ABU DHABI – DNSSEC for Everybody: A Beginner's Guide
Sunday, October 29, 2017 – 15:15 to 16:45 GST
ICANN60 | Abu Dhabi, United Arab Emirates

UNIDENTIFIED MALE: October 29, 2017, DNSSEC for Everybody: A Beginner's Guide, in Capitol Suite 1, starting at 15:15.

WES HARDAKER: All right, welcome to the DNSSEC for Beginners. We're going to give a few more minutes for the rest of the slides to get set up. We're one of the distant rooms, so we'll give people a few minutes to get here. So go ahead and be patient for one minute, and we'll get started in a second.

One quick comment: you guys are welcome to come sit at the table if you want. There are microphones up there, and it makes it really easy to ask questions later if you want to do. You don't have to. The table spots are not reserved for anybody, so feel free to hop on up if that's more comfortable for you.

All right, we are almost ready. One more second.

All right, so welcome, everybody. This is DNSSEC for Everybody. It's a beginner's guide. The goal of this whole presentation is to explain how DNSSEC works. It's a complex topic, but we've come up with some ways to make it easy to understand, or at least we hope so. We're going to do some explanations through some visual tutorials as the day goes on. Next slide.

To start with, we're going to go through a little story. The story explains how communication works over a distance and what happens when communication goes bad. Next.

So the story is, this is Ugwina. She's a cavewoman. She lives in a cave on the edge of the Grand Canyon. Next.

This is Og. He also lives at the Grand Canyon but in a cave on the other side of the Grand Canyon, and it's a long way down the Grand Canyon and all the way back up the other side. Ugwina and Og are good friends, but they don't get to talk much because they live so far apart. On one of their rare visits, they notice that smoke is coming from Og's fire. Soon they start chatting back and forth across the canyon using smoke signals.

This is how the early days of the Internet were formed. People just communicated randomly with smoke signals. Until one day, the mischievous caveman Kaminsky moves in next door to Og and starts sending smoke signals too. The caveman "Kaminsky" is a phrase for Dave Kaminsky who created one of the DNS problems that allowed communication failures to happen, especially in DNS.

Now Ugwina is really confused. She doesn't know which of the smoke signals she sees to believe. This is the problem with communication that isn't authenticated or isn't secured: you never know which one to believe. Next.

So Ugwina sets off down the canyon to try and sort out the whole mess, and Ugwina and Og consult the wise village elders. Caveman Diffie thinks he might have a cunning idea. In a flash, he jumps up and runs into Og's cave. Right at the back, he finds a pile of strangely colored sand that has only ever been found in Og's cave. That's the only place it exists. With a skip, he rushes out and throws some of the sand into the fire and smoke turns a magnificent blue. Next.

Now Ugwina and Og can chat happily ever after again, safe in the knowledge that nobody can interfere with their conversation because Ugwina knows that she only has to watch the blue smoke. That's actually how all Internet protocol security works from an authentic point of view. It marks certain packets with a way that there's only one acceptable answer and that all the rest of them will be discarded.

That is essentially what we're going to talk about today: how that concept maps to DNS and how it maps to the DNS system so that we can believe answer and discard other ones.

Did any of you go to How the DNS Works? There are actually other tutorials that happened earlier today. I think there's another one tomorrow. But most people here are already familiar with the high level concepts of DNS which are that at the very top there's a root. It's where all resolvers start and it's where they ask their questions first. They go to the root resolver and ask, "Where do I go next?" and it chains all the way down.

Then there's a big tree underneath it. There are top level domains like country codes like .uk and .com. Then underneath that there are second level domains like co.uk and bigbank.com and nic.ma, for example. Today, we're going to talk a lot about bigbank.com, which is a fictitious bank that we've created for the discussion today. Next.

The important thing is that the resolver knows where the root zone is. That's the only thing. When a resolver first starts up and it has to answer questions for its clients, the only thing it really knows is where the root zone is. That's all it knows to get started.

Then it walks down that whole tree trying to find the answer for the client, be it your cell phone or your computer or whatever it was that asked the question, "How do I get to bigbank.com?" It has to know how to get there, and it starts at the root and goes down.

Every level refers [to] the [next] resolver to the next authoritative server in the chain until the question has finally been answered. Then the resolver caches that information for later use. It actually remembers stuff to make the next person that asks the question get a faster answer because it remembers it for a while. Next. Work? Okay, excellent. Now I have a pointer. Yay!

Another important thing is that when DNS was created, there was no security in it. It was created back in the days of the Internet where there was no malicious activity going on, so there was no security. The protocol doesn't have a way to tell you you're getting the right information. So names were very easily spoofed. You would listen to whatever answer you were given regardless of whether it was the right answer or the wrong answer.

And caches are easily poisoned. If you remember a second ago when I said that resolvers cache that information for future use, that also means they cache the bad answer if they happen to get the bad answer. Which would mean they would replay that same answer for minutes, hours, days, or whatever the lifetime of the cache was.

To further educate you in how this works, we're going now to do a silly demonstration. This is going to involve me actually taking my jacket off, and we're going to switch into some different roles. I’m going to have my volunteers stand up, and we’re going to show you how DNS resolution works in a skit.

UNIDENTIFIED FEMALE: I had to let the remote people know that we’re doing a skit because they can’t see it [unfortunately].

WES HARDAKER: I apologize to the remote people today. Normally, we have a remote camera that people can watch when we do this, but we don’t have one today. I’m going to hand this thing to you because you’re going to do most of the talking. I’ll hand it to you. Okay, so the way this works is let’s pretend I’m a user, Joe User specifically.

JOE USER: I want to go do some banking, so I’m going to go to my computer and I’m going to type into my web browser.

I’m going to go, “Hey, I want to go to www.bigbank.com” but my browser doesn’t know where it is. But it does know that my local ISP happens to have a resolver that can answer any question for me.

“Can you, Mr. ISP, tell me where www.bigbank.com is?”

MR. ISP: Well, I don't know, so I have to go ask some name servers. We will first start at the root of the DNS right there. "Hello, root of the DNS, can you tell me where www.bigbank.com is?"

ROOT OF THE DNS: "I'm afraid not, but I can direct you to .com. It is at 1.1.1.1."

MR. ISP: "Ah, thank you. I will go there and try next."

"Hello, .com, I would like to know where www.bigbank.com is."

.COM: "I don't know where www.bigbank.com is, but I do know where bigbank.com is. It is at 2.2.2.2."

MR. ISP: "Thank you, .com." Well, now I know where bigbank.com is, so I will go ask bigbank.com.

"Bigbank.com, do you know where www.bigbank.com is located?

BIGBANK.COM: “Well, hello, Mr. ISP. Nice to see you again. Actually, I do know where www.bigbank.com is located, and it is at 2.2.2.3. Here you go. There’s your answer.”

MR. ISP: Thank you very much, bigbank.com. I now come back to Joe User. “Hello, Joe User. I am back.”

JOE USER: “Wow, that took a long time.”

MR. ISP: “Well, it kind of did, but it really didn’t take that long – www.bigbank.com is located at 2.2.2.3.”

JOE USER: “All right, thank you very much.” Now my web browser pulls up my web page and I see the login interface and all my banking can be done.

WES HARDAKER: From there, we’re going to go on and talk about the next part. You actors need to stay there for a minute.

Ugwina, the resolver, is chatting with Og, the server. So this is sort of how it works. Ugwina was really the resolver in this case, and she’s listening to Og, the server. Ugwina, the resolver, is confused. She doesn’t know who the real Og is. You remember this before. And Ugwina, the resolver, can verify when the real Og sends a message because of the blue smoke.

Now let’s take another example of the same thing. So we’re going to go through the whole scenario, but we’re going to watch what happens when bad things happen. We’re going to go through the same process again, and we’re going to see what changes.

JOE USER: I’m going to go to my ISP and say, “I need to do some banking today and make some deposits. Can you tell me where www.bigbank.com is?”

MR. ISP: “Well, Mr. Joe User, I’d love to be able to and usually I cache the answer from before, but I’ve forgotten it so I have to start over again. So I will go to the root.”

“Hello, root. Can you tell me where www.bigbank.com is located?”

ROOT OF THE DNS: “I’m afraid not. I’m still not smart enough. But I can tell you where .com is. It is at 1.1.1.1.”

MR. ISP: “Ah, thank you, root. I will go ask .com.”

“Can you tell me where www.bigbank.com is located?”

.COM: “No, I don’t know where www.bigbank.com is located, but I do know where bigbank.com is located. It is located at 2.2.2.2.”

MR. ISP: “Thank you, .com.”

“Hello, bigbank.com. I would like to know where www.bigbank.com is located. Could you tell me?”

“BIGBANK.COM”: “I certainly can. You can find www.bigbank.com at 6.6.6.6.”

MR. ISP: “Ah, thank you. Thank you very much.”

“Hello, Joe User. I am back - www.bigbank.com is at 6.6.6.6.”

JOE USER: “Oh, thank you very much.” Typety-typety-type. “Where did all my money go?!”

WES HARDAKER: Malicious people on the Internet can answer with the wrong answer sometimes, and the problem is that what you just saw is that the resolver will believe the first person that talks back to him. If the evil attacker can answer quickly, he’s going to win every single time. Not only that, as I said before, that information might be cached for a long time.

DNSSEC improves on this. DNSSEC is designed to authenticate the packets that they get, the answer that you get so that you know when they’re wrong and you can only believe the good one. So our final skit in Act 3 will be what happens when DNSSEC actually works.

JOE USER: I’m going to ask the same question about www.bigbank.com. I need to do more banking and find out where my money went.

MR. ISP: “Well, thank you, Joe User. I will go find an answer to your question.” But if everyone recalls, we have now done DNSSEC signing of the zone of root, .com, and bigbank. So everyone along the line knows who they are.

I know, again, where the root is located as the ISP, so I will go ask the root again. “Hello, root, could you tell me where www.bigbank.com is located?”

ROOT OF THE DNS: “Hi, ISP. I’m afraid I can’t, but I can direct you to .com. It is located at 1.1.1.1. But before I do that, I think you probably should check my signature.”

MR. ISP: “Well, thank you very much.” I see the signature checks off, and so I will go visit .com.

“Hello, .com, could you tell me where www.bigbank.com is located?”

.COM: “I can’t tell you where you www.bigbank.com is located, but I do know where bigbank.com is located. It is located at 2.2.2.2. But before I give you the answer, let me just sign that for you.” Perfect. Now it’s signed.

MR. ISP: “Thank you, .com.” Now let me check. Yes, I agree that .com and root are the right places and they know each other. Now I will go to bigbank.com and ask where is www.bigbank.com.

“BIGBANK.COM”: “I can help you with that. It’s at 6.6.6.6.”

MR. ISP: Let me look. There is no signature on here. “Out! Out! Out! On the floor!” Let me try again.

“Bigbank.com, could you please tell me where www.bigbank.com is located?”

BIGBANK.COM: “With pleasure, ISP. Before I do that, let me go ahead and sign my response. It is at 2.2.2.3.”

MR. ISP: Well, I will look. Yes, it validates. “Thank you very much, bigbank.com.” Now I know. I have checked the signatures all the way to the root, and 2.2.2.3 is the right answer to present to Joe User for where www.bigbank.com is located.

JOE USER: "Oh, thank you so much." Now I can go open a new bank account since my last one was compromised.

WES HARDAKER: So you can see how DNSSEC actually helps us out here. It lets us know exactly when things go right and when they go wrong.

Can we give my actors a big round of applause real quick please? Thank you.

So specifically, the most important thing is that the root has a key associated with it that everybody knows. So that's where they can start. Everything else beyond that in the rest of the chain down is all signed and linked together securely so that as long as you start at the root, anywhere else in the tree that has been signed you end up at the right place. And in the end, you know whether you're getting to the right bigbank.com or whether somebody is lying to you and giving you the wrong answer.

One second while I take off my skit uniform. There we go.

So DNSSEC is solution to this whole communication problem. It's what allows you to believe, as I've said, that the answer is authentic. It uses digital signatures to assure that the information is correct and it came from the right place. The keys in the signatures are used to verify the information, and all of that is store in the DNS as well. So the keys for the next layer down are stored in the layer above it. Because it's signed, you know that you're getting the right keys for the next layer down. Since the DNS itself is just a lookup system, you can always look up the keys for wherever you need to get after that.

The only thing that a resolver needs to start with is the root key. As long as it knows the root key, it can find anything else anywhere down in the chain of trust. It builds this chain where every level, the next key is signed by the current level until the chain all the way down is complete.

Back to that same diagram that we looked at a second ago. It's like having checkboxes on everything where you can check and make sure that the checkboxes are correct all the way down. If you get the wrong answer, you'll know it.

We already did the skit and the play. The slides are in kind of – we need to talk about this in the future for the order of the slides. This one should go earlier.

Next up, I'm going to hand it back to the previous person that did the ISP which is Russ Mundy. He's going to talk about Examples of Why You Need to do DNSSEC and a Simple Guide to Deployment. Thank you, Russ.

RUSS MUNDY: Thank you. All right. Oh, good. It has forward and back arrows instead of this funny little round wheel that I always seem to push the wrong direction. Okay, let's see. It was working. Right, there we go. Okay.

Well, thank you, Wes. Folks, this is a short explanation of the DNSSEC functionality and the things that you need to think about and go through as you look at deploying DNS security.

But first, I want to talk about a little bit why are you even worried about DNS. You can recall from the skit that you could end up being sent to a fake bank site and end up giving them all the information they needed so they could go steal all your money. That's just one example, and it's an easy one to talk about. But the same thing can apply for things like electronic mail or your instant messaging [jabber] or things of that nature.

Why you should worry about that is most of the time you really want the place the you're trying to get to on the Internet to be the place you actually do get to. It's DNS that provides that translation mechanism from a name into a network address. That was the www.bigbank.com was the name, and the network address was 2.2.2.3 which is an IPv4 address. What happens is if you don't get to the right place, lots of different bad things can happen. Bad things can and, in fact, do happen.

One of the things that we've seen occur actually on the Internet where the DNS hijacking is used is that passwords can be stolen. They may do the DNS hijack and set it up so that you still do get to the place that you want to get to, but you are going through the person who is doing the DNS hijack and he's just capturing everything you sent, including your passwords.

You can go and set up a man-in-the-middle attack where, again, you still get to the end place you intended to get to, but everything that you sent is collected and monitored by the man in the middle that's doing the attacking.

Awhile back, I went to look around on the Internet and just see how many different things that exist that I could find easily by just doing a search on the Internet. In fact, at that time there were multiple hijack tools out there in existence as open source that you could simply pick up and use.

In addition, I did find one university course online that the student lesson that the professor was teaching for that course was write the software to do a DNS hijack. When I looked through the course syllabus, there was nothing that I could see in the course syllabus that even said this isn't a nice thing to do to people. But there was at least one university out there that was teaching in their coursework how to hijack DNS.

What is it that DNSSEC does for you? As Wes said and we showed in the skit, it’s a way to use cryptographic mechanisms so that the name that’s loaded into the DNS can be verified by the resolver that is fetching the name for an end user. It can check those cryptographic mechanisms to make sure it’s correct and that it hasn’t been changed in flight. What that means is your packets and your connection should all get to the place that you intended for them to get to.

Here’s a simple little diagram that [has nice short little names]. We have Joe User down at the bottom, and he sends a query to his recursive name server. The recursive name server, I didn’t do all of the levels in this particular diagram but just to show one name server before the answer goes back to the user.

So the recursive name server asks the authoritative name server. In this case, it’s name server ab.org. In our skit, bigbank.com was the authoritative name server that was being asked. The bigbank.com or ab.org gives the answer of where www.-name is located back to the recursive name server. The recursive name server in turn sends it back to the user. Then after those interchanges, that’s when Joe User actually opens the Internet connection to the web server that he wants to get to.

One of the things that we did a while back was to set up a modified Firefox browser that does, internal to the browser, DNSSEC name checks, validation. We set up a specific website that will work very effectively in doing a DNS hijack. What you see here is the site that shows as the accurate [name] site with the accurate content when viewed with the DNSSEC validating browser. What you see on the bottom is the instrumented name [of] the website coming back and saying you're not doing DNSSEC, but in this case the website looks the same anyway except for here DNSSEC is off and DNSSEC checkmark is on up there. That's the only difference on this page when there is no attack in progress. Next slide, Kathy. I'm not getting my little clicker here working right.

Okay, so let's put Dr. Evil Hijacker in place. Remember, Joe User is still down here. The recursive resolver is up on the far right, and the authoritative name server is on the far left. The same query is sent by Joe User from his end machine to the recursive resolver, but Dr. Evil sees the query go out and immediately gives an answer. That, if you remember in our skit, Dr. Evil jumped in and ran over and handed the wrong answer to me. I then turned around and gave it back to Joe User. You can see it is a different IP address than the actual www.ab.org. So he goes to the wrong place, and Joe User doesn't even know he has gone to the wrong place.

You can see by this very simplistic network diagram that things on a topological basis, what's connected to what where, sometimes are quite a ways apart. And you can see that flow. There are lots of places where bad guys can get in the middle and intercept it.

In the meantime, the other query – the real query – comes back and just gets dropped on the floor because Joe User's machine has already got an answer. So he even forgot that he asked the question to begin with. Now when we go and insert DNSSEC in there, then DNSSEC will prevent Joe User from getting and taking the wrong information. In which case, he actually gets to the real web server. Again, DNSSEC validation stops the false answer going back to Joe User.

This is, again, what the website looks like when you view it with a DNSSEC validating [browser]. The validation is occurring on the end machine in this case because it is a web browser that's doing DNSSEC validation in the browser itself. Now you'll notice, this time there's a difference between the two web browsers' content.

This are screen shots taken of an actual demo where we did a live hijack of the website. What we actually did in this hijack was inserted a false piece of information into the stream. That was injected by the DNS hijack. Since there was no DNSSEC validation on the front lower right browser, it was a simple matter of giving them the answer that inserted that "Steve crocker (with strange capitalization) admits that DNSSEC won't solve world hunger."

How many different places are there in the whole process that can involve a DNS hijack? A lot. Each one of these is a query or a response that shows a place where a DNS hijack could occur. The first picture was a drawing from about six years ago now, and this is a drawing from about two to three years ago now using a tool that actually will capture all of the DNS queries and responses and then put them up graphically. You can see there are well over 100. It's from doing a query on the homepage of www.cnn.com, but you get a similar thing for any one of the large commercial websites because they use many different links and many different URLs on a given page. It shows up in front of you as one page in the browser, but it goes to many places on the network to get the information. As you could see earlier with the hijack where we inserted a piece of information, then it could be any one of those pieces of information or any one of those sites could be wrong.

The real point of the importance of DNSSEC is to protect the DNS zone data itself so that the answer that is put into the DNS by the original operator of the zone with the DNS signatures is the answer that is used by the validating resolver when it asks the query and gets the information back from DNS. That is the heart of what DNSSEC does: make sure the resolver gets the proper and correct information and that it wasn't changed in flight.

Here's a quick run-through of another view of the same sort of thing. Joe User is eventually going to be up over here on the far right talking to his recursive resolver, talking to the authoritative server, and the zone data was put into the DNS by the operator of the zone. In this case, we're talking about bigbank.com.

If bigbank.com is an operator of DNS that is very large, capable, and DNS dependent, then there's a high likelihood that their organization will have a lot of DNS knowledge and will be operating their own DNS. In which case, having DNSSEC added is most likely going to be best accomplished by the organization itself that's operating that name for that particular zone.

If you're, for instance, a registry for a TLD – .com being the biggest, that's always a good example – that registry operates their own name service. When they did the signing of .com, Verisign (who happens to be the registry operator that operates .com) did all of the work themselves. Highly capable, highly knowledgeable DNS staff. They have a lot of knowledge and experience in that space, so they did it themselves.

Another big substantial enterprise is HP.com. I think they did the same thing, but HP.com is a very big, huge worldwide network with a lot of DNS knowledge in the organization. So business critical zones, again, drive the need for having knowledgeable people with folks already in the organization who know how to do DNS. They can probably also do DNSSEC.

Then activities here the DNS is important but if it gets messed up or there’s a problem, it’s not really critical to an operation. Net-snmp.org is an open source project that provides network management software to folks. In and of itself, it’s not really a mission critical thing. If it broke, we would get excited and want to make sure it was fixed, but it wasn’t going to cause a big, huge business loss or anything like that.

Then there’s all of us end users. Everybody here uses DNS in some manner. So you can see there’s a huge range of knowledge and expertise relative to DNS and operating it and how much you should be doing yourself and how much you should be asking others to help you.

When DNSSEC is needed to be put in, again, it is to make sure the zone data is correct. The zone data itself needs to be properly protected, so the zone DNSSEC private keys that are used to sign the zone should be getting protection very comparable to the protection you give to the content of your zone. If, for instance, bigbank.com was not careful in how they managed their DNS even after signing it, if they somehow made a mistake in getting the wrong information in, it could still be signed and be wrong. So it's the content of the DNS zone that's the most important thing, so the content needs to be protected too.

The difference from the earlier slide and this one is really between the validating recursive server and the DNS signed data going into the authoritative server. This is a simple drawing. This is intended to be a straightforward session, but this is how simple the difference is between non-signed DNS and signed DNSSEC.

Now if any activity is doing a huge amount with DNS, they can probably also take care of doing their DNSSEC activities. If an organization is not particularly familiar with DNS operations or they've already outsourced their DNS operations, then there's a good likelihood that it makes sense for that organization to also outsource the DNSSEC additions and operations to the same activity that they're using for their DNS operations. When you do that, one of the important things to make sure is that your providers of your DNS service can also provide you with DNSSEC capabilities.

That's the short little overview and the simple little "how we get there." I think Wes will be back here shortly, and we're going to have another special presentation this time. Usually, we don't have anything particularly about the key, but since there has been a delay in the change of the root key rollover….

WES HARDAKER: Yeah, so Roy is going to come up next and talk about – if you've been following the news about the root key signing, we have delayed the execution of the rollover for a little bit and Roy is going to explain the background and history behind that.

Then we're going to come back and the rest of the afternoon is subject to questions and answer. So you guys have the ability to ask because we have a decent number of experts in the audience and you can ask any questions you might have about DNSSEC. So start thinking about good things that you don't understand or beyond the scope of the skit that you saw or anything else. But in the meantime, Roy, go ahead.

ROY ARENDS: Thank you. Julie, the clicker works? Yay!

Hi, my name is Roy Arends. I work as Principal Research Scientist in the office of the CTO at ICANN. This is about why ICANN delayed rolling over the root key. When validate DNSSEC signed zones, you need to start somewhere. You basically start with a trust anchor. Trust anchor is nothing else than the public key.

Public keys can't live forever, just like your browser certificates. They won't live forever. They have an expiry date. In DNSSEC, we agreed through the community in 2010 that an expiry date should be somewhere around five years. We're a little bit further now.

Like I said, this needs to be rolled, and this can be done automatically in your resolver or you can do it manually in your resolver or your validator, basically. But at ICANN, we have no way to test if you actually have done that. And the problem is if you haven't rolled to the new key and we stop using the old key, you can't validate anymore.

So what we did a couple of years ago, we started with a Design Team. We did an outreach campaign for about two years. We had [blocks]. We had presentations, various venues, [inaudible] meetings, at IETF meetings, etc. We contacted governments just to make sure that everyone was clued in that we on 11 October would roll the key.

So what's the people? It's not that we're going to do things ad hoc. There's a very well defined plan of how we are going to roll the key. This is only a small part of the process, but this has to do with the root zone. In July 2017, we entered the new key into the root zone. What we then did is basically we watched the traffic going into various root servers and made sure that nothing bad happened. You can basically see in the traffic if someone has misconfigured that traffic basically goes up or if a resolver can't get their data, it will get very aggressive. Same with DNSSEC.

So there's nothing going on. July 11 we did this, entered the new key. All was fine. Traffic didn't increase. It looked very good. On August 10, this is 30 days after resolvers got their hands on the new key, there's a protocol RFC 5011 that has this hold-down timer for 30 days. If it has observed this new key for 30 days, it will configure it locally. So on August 10, you basically look at the traffic. If you see nothing going up, nothing going down on average, we're fine. And indeed, we were fine.

So then 19 September, Verisign was doing their regular zone signing key roll. In DNSSEC, you have a key signing key that signs the key set including the zone signing key.

But Verisign, that's out root zone management partner, they will roll the [ZSK] every three months. They've been doing that since 2010, and there's nothing new here. This is an automated process. The only different thing is now we have an added key. Remember 11 July 2017 we added this new key, so now the response size of asking for the DNSKEY would become large and we know that some infrastructure might have issues with that. But we checked again, nothing was going on.

Now we're at 19 September 2017. As a reminder, this was only a month ago. As I said, we don't have any data. We can't go out and ask all of these resolvers, even if we wanted to, we can't. It just doesn't work that way. However, there is a little bit of data. This is a slide from Duane Wessels. Duane Wessels works as a researcher at Verisign, and he has some key tag data. Key tag data is nothing else than a resolver signaling to the root servers which keys are configured as a trust anchor. It's quite cool if you look at this.

The red line you see above, that's the old key, resolvers signaling the old key. Time goes from left to right, starts in May basically. The KSK, the new key, was published on July 11. That's the line over there. You see things trickling down a little bit. The green line below here are basically resolvers that already have the new key. So green is good; red is bad. Then this beautiful swap on basically August 10, you see that a lot of resolvers signal that they have this new key configured. However, there's a small issue here, and it's this little line. I'll go to the next slide now.

This is basically still resolvers that only have the old key. We looked at this as well within the office of the CTO, and our analysis is very similar to Duane Wessels' from Verisign. About 6% of reporting validators – and it's in red, and it's important that I make that clear – were not ready for the KSK roll on 11 October. What does 6% of reporting resolvers mean? There are about 4.2 million individual IP addresses that ask the root server for information every day. Of those 4.2 million, only a handful, a few thousand, were signaling this information and only a few hundred were signaling that they have the wrong key configured.

So we had doubts on what was going on. We were not sure what this signal meant. Was it basically a wrong signal, or could we trust the signal and 6% of these reporting validators were having the wrong key configured? It turns out it was both. We found some resolvers that were basically misconfigured, but we also find some bugs in modern software. Unbound has some problems. BIND has some problems.

But we needed to go back to the plan in process 11 October was coming. So we looked at our plan, and in the operational plan it basically abilities the root zone management partners ICANN and Verisign might also decide to extend any phase for additional quarters. For example, if new information – and that happened in this case – indicates that the next phase may lead to complications, the current phase would be prolonged. This is referred to as an extend scenario. So we took the decision on 27 September to have this extend scenario, and this is when we announced that we have a delay of the key roll.

There are still some issue. We don't know, we are looking currently. We are actually talking with those resolver operators. We have approached a consultant who is basically going out to all of these IP addresses. What we will not do is name and shame those IP addresses. You won't see them from us at all.

Also, you need to know that these few hundred IP addresses is not the same as end users. We don't know and we can't easily measure how many end users are behind that, but we are going to use APNIC's – I don't know if you've heard of this – APNIC has this [Google ad network] that it can measure things with. We're going to collaborate with them to get a little bit more data. Mitigation is also very hard. We already had this two or three year process in order to reach operators.

So I get a lot of questions on this. When is the key going to roll? My honest answer is, we don't know. What we do know is when we roll, we can only do it on the 11th of the first month of a quarter. This is because everything is basically set in stone. When you generate new keys, when you generate signatures of those keys, this between us and our root zone management partner Verisign, that's all set in stone. So if we're going to do this, it will be on, for instance, 11 January, 11 April, 11 July, 11 October.

As I said, we've hired a contractor, a consultant to track down those first 500 resolvers as a sample and data collection continues. One thing, we've already had people removing the new key from their validators. Don't do that. If you have the key configured, you're fine. Thank you. Any questions on this? Or do we do this later? Okay, if there are any questions on this, I'm here now. Go ahead.

UNIDENTIFIED MALE: Can you go back to the chart please? After July, in the middle, you published two keys, the old key and the new key?

ROY ARENDS: Yes.

UNIDENTIFIED MALE: And you planned in October to remove the old key, yes?

ROY ARENDS: That's correct.

UNIDENTIFIED MALE: And keep the new one?

ROY ARENDS: And keep using the new one, yes.

UNIDENTIFIED MALE: Oh, okay. That's my question then.

UNIDENTIFIED FEMALE: Could I just remind people who ask questions to state their names and their organizations. Thank you so much.

UNIDENTIFIED MALE: Yes, [inaudible] from STC. I'm wondering, don't you think that you take the decision much earlier? If you would wait until 10 October or 9 October, that you have nothing to lose before you decided to keep the old one or not. You decided to cancel the rollover in end of September. Why you didn't wait until beginning of October? I mean one day before the migration, let's say.

ROY ARENDS: Yes, I can answer that. We had this information basically the week before 27 September. We already knew by doing our own research that bugs in resolvers – and these don't go away, not that easy – bugs in resolvers need basically a new cycle of release of resolvers in order to get them fixed. So we knew that people were suffering from this bugs.

Now we had a choice of waiting until the very last minute and hoping that this line goes down, but there isn’t really a trend. It might look like there’s a trend, but it won’t really go down – and it didn’t actually. I can tell you that now. It won’t really go down by 11 October, so taking the decision to delay the roll gave users who were misconfigured more time to get things done, but it didn’t impact the people who were prepared, if that makes any sense. So that’s the reason why.

There’s also something fundamentally wrong if we take internal within ICANN the decision to delay then to wait to the last minute in order to announce it. So I think it’s actually a good thing that we took the decision earlier rather than later, in my opinion.

UNIDENTIFIED MALE: [inaudible]. From this graph now, the percentage of people using the new key is quite high compared to the people who are still using the old key. Now comparing the two, are you waiting until you have 100% compliance before you revoke the old key?

ROY ARENDS: Sorry. The last part I didn’t hear.

UNIDENTIFIED MALE: What I am saying is that if you look at it from the graph, the percentage of the resolvers using the new key is quite high compared to the old key. Now the question I'm asking is this, now we are waiting because of that [cycle], the trend in that [cycle, right?] Now the question is this, are you waiting to have 100% compliance of resolvers using the new key before the old key will be revoked?

ROY ARENDS: No. It's a bit of a bold statement, but no. We won't wait until 100%. You can never get to 100%. There are configurations out there that are [stale] that people don't actually look at anymore. They really don't care if they do DNSSEC or not. The operator has left the building, if that makes any sense, and these machines still talk to the root servers. About 98% of traffic to the root server is questionable, if that makes any sense.

UNIDENTIFIED MALE: [To top it off], what indicator are we waiting for now to finally revoke the old key?

ROY ARENDS: Just a small clarification, it's stop using the old key that's the problem. Revoking is much later. That was planned in January. But stop using the old key 11 October, you're basically asking what percentage is good enough. It's not a question about what percentage is good enough.

The only reason we delayed, and there's no other reason than the following, is we needed to understand the signal that we're currently seeing. If the signal is basically a false signal, as in there are bugs in resolvers, then we need to figure that out. We need to have a look at that. But if the signal is basically people doing silly things with their resolvers, not having listened, not having tuned in, we have done this two year/three year outreach campaign, then so be it. So we're not going to wait until a specific percentage, but we hope using this sample of 500 we can classify the signal.

WES HARDAKER: Let me add one more point to that, and Roy alluded to it, which is that this point on October 11 which is where we were going to switch [to using the old] key did not revoke the old key. The old key was still going to be good. The plan was to only sign with one of them. If we decided that on October 12 things weren't going well, we could have shifted back to using the old key immediately again and that wouldn't have been a problem. It was not until sometime in January where it was actually planned to tell the world, "This key is no longer good, and you can never trust it again." That was going to be long after we actually switched which one we were using. Does that make sense? Okay.

Let me take the opportunity now to say we'll go on to ask any questions. We'll continue asking questions [about it] here, and I'll make Roy stay up here because he happens to be an expert in all sorts of DNSSEC related stuff. But if you have any questions about how DNSSEC works or about the KSK rollover topic, now is the time to ask. If you have a question in the back, we will bring a microphone to you. so just go ahead and raise your hand if you're in the back audience as well. Anybody have questions?

UNIDENTIFIED MALE: I'm not quite sure why we need to change the [public] key. I mean the most [fallible] information is the [private] key. If we keep it secret, why we need to change the [public] key?

WES HARDAKER: Well, it would change both. The question is, why do we need to change the key? Go ahead, Roy.

ROY ARENDS: The answer is not a technical answer. You are absolutely right. The current key is fine. The current key that we generated in 2010 works, still works, will work for a number of years. There's nothing wrong with that. However, we made an agreement with the community wherein we agreed, and this is in the DNSSEC Policy Statement (DPS). It's basically an agreement with the ICANN community where we state because it needs to have a timeline, it can't live forever, we state that in five years we will change this key.

Now if you make this commitment in 2010, it takes an enormous amount of effort to basically – and really, not many people want this – it takes an enormous amount of effort to convince people not to roll the key ever.

There's another thing. We would also rather roll the key while there is nothing wrong with the key. If we have to roll the key when the key is compromised, we are in far bigger trouble than we [were] ever before. So it's safer to roll the key when things are good. We promised the community we would do this in five years. In fact, we're seven years further, so we basically owe it to our community to roll the key.

WES HARDAKER: Any other questions? Yes?

UNIDENTIFIED MALE: From the answer we just got now, does it mean that the new key will also be due for replacement in another five years?

ROY ARENDS: Very good question. I was hoping someone would ask this. You are basically asking, when is the next key roll? If I might change the question if you don't mind into, what's the frequency going to be on the key roll? There is I think an open session on Wednesday evening, the Technical Experts group. I'm looking at Julie. She knows everything. No, that's all right. Sorry, Julie. I didn't want to put you on the spot. On Wednesday evening, there's a Technical Expert group meeting and a presentation on there is about the frequency of the key roll. What we don't want to do is wait another seven years to do this because in seven years we forgot what we did now.

WES HARDAKER: One of the reasons to roll keys on a regular basis that a lot of people have argued for is that when you do it regularly, everybody knows how to do it and knows it's going to happen. Some people say you should do it once a month and some people say you should do it once every ten years. The reality is that somewhere in the middle is a frequency at which it makes everybody that runs a resolver know how to do it. It makes ICANN exercise that process on a regular basis so that it becomes a smooth operation. And it's under debate as to how frequent that it should be.

ROY ARENDS: Yes. Pre-empting the discussion on Wednesday a little bit, we cannot do this faster than every three months. Every three months is how often Verisign rolls the [ZSK]. If we want to do it faster than three months, we have to break out [all the] procedures that we have established over the last couple of years. So it won't be faster than three months. Makes sense, doesn't it? It also won't – in my humble opinion, but I'm just a researcher – it will also not be another seven years. So that gives you basically a frequency band. It's going to be somewhere within three months and seven years.

WES HARDAKER: I know in the skits and the previous diagrams that we did in the beginning that explained about DNSSEC, we didn't talk about the fact that there are actually two different types of keys. That's a complexity that, unfortunately, has to come out for this discussion. But really there are both zone signing keys and key signing keys, and they're used slightly differently.

I saw a question. One sec. Russ, do you want to add something?

RUSS MUNDY: Yeah, I wanted to also add that each of the individual zones – we're talking about the root zone here, but if you remember from the skit there's .com and then there's bigbank.com. Each of those zones also have their own keys and their own signing, and they also go through a key change/key roll mechanism on some periodic basis.

The thing though that makes the root zone key roll so much different and so unique is that's the starting point that the validating resolvers have to use to go down the chain. But the fact of doing a key roll in DNSSEC signed zones for the rest of the zones is a much more frequently done basis, often as much as once a month for some of the zones further down in the tree.

UNIDENTIFIED MALE: Is there any plan to use Public Key Infrastructure (PKI) with automated way to refresh the public key so that for such cases in future?

ROY ARENDS: [Sorry to] take this question again. There is on the IANA web page a list of links that point to various different ways in order to get this trust anchor. One of them is actually – sorry, I'm doing this from the top of my head – is the identifier or the unique [hash] basically of that key [embedded in certificates]. So the distribution of the key is already kind of PKI-ish. I think that's the closest we're going to get with doing anything with certificate authorities PKI infrastructure. I hope that answers your question.

WES HARDAKER: PKI infrastructure and DNS keys are quite different. It's a different security model and a different system, although the math underneath is actually kind of similar between the two. But they're really not tied together. It's a different thing. And they both operate very critical aspects of Internet security in general.

Any other questions about anything at all? Anyone? Now is your chance. It's a complex subject and it's a complex topic. I know if you ask any of the people up here today in the hallways if you run into us again if you come up with [other ones], please do feel free to tap us on the shoulder and ask more questions if you run into us and don't want to ask publicly too. But we're here now for another few minutes, so I guess last call. Any last questions at all?

Okay, I think that sounds like we are about done. Thank you all for coming, and hopefully you got a little insight into how DNS works.

[END OF TRANSCRIPTION]