to block. After all, the zone doesn't have those, so we can just block those queries, right?

Well, not such a good idea because blocking such queries certainly breaks DANE, your TLSA denial of existence won't work, but also breaks certificate issuance if they're CA records, or breaks your potentially someday your KSK rollovers, if you're blocking CDS records and so on. So, do not deploy firewalls that block DNS queries by record type. They're a bad idea, they

should not have been implemented, but if you have such a firewall, make sure to never turn that feature on.

And you can test, I'm showing a couple of examples with Digg, of queries you should make and they should return either no data or no such domain, not some kind of failure of DNSSEC validation or simply no response. They should not time out. They should return what you expect.

So, here's the DNSSEC checklist. I'm not going to read every detail on it. I've thrown in some gratuitous bullet points on the end about NSEC3 hygiene, which is not really related to DANE, but something that I just covered while running the survey. Isn't there some issues with NSEC3 and you can read the links and learn more about NSEC3 hygiene.

Here's a picture with DNSViz of one site that DANE-enabled domains won't be able to mail to. Techtrack.gov consistently after signing their zone does something to bump up the serial number after the SOA is signed. So, the SOA record never validates, and so the denial of existence never works. And you could actually, if you write some code, [write a Perl] script that kind of reverse engineered their signatures, and figured out that they're exactly off by one. If I roll their SOA serial number back by one, their signature validates, but they're always one higher than what it should be. Don't do that. So, be careful about, but if

they monitored, they would have noticed such a thing. All right. So, that was DNSSEC hygiene. Do practice it, even if you're not doing DANE.

Okay. Now I'm going to talk about where there are more slides with examples of DNSSEC bad hygiene in the appendix, if you're curious.

Adopting DANE. So, let's suppose that you're very excited, like I am, and you want to adopt DANE, and so the first thing is you're not going to get very far without DNSSEC. You'll need to sign your zone. And so once you've signed your zone, that's the hard part. Managing DNSSEC is harder than managing DANE. Managing DANE, I believe you'll be, believe me, by the end of this talk is easy.

The tricky part that I'll show you how to do correctly is coordinating your TLSA records and your certificate chain. DANE requires in principle that you make changes when you rotate your keys for your TLS SMTP server in two places. One is you need to deploy certificates need private keys and the other is that you might need to update your TLSA records in DNS. And juggling these two sets of related changes looks hard but let's make it easy.

Okay. So, the first thing, though, is I'll talk about outbound DANE. Suppose you're not ready to sign your zone or haven't

figured out operationally yet or trained your team how to do the TLSA record maintenance. You can still enable DANE for checking mail that you're sending to sites that have implemented DANE. For this, you just need a DNSSEC validating resolver, ideally one that runs right on the mail server. Mail servers are perfectly capable machines. They're not laptops, they're not phones, they're machines and data centers that can run a local resolver. Having one improves your performance and also lets you leverage it for the validation and many of the MTAs that are implementing DANE don't do DNSSEC validation themselves. They just rely on their local resolver to tell them whether the record is validated or not, so you want to have a secure path to it, nothing more secure than running on the same machine and just speaking to the loopback interface.

So, the DANE-enabled MTAs that you can easily get off the shelf or postfix XM. Cloudmark has an offering to providers that sort of scales larger than these smaller MTAs like postfix and XM, and there are more coming. I'm aware of some. I don't know that I should specifically mention people who haven't released yet but I know that various other vendors are working on this.

And then once you have such an MTA that can do DANE, you have a local validating resolver, look up the documentation, turn it on, it largely works. Just like with Comcast and their negative trust anchors, there are occasionally needs to do negative DANE

anchors, whatever, domains that you want to exclude because they mess up.

There's a GitHub page, which somebody volunteered to maintain a partial list of domains with DANE failures that they're finding occasionally contribute to it. Feel free to contribute to it if you also find other domains that can be validated to be broken. There's not that many but you can have an exception list from time to time.

Okay, so the focus of the talk is really inbound DANE. So, how do you implement and manage TLSA records? Well, the first thing is you're not going to get very far unless your MTA supports STARTTLS. So, definitely make sure that your MTA supports STARTTLS reliably.

Then you need your MX records to be DNSSEC signed. If the indirection from your domain to whoever it is that manages its e-mail is insecure, then the man in the middle can redirect your e-mail to some MX host of their choice, and you're not going to get mail security unless your own domain is signed.

But then it gets a little bit more interesting because once your MX record is signed, if your e-mail is hosted by some provider who's not you and they're operating your mail servers for you, then you're done. It's up to the provider now to implement the

rest of the DANE inbound story, the TLSA records belong with the MX host, not with the hosted domain.

So, if you're a customer of a hosting provider who manages a lot of domains, your job is easy. Just get your domain signed, and managing all the mail stuff and the certificate rotation, all the security bits, are done by the outsource provider.

On the other hand, if you're managing your own mail host, then essentially you're a provider and so the rest of the stock we'll talk about how to be a provider.

So, the provider gets to publish two kinds, one of two kinds of TLSA records. The specification defines 24 different types of TLSA records, 22 of them are a bad idea. Don't use those. There are only two that you should ever think remotely consider publishing. The first one, 311, has a certificate usage code DANE EE. The E stands for end entity, really meaning the server. You're publishing the SHA256 digest of the public key in the server certificate. It's a kind of a pin, you're committing that my server will have a particular public key.

The alternative is that you could say you know what? I'm not sure what certificates my server will necessarily have, but I'm sure it'll be issued by a particular certification authority, particular trust anchor, and so I'm publishing the SHA256 hash off the public key of whoever it is that's going to issue my server

certificates. Of course, you can publish both and we'll see in fact there's a good use case for publishing both.

So, TLSA record has either 211 or a 311 or it has something else if you're not listening to me and you're doing the wrong thing. So, the rest of the record is a hash value. This is where you get to say what in fact is the digest of your public key. So, for IETF.org, at the bottom of the slide, you'll see that their hash starts with OC72 and ends with ZRC72. Ends with D3D6 this last time I looked. Maybe they rotated it in the meantime.

Okay, so how do we manage TLSA records? The thing that you have to do is to get your TLSA record into the DNS before the certificate change is deployed. The DNS has extensive caching. This caching between the resolver and the authoritative servers and even between authoritative servers, there's caching between slave servers and the master zone, in many cases.

And so there's a significant lag between when you make your changes in the authoritative server, updating your TLSA records, and when clients start to see those rather than whatever you had before. And therefore, the TLSA records that say I have such and such a certificate, have to be in place for some time before the certificate is actually activated on the mail server, so that by the time it's activated, the clients who are going to see that certificate will look at the TLSA record, they'll see an

appropriately recent one and say, “Aha, this server certificate is good.”

So, but fortunately, we don’t have an impossible problem of exactly synchronizing the updates to the TLSA record and the certificate. Impossible because of DNS caching. We can publish multiple TLSA records, some that work now, and some that will work in the future. And the verifier who’s consuming the TLSA records doesn’t have to have every TLSA record match. It suffices for just one TLSA record to match this record of chain.

So, what we do is we publish keys well in advance and we make sure that always at least one of those TLSA records will match either the present or the soon-to-be present key. There are two ways to do this. The one I really recommend is the first one, where you publish two TLSA records, both of them identity, both of them match the server public key, one matching the current public key, and the other one matching the next public key that you’re going to have next time you roll your certificate.

And the other model that I can recommend is that you publish two keys, one for your server and the other for the issuing CA. And I’ll explain what the advantages of each model are. So, let’s go. So, current plus next, this is both matching the server public key, one now, one in the future. So, what do you do to make this, to stay sane while doing this?

So, let's say you want to rotate your keys hypothetically every 90 days but you don't want to have to make lots and lots of coordinated sequenced steps so that you have to three or four times during that 90 days, remember to do something and then wait for it to finish and then do the next thing and so on. You just want to have one change cycle every 90 days.

In order to keep it simple, you generate the next key on the same day that you deploy the current key. So, let's suppose this is Day zero, I've deployed the current key, but I also generate the key that I'm going to use 90 days from now. I can squirrel away the private key, keep it offline, whatever, so long as I can get my hands on the public key value, I can immediately deploy both the new certificate change with the new key and publish the TLSA record that will match the key I'm going to have 90 days from now.

So, in the slide you see two TLSA records, both 311, one matches the current public key I just turned on, and the other one matches the public key I'm going to have 90 days from now. Then weeks, months, whatever, years later, I encourage you to do more often rather than less, because it's easier to debug operational processes that happen quickly, that happen frequently, rather than things you only do once every three to five years or something, you're never going to figure out how you're doing it right or wrong.

So, whatever time, when it comes time to the next maintenance window, you obtain a certificate for the pregenerated key. This is easy. The keys, you just sign a CSR, you can in fact have signed the CSR the 90 days ago time, and then just go to your CA, obtain a new certificate, and but before you even think of obtaining the certificate and deploying it, make sure the TLSA record matching the key you're going to use is already in place. It should have been in place for a long time. And this check will keep you out of trouble. You're never going to deploy certificate whose TLSA record isn't already there and hasn't been there for a long time already.

And then once you do that, you've deployed the new certificate, you'll again go back to the first step. Generate the next key again. And with this kind of process, you can make sure that you're not scrambling at the last minute to update your DNS with new TLSA records because you got yourself a new certificate chain.

So, this works well and it can work even with let's encrypt to automate your key rollover for you and certificate generation. If you use the CSR option, which allows you to specify your own key rather than have them generate one for you. Okay.

So, that was the first model. The second model is one where instead of having the current key for the server and the next key

for the server, you have the current key for the server and the current key for your CA.

In this model, we publish both and if either of them verifies you're good, and then when it's time to do a rollover, so long as the CA stays the same, it's okay if you deploy a new server certificate with a new key that doesn't match the 311 record, your 211 record will continue to work as it's still the same CA. And then you deploy it, you test it, you make sure that the 211 record works correctly, the 311 record will not match, but once you're happy with the certificate and everything is fine, you can then rotate the 311 record after deploying the certificate.

If you don't like planning ahead and you like to be reactive and deploy stuff and then make changes that correspond to it, this model allows you to be a procrastinator and deploy a certificate at the last minute and then update the 311 record. On the other hand, if at some point your CA decides to use a new certificate and a new public key, then and only then you would make sure that you obtain from the new CA, their new public key, a certificate with the same key you're using before. So, again, you maintain continuity, the 211 record will change, but the 311 will stay the same. And so you can leapfrog periodically rotating 211, keeping 311 the same, or periodically keeping 211 the same and changing the 311, and so long as you're disciplined about this, again, the process is fairly simple. So that's that.

Of course, any process you're going to do regularly is subject to human error if people are reading SRS script of steps they have to manually perform, as much as possible if you're going to implement DANE, especially if you're a service provider, you don't want to have your team doing this by hand. You really should have some automation in place for doing this.

Unfortunately, I can't tell you how to do the certificate deployment because how they're stored and how they're deployed depends on an MTA, and I can't tell you how to do the DNS updates because, unfortunately, different authoritative server back ends have different mechanisms for inserting data. And this update maybe and we're trying to put together some scripts. I might post about that at some point, but unfortunately, automation will continue to be somewhat site dependent for a while.

If you are deploying TLSA records in DANE, please have working context in WHOIS or in the responsible e-mail address in your SOA record or at least a working postmaster address. Occasionally, other people will be unable to send you mail and they'll want to contact you and let you know this. And I've struggled occasionally with finding working contacts for people who make mistakes.

Monitor your DANE. I'm currently monitoring it but I'm not the world's monitoring system forever, so please do monitor your own deployment. Make sure that it's working correctly.

So, here's a list of best practices for DANE. Don't use wildcard certs. Don't roll all of your MX hosts at the same time. All kinds of good stuff like that.

All right. Okay. So, that was how to deploy DANE. There is some DANE software. I mentioned Postfix, Exim, Cloudmark. If you're running your own site for your own personal domain, I can very strongly recommended mailinabox.email. They give you a turnkey appliance that does the DNSSEC and the DANE and the certificate rollover with let's encrypt fully integrated. You just turn it on, delegate it, box.example.com to it if you want example.com, and it just takes care of all the details, including antispam and antivirus and excellent software.

If you're a developer, you can use OpenSSL 110, to which I contributed the DANE validation library, and there's some documentation that you can read about how to do that.

If you want to use GnuTLS, you can do it but there are some caveats, so I don't strongly recommend it. Again, touch with me if you want to develop something that does DANE over GnuTLS, there are some warts.

To maintain or planning to write DANE-related software, get in touch. We can exchange ideas, comments, that sort of thing.

Here's a list of DANE tools. There's a site where you can test your own domain on the web. There's a user list to which I post monthly statistics. Paul Wouters wrote this hash-slinger thing that lets you generate TLSA records and check them. Phil Pennock of XM Development Team wrote something called SMTP Dane and Go, it's a validator that you can use to test your domain. I wrote one in [inaudible] and everybody's afraid of [inaudible], so probably nobody will use mine. Or you can even do it yourself with a shell script. I have an example in the appendix of how to use the OpenSSL client command to test your MTAs correctness of the TLS records.

Okay, and now I have about two minutes to cover my survey, so I'm doing okay. So, I operate a survey that scans many DNSSEC domains as I can lay my hands on. I pull in about 200 million domains worth and off those, 5 million or so are currently DNSSEC signed and interestingly enough, 180,000 or so domains do actually DANE SMTP and they cover millions of users, Comcast, GMX, etc.

Here's some numbers on MX hosts and there's a small percentage of people with problems. A hundred or so domains with the DNS hygiene problems, around 150 with wrong TLSA

records. All my attempts to notify them of trouble have not yet succeeded.

This is growth over time, if we go back to 2016. We had about 1,800 different organizations with DANE TLSA MX hosts. Now we have around 3,200. So that continues to grow fairly steadily.

Okay. Here are some domains, registrar.br, who spoke to us, Comcast is on there, GMX is on there, Domeneshop has a very large deployment, and some obvious .orgs like IETF.org and security minded things like torproject.org, and so on, have Dane. There are, of course, a lot more but these are prominent ones.

This is something I want to as an appeal, these are providers who host a very large number of DNSSEC-signed domains, and host the mail host for them, but the mail host does not have DNSSEC or has DNSSEC but doesn't have DANE. The world would be a much better place if these particular providers, and if you are one of these providers or you know people there who you can influence and twist their arm – so ovh.net has 1.4 million MX records point at them, they're signed but OVH MX host isn't.

One.com, similarly, a very large population of domains that they host mail for that could go live with DANE but haven't yet. Interestingly enough, Google is slowly moving from google.com to googlemail.com and I understand that doing DNSSEC for google.com is a very big and hairy problem, but doing DNSSEC

for googlemail.com should be easier, so I'm becoming a little bit optimistic that maybe the 335,000 domains hosted by googlemail.com will at some point over the next year or two maybe, if all goes well and they get excited about this, go to DANE, and there are various other providers who I'd like to see implement DANE.

So, I'd like some help. Certainly, I'd like to be able to monitor more of the Internet, so if any ccTLDs are willing to share data with zone data with me on a research basis, I'll see sign whatever NDAs you want to throw at me, but I'd like to have more data. I'd like people to remediate the remaining name servers, the hundred or so, that still don't do denial of existence correctly or simply block TLSA lookups.

Please enable DANE outbound, especially if you're a very large sender, because the people who screw up their TLSA records will notice that they're doing it if there's pressure from large providers all of a sudden not sending them mail, then they'll quickly notice the problem and I won't have to be the one notifying them.

Please enable DNSSEC and DANE, as I mentioned, if you're OVH or one.com or any one of these large providers, go ahead and do that. And of course, finally, I'd like to appeal to people like GoDaddy, who has last I heard about 37 million or more

domains that they host, both the DNS and mail and so on, very little of that is signed. Certainly, there's no TLSA records there yet, but it would be fantastic step forward if GoDaddy were willing to work with me to get DANE done for the tens of millions of domains that they have.

Appendix. I'm not talking about the appendix. Yes. Any questions or am I out of time for that, too?

RUSS MUNDY: I'm sorry, Viktor. Thank you very, very much for the presentation. Will, if you can stick around like in the hall afterwards, then people can grab you and ask questions as they have them. Thank you, Viktor, very much. And Warren, if you could come join us, too. Here we go. Matt needs the clicker.

Pick a seat. Cathy has the slides. Yeah, Matt's slides. He's the only one who has slides here.

So, now, our perhaps most exciting panel of the day and I'm certain that everyone in this room is aware that the KSK roll plans that were originally put in place have been deferred, and there's a set of activities underway at this point, 15 minutes [inaudible], and then Matt's going to give us a presentation about that, and then we'll have a short at the table discussion

from Joe of Comcast, Jacques from CIRA, and Warren from Google.

So, with that, Matt. We have a timer here that'll help you keep track of things.

MATT LARSON: Thanks, Russ. Hi, everybody. Thanks for asking me. This is the same slide deck I'm going to present in two hours in Ballroom A. We have a session on the main agenda about this topic. So, I'm going to go really fast here. I'm also aware that I've shown portions of these slides elsewhere that at least some folks here have probably seen. So, in the interest of leaving time for the panel and discussion, I'm going to go through these very quickly, and I guess I'll just say if you like them or anything's missing, you can come back and hear me do it again, Ballroom A, 4:15.

Let me give a very fast recap of how we got to today. In September last year, we postponed the KSK roll after analyzing the RFC 1145 trust anchor report data. The first analysis was done by Duane Wessels at Verisign and then ICANN OCTO repeated the analysis, and basically, we found various percentages, depending on you slice and dice the data, of more resolvers reporting only the old key KSK 2010, namely the current key, and that then we didn't really know what to anticipate but high single percentages seemed higher than what

we were comfortable with, and more importantly, we didn't know why we had that value, so we wanted to investigate.

So, we did investigate and someone in this room did the investigation, and we attempted to contact 500 resolvers who had reported only the old key in September 2017, and we found, not surprisingly, that tracking down operators based on just the IP is hard. They either can't find them or they won't talk to you or various other things.

Of the 20% we could contact, it looked like 60% of them were in ranges known to host dynamic IPs. So, in other words, those are VMs or containers or ephemeral instances that come and go. And the other thing about 8145 reports is they're regular DNS queries, so they're subject to forwarding.

We've also discovered other weird things like implementations that send 8145 reports, even if they're not validating, so they might have the old key, but who cares? They're not doing validation.

So, not that we really expected it, it would have been ideal to find one or two root causes and then we could adjust our messaging appropriately and maybe contact vendors if there were vendor issues, but that's not what we found and there was no obvious path forward then.

So, we decided to ask the community for guidance on how to proceed, and this happened in late December, early January, late December was the announcement that we were going to look for feedback on the KSK-rollover@ICANN.org list, which I would encourage everyone to please subscribe to stay up to date on the project.

So, there was agreement that there really wasn't a way to accurately measure the number of users is what, if we go way back to the design team report from a couple of years ago. That's what they suggested was a guideline for assessing whether or not the KSK roll should proceed and after they proceeded, if it needed to be rolled back because ultimately its users that are affected and the discussion led to the conclusion that it was really hard to accurately measure that.

There was hope that better measurements would be available in the future and then sort of buried the lead here, the third bullet, the consensus was that ICANN should roll the key and, of course, keep doing the outreach that we've been doing.

So, we published on February 1st a draft plan, emphasis on draft, that calls for rolling the KSK exactly when you're delayed on October 11th, 2018. No specific measurable criteria emerged during the community discussion and we're going to continue our outreach and we're going to publish the 8145 trust anchor

data periodically so that people can see what's happening with that.

Most importantly, though, there's a public comment period open on the draft plan. There's a little time left that closes in a couple of weeks, April 2nd. We really would encourage everybody to please comment because ho we proceed depends on the public comments. The plan is draft for a reason, it's draft to get the community's feedback.

So, here's a quick recap of where we are and where we're headed. Mid-April, we'll publish the staff report, which is required on every public comment, and the revised plan. Now, if the revised plan still has the date of October 11th, then we'll proceed as the rest of the slide indicates, in a Board Workshop on May 10th, around there midway, the board's going to ask, not only SSAC, but also RSSAC. I don't have the slide updated, to review and comment on the plan. There will be another session. We'll talk about this again in Panama in ICANN62. Our hope is by August 1st to hear back from SSAC and RSSAC. Then we publish the final plan on mid-August but proceeding will be contingent on a board decision.

The ICANN Board actually meets six times a year, if you're not aware of the timetable, at every ICANN meeting, so three times at the ICANN meetings, and then three times in between at

events that are called Board Workshops. So, there's a Board Workshop in September and then the request to be that the board passed a resolution authorizing the ICANN Org to roll the key, and then that would be October 11th.

Let me get to my data. We have slowly added data, RFC 8145 data, that we've been getting from the different root servers. Actually, this slide is also now up to date based on recent developments. We now have data from, I believe, it's 12 root server letters, so there's only one who's not participating. And we're getting these statistics based on Duane Wessels' excellent plugin to DNSCAP, so the operators are DNSCAP and the DNSCAP every 60 seconds reports via a DNS query, very cleverly, what trust anchor reports it's seen and from which IP addresses. We are collecting all of those and here then is my main graph.

This graph is maybe a little difficult to read it first because note, there are two Y axes. The red and the green lines, you read the left Y axis. That is number of source IP addresses, unique source IP addresses per day that are reporting to us about 8145 data. So, the green line is the total number of sources reporting, so you can see it's on the order now of 50,000 unique sources a day, and then the red line is those who are reporting only KSK 2010, so they missed the boat somehow.

And if you do the math, you get the black line, which is the percentage that the red line is of the green line. And right now, you can see we're hovering around 20%, which is higher than it was back when we first started looking at this data in September, and as of yet, we do not know why. We're in the process of trying to figure that out but you have to at some point stop making slides and come to the ICANN meeting and present them, which is where we are now.

So, you'll notice there's a big spike in mid-January. What's that? Well, we're pretty sure that it corresponds to an unbound release that was itself security fix, so the suspicion is that people were motivated to upgrade because it was a security-related fix. That still doesn't expect why there's not a drop-off after 30 days because even if these are new containers, let's say you bring up an old unbound that still has the old key after 30 days, the RFC 5011 process should kick in and it should get the new key. So, that strongly suggests to me that we're dealing with ephemeral VMs or containers here that never run long enough to have the RFC 5011 process complete.

Here the user breakdown from all the root servers, you can see when we have data from aside from J root, the graphs are remarkably similar and Duane reports that we're not getting data from all J root instances, so that could be why the percentage on J root is relatively lower.

Here are unique IPs added over time. So, this is how many new unique IDs that we've never seen before we get each day. So, again, you can see that the spike here, we're getting more and more new IPs, so if you see the spike in mid-January, it looks like it sort of gets us to a steady state. One could imagine a bunch of new – so, something's updated to now do report 8145 data, but then you can imagine that whatever this thing is, let's say it's a container or something, you can imagine it being deployed at all kinds of different IP addresses, which would explain why we get all these new unique IP addresses per day.

Here then is the unique sources over time, so the green line shows the unique sources over time that have reported 8145 data to us, so what that means if you follow it all the way to the right, there's about 730,000 unique IPs since we've started collecting data on September 1st that have reported an 8145 data point to us, and the red line is the number who have reported at one point or another on the KSK 2010. This percentage is even worse. It's about a third. That's about 250,000 over 750,000.

I've sliced and diced this with /24s and same unique /24s per day and cumulative /24s over time. It's a lot of /24s relative to the number of IP addresses.

There's not a lot of difference between the number of sources that report – well, let me say this in another way. There appears to be very little evidence of upgrading actually happening, of somebody being in a state where they had KSK 2010 and then moving to a state where they have the new key.

You can see these graphs published on a weekly basis at that URL. There's one for each root server. And then this is my last slide. I do have clearance from ICANN Legal to publish the actual list of IP addresses, and I'm going to be doing that and it's going to be something similar to this. I think it's going to sorted, a list sorted in reverse order of frequency by ASNs and you'll be able to click through and see all of the IPs associated with a given ASN.

So, clearly, you can see there's some places we can reach out and contact and say what's going on and we haven't done that yet, as I say. There's only so much time before you have to stop what you're doing and go tell people about it at the ICANN meeting, so obviously, a phone call to ASN 55836 is in order to find out what's happening.

All right. Why don't I stop there with enough time to let the rest of the panel talk?

RUSS MUNDY: Okay. Thank you very much, Matt. We appreciate the presentation. Are other members on the panel – there's no slides, and we just want to get different perspectives. Jacques Latour is going to go first from the perspective of TLD.

JACQUES LATOUR: Well, TLD in a country and a concerned citizen, I guess. I've been following this closely and like I said in the comment in the KSK rollover discussion is it's virtually impossible to measure this. It's virtually impossible to do the cutover without some collateral damage, and in my view, if we want to maintain the perceived trust and the root key, we got to be able to do this in a confident manner that we're going to change the… We're doing it, we support it, and if it breaks, plan will fix it pretty fast because it's easy to fix. It's not days and weeks and months before somebody fixed this. It's either you update a trust anchor or you disable validation.

So, it's a quick fix on the human side to resolve this. It's going to have a big impact in some region. But in my view, the more we delay, the more uncertainty we put in this process, it's going to impact the overall trust of what we're trying to build here is a… Even though the key, the ceremony, and all the process to create this is impeccable, if we delay this longer and we put our uncertainty around it, it's going to erode the trust.

So we need to do it, there is collateral damage, it's easy to fix, and it's not long lived, and we should rotate the key next year, also, and then rotate every year as a standard process and automate the hell out of it. So that if we deem if one day we need to do an emergency key rollover, it's going to work. There's going to be no question about it.

And DANE, if we start using DANE inside large scale enterprise and it's the central CA, all of that depends on the root key, that's the root of all the thing we want to build, so I say we need to do it and just live with it. Make sure we PR the hell out of it to say this is how you fix it, either turn off or disable validation, but if somebody goes DNS in the search, it needs to come up with the resolution in Google and Bing and all the search engine to say this is what you need to do, but waiting longer is not good for the community.

RUSS MUNDY: Thanks, Jacques. Now, Joe, if you could give us your perspective from a large [it] scales ISP, please do.

JOE CROWE: While I agree that you want to be kind of "let's roll this every year on an operational level" that could run into at scale, could run into some issues where one, we don't want to turn off

DNSSEC validation at all, and when the first key rollover was supposed to happen, we validated and tested for months and made sure that we're 5011 compliant, we might not exactly use 5011, and rely on the resolver themselves, but if something were to fall through, we were there to be able to do it.

So, hopefully, at that point, if we had the automation there to roll every year and that's still in place and that process gets better and better, it would keep the burden off of the operators themselves to actually keep the resolvers up to date with keys but, I mean, the biggest key to success with any of this right now is test, test, test and actually keep your vendor software up to date and that's my biggest suggestion on this.

RUSS MUNDY: Thank you, Joe. Now if we could hear Warren's comments in general from the perspective of the large public resolver operation that is well-known to all of us. Warren.

WARREN KUMARI: So, yeah. Warren Kumari, Google. So, I don't really have a huge amount to add, other than I think it's fairly well-known that we run custom software, and we do not do RFC 5011. It's a lot of additional complexity, it's very good for automated systems, but what we do instead is we have a bunch of people who when the

new key is published, check it, import it manually into sort of a canary cluster, make sure that it works properly there, stage it, and then sort of do the key roll manually.

A number of us, not at Google, sort of here, have actually been talking about if is 5011 the right way to be doing key rolls and possibly something that allows for better faster key rolling might be a much better solution. But I will stop my comments here because when we get to question and answer time, I have some questions for Matt with his nice presentation with charts and graphs.

RUSS MUNDY: Okay. Thank you, Warren, and Joe, and Jacques, as well as Matt. And now it is question and answer time and since Warren did have the mic and said he had a question, I'll let him go first. Everybody else, be thinking of yours, raise your hand, and we'll get to you next.

WARREN KUMARI: Can we go back a couple of slides to around slide 11? I can steal the clicker and do it myself.

So, I'm probably just misunderstanding or not getting something, but if the thought was that maybe this is unbound being upgraded to fix this vulnerability, the bit that I don't get is

why would you see a bunch more unique sources? You think that they were upgraded from something that did not do RFC 8145 at all to something that did.

MATT LARSON: Yes.

WARREN KUMARI: Oh, okay. Well, now I figured it out. It's really obvious when you start asking the question in front of a room of people and like, "Oh, I think I know where this is going."

RUSS MUNDY: Okay. Thank you. Peter Koch.

PETER KOCH: Peter Koch, DENIC. Yeah, pain in your back, actually. Sorry. I have a couple of questions or more or less, first I think I have a remark in Matt's direction because I have heard rumors this week, which is a bad introduction, I know. Could you clarify what would happen if the rollover was postponed or could you actually clarify that if the rollover was postponed even further, the Internet would still not come to a halt.

MATT LARSON: I am not aware that postponing the rollover would cause the Internet to come to a halt. That is correct. It would not stop.

PETER KOCH: Well, I can maybe clarify the rumors so that some people might have had the perception that the rollover was in kind of urgent and it had to happen, so we have on the record that that is not the case. Thank you so much.

MATT LARSON: No. It's entirely possible that the community feedback could overwhelmingly say you should hold off until some other development and then we would revise the plan and, perhaps, have even later date.

PETER KOCH: Thank you.

WARREN KUMARI: I guess I was next. I think that the big concern here is when the key was initially signed, we said it would roll after five years, and for various reasons, transition, NTIA, etc., that needed to be pushed out a bit.

So, the only real risk or is or the main risk is a loss of confidence in the keying material over time. So, the sort of tradeoff I think

is, is it more risky to the reputation of DNSSEC as a good thing to be doing for us to roll it and potentially have some stuff break, or for us to not roll it and have people say this key is really old and starting to smell like an old fish?

So, where is the trade off? And I mean I think it's great that ICANN has the public comment period open and so the obvious thing that [inaudible] $5 under the table, ask people to please comment and provide feedback. The public consultation period is open. Please provide comment.

JOE: Hi there. Joe [inaudible]. Just to recast a couple of comments that have come through. I think it's a mistake to think of this as a one-sided risk assessment, risk equation, where there's a risk of rolling the key, so should we do it, because there's also risk in not rolling the key, and it's not just the – I mean, the numbers, the digits are not going to start smelling like a fish, but in any crypto system, the ability to replace a key kind of goes hand in hand with all your precautions for physical security. You cannot start physical security and declare that it's perfect because it never is.

There's no black and white. It's always a risk assessment, there's always shades of gray. So, we're balancing the risk of not having any operational experience of rolling this key, which by all

accounts is an important key, against what is quite likely to, I think, to be a very, very minor risk that a few unmaintained system will break briefly, and then if they're important, they'll get repaired.

So, to me, that risk assessment is quite clear. The risk of not rolling the key is that we never know how to roll the key, and if we have to do it in a hurry, it's going to be a much bigger mess than the potential fallout from rolling it now.

WARREN KUMARI: So, yeah. I guess I will largely agree with Joe but also point out that the if we have to ever roll the key in a hurry, it's a completely, completely different process to this process. Right? If we have a need to roll the key in a hurry, it's because we think that we've lost confidence at the key, in which case you can't use 5011 because you can't trust the old key anymore.

But yeah, I fully agree. What we'd originally hoped we could do was publish the DURZ, the Deliberately Unverifiable Root Zone, and then every month or two after that, roll to a new key, just so we could test it. And then after that, every year or two, roll to a new key just so that we could test it. But for various reasons, we weren't able to do that, and so yeah, the first one is going to be an entertaining role. We also feel bad for [Peter], who must be getting tired standing.

MATT LARSON: I just want to say, Warren, I'm not sure who your "we" is there. I mean, in terms of the root zone management partners that we're working on deploying this back in 2010, I mean, our intention was never to roll, as you described. I mean, I think there are some people in the community who advocated for frequent rolls, but that was never the intent. The DPS does say, I just want to comment on your earlier comment, as well, the practice statement for the KSK does say after five years, but it doesn't say after five years and before six years, or something like that. So, it simply says after five years.

UNIDENTIFIED MALE: Matt, could you just move your slides forward to that list of AS numbers? So, the top with 41,482 individual sources is AS 55836. The amount of visible DNSSEC validation in AS 55836 is basically zero. So, what it does say in a very, very real sense is that the [inaudible] coming in from RFC 8145 is severely warped in ways that we don't understand.

Now, I think it was responsible back in September last year to go "Whoop, we don't understand this." But in some ways, it's not our appetite for risk that is changed, our confidence in the RFC 8145 signal is eroding exponentially. This is really whacko data

that doesn't actually mirror validating resolvers, and if you're not validating, you don't care about a key roll.

MATT LARSON: Well, I will say validating resolvers with eyeballs behind them who are seeing Google ads.

UNIDENTIFIED MALE: Well, in my case, that's what I'm using but there is a broader picture of here it does reflect the larger environment, and if there is a pool of validation that doesn't happen with eyeballs, that's even stranger and more inexplicable. So, in some ways, what I'm saying is it's not that our appetite for risk has changed, but our confidence in the signal of 8145 is eroding daily. It just doesn't seem to be signal that can guide reasonable decisions.

We've been at this key roll for seven years and we could spend another seven years but quite frankly, I don't see the point, because at some point, we're going to get an event that's going to cause us to roll, and if that's the first time we've ever rolled and we don't have planning, all of us are in deep, deep problems.

I would encourage you, and will encourage you in feedback, to roll as planned. And I would also encourage you, because it's going to be the next conversation, to do this on a regular basis

and for the lack of any other regular basis, I would say 12 months, as a strawman and go let's debate that as the next debate, but I would certainly give you strong encouragement to roll, roll this year, and continue to do so every October. Thank you.

PETER KOCH: Peter Koch with a second question. While in the previous question, I conveyed third party confusion, this is now my own confusion. I heard people argue for regular rolls and one of the benefits that was conveyed was it would help with a working rollover mechanism in place just in case we need an emergency rollover.

My understanding, though, is that 5011 is incompatible with emergency rollover. Nobody can hear your nodding.

UNIDENTIFIED MALE: I think we all agree and Warren and I started thinking about getting beyond 5011. I think we need a new standard.

PETER KOCH: So, for the regular rolls, we need another explanation and another motivation.

UNIDENTIFIED MALE: No. I think we need to replace 5011 entirely, so that's [inaudible].

PETER KOCH: That's two different questions. I mean, like I'm wondering. So, assume that we need to communicate this to the resolver operator not in the room, maybe not the whole day working on this, what is the purpose of the regular rollover? Because we just come to the conclusion that emergency preparation is not what can motivate us.

MATT LARSON: So, with the disclaimer that I am not part of PTI, I'm part of the Office of the CTO, long-term, I think I would go out on a limb to say that we all, we the ICANN Org, would like to be in a position where we have a different setup and don't have to rely on the current methodology for a regular scheduled key roll. I personally would like to have standby keys in the apex, so and that means having the current keys, anything we would do to generate another key with the current infrastructure we have, which share fate with the current key, so almost any scenario you can imagine that would compromise the current key would compromise the standby key.

So, if we can come up with other ways to generate keys that don't share fate, then we can talk about having multiple keys in

the key set and being in a position to luxuriously roll on a dime without a problem.

Now, 5011 has its own issues with that in terms of size of the apex keyset, so I think there's work to do here both on the protocol and in terms of operational procedures, but I think we can see a path to a way to do an emergency roll, even a regularly scheduled roll, in an easier fashion.

PETER KOCH: Thank you.

FEDERICO NEVES: Hi. Federico Neves from NIC.br. Nice presentation, Matt. And I would like to agree with you and besides the fact that I already commented on the public key record. Going that directly, as you said, of having spare keys and publicly spare keys, we definitely need to think about an algorithm roll because we have a problem with having spare keys and the current algorithm and size of the keys that we are using, so this is probably the next thing that we need to think about using a technology that use pretty smaller public keys.

WARREN KUMARI: Okay. So, I was originally putting my hand up to respond to Peter's question. I mean, there are two different types of emergency key rolls. Right? There's the key roll where somebody accidentally publishes the private key on the front page of the New York Times, in which time, you panic and you need to redo this and 5011 will not work for you.

There's the other type of emergency key roll, where it becomes clear that the algorithms you are using are not nearly as strong as you thought. Right? I mean, some people might say this kind of is happening for SHA1, as an example, and you want to roll it and you want to roll it kind of now-ish, but a couple of weeks is okay or a couple of months is okay.

So, depending on what your level of emergency is, 5011 might or may not work. For the next comment, though, we've been talking about maybe having something different for future key rolls and not 5011. And for that, we could have standby keys by not publishing them at the zone apex. Right? It would be perfectly reasonable to publish extra keys as records as the DNS zone, but not actually currently signing anything, and not as the current DNSSEC and the keyset, but as an additional record and you just say, "I have seven keys, here is a list of them. Go look up those two names that point at the public keys." So, that way, we wouldn't need to do an algorithm roll, but we should probably do an algorithm roll for other reasons anyway.

RUSS MUNDY: [Benedict] is next.

[BENEDICT]: Hi, Matt. Can I say how glad I am that you're in charge of this? Just a reminder that shadow server can notify some of these network operators and we have an existing system that they want to receive alerts about such issues, even if that alert is your resolver is doing something funky as opposed to it's got the 2010 key and we're ready and willing to help. I'll just run the numbers from the slide we already have that 30% or 40% of your ASNs, we want to receive reports so we can help with the notification process, be glad to.

MATT LARSON: Fantastic. I'll take you up on that and I know I've traded e-mail with shadow server before. Another, I don't have this on the slide but another tactic is to purchase from the other direction, which is to find the heavy hitters and we have good data from APNIC on that, and to go down from the topic and verify that the people who really do have a lot of eyeballs looking at them are ready or that they're not doing DNSSEC, so that we can check them off and get to a point where we can say, well, we don't know maybe who's not ready but we know who is ready and

who is ready is a very comfortably large percentage. That's another direction we want to take, as well, so you could help us with that and we'd appreciate it.

RUSS MUNDY: Okay. I think we are about out of time. If there's any last burning questions… We'll, okay. Yes, go ahead.

UNIDENTIFIED MALE: [inaudible], ICANN fellow. I had a couple of maybe basic questions. What's the role of the old keys and the rollover? Are they used to validate the new keys? And if they are, what if one of them gets compromised and if it does, is there a mitigation plan available for that or not?

WARREN KUMARI: So, yeah. The way that the sort of current protocol for doing this, which is RFC 5011 works, is the old key signs the new key, and that way, if you believe the old key, you can believe the new key. Once you stop believing the old key, your signal, don't believe the old key anymore, so the old key gets revoked or no longer trustable.

However, any of these sorts of protocols have the problem that if the old key is suddenly compromised or you lose faith in it, you

can't use it anymore to sign and trust the new key, so there is another process, which I'm not sure if it's widely publicized. I think it would be great if it was, on what exactly happens if the key is properly compromised and there's evidence of it, and there's some little bits we've heard like there's immediately an emergency session to generate new KSKs, etc., but how exactly the new key is distributed, where it's published, how people should know that they can leave it, because an event like that I don't really think go to www.iana.org and trust it because the CA signed it is really a good answer.

So, there needs to be a well-documented and possibly tested process for if the key is actually compromised, what do you do then, other than panic.

RUSS MUNDY: With that, I think we have to close and leave the room, but I want to thank very much Matt Larson, Warren Kumari, Joe Crowe, and Jacques Latour for participating in the panel. And there's a special thanks we need to give to Kathy Schnitt, who has run this session all by herself and there's normally one or two other people here. You've done a fantastic job, Kathy, and you've also helped us organize. Thank you, thank you, thank you.

KATHY SCHNITT: Thank you too, Russ. [Jacques] is taking over as usual again.

RUSS MUNDY: Thank you. So, we look forward. We'll be putting our program callout for the next ICANN62 probably in a month or so, and we'd love to hear more from more people, so think about your ideas for the next workshop and come see us at the next workshop. Thank you.

[END OF TRANSCRIPTION]