ABU DHABI – DNSSEC Workshop -- Part III
Wednesday, November 1, 2017 – 13:30 to 15:00 GST
ICANN60 | Abu Dhabi, United Arab Emirates

UNKNOWN SPEAKER: DNSSEC Part III, Hall A.

JACQUES LATOUR: So, we're going to be a few minutes before we get the demo up and running. [AUDIO BREAK]

Alright, welcome everybody to the DNSSEC Workshop. Now we're in the part III and next is Wes Hardaker. He's going to do a presentation on LocalRoot - Serve Yourself.

WES HARDAKER: Alright. So, out of curiosity, how many people here run a recursive resolver somewhere within their infrastructure? A good number. Even better, how many people manage somebody that runs a recursive resolver, so you can tell them what to do? Today I'm going to talk about -- do you have the full screen button? Where is it? [AUDIO BREAK]

We lost the document. [AUDIO BREAK]

Alright, so, how many of you are familiar with RFC 7706, which is serving the Root Zone on the loop back address? Good, excellent. So, there's a few people. The important part about it is that it allows you to serve the Root Zone to your local networks that are using your recursive resolver without ever having to go talk to the Root Zone. So that's what this is going to be about today. So I started a project called LocalRoot, and we'll go through the details of what that means.

So, first off, what is LocalRoot. As I just said, it's a project that allows you to load the Root Zone into your local resolver. So, as a quick background, here's a sort of classic DNS Resolution. On one side we have a list of clients that are all trying to do DNS lookups. Maybe they're going to a webpage, whatever they're doing for the day, maybe it's their phone doing automatic updates, and they talk to an ISP. So, it's whatever network they are connected to. At ICANN, it would be the wireless network here for example. And in that ISP is a recursive resolver, and that recursive resolver's job is to answer every query that the client sends to it, and it does that by talking to the rest of the DNS infrastructure.

So, for example on the right-hand side, you'll see that there's the root zone, there's common org and underneath that, we have examples, so example.com, and then on the right-hand side we have ICANN.org.

So, what happens is in this classic example, say that top client asks for, "Hey, I want to go to www.example.com," the resolver in the middle sends that question. It starts off sending it to the root. So you can see that www.example.com goes up to the root first and then the root is really just responsible for giving a next iteration down to where to go next. So, it says, "Here's com. Go ask him." Then the resolver asks www.example.com to the com server and then com finally gives it down to example.

A couple of important things about this, that most people at least know the concept of is that there's also a cache in the resolver that remembers stuff. So it remembers that com exists and it remembers that example.com exists. That doesn't help if the next client comes back and asks for icann.org, because org is not in the cache and icann.org isn't in the cache, so it goes through the whole process of chaining to the other side of the tree.

Where it does help is with the -- did I ever mention I'm not a fan of Adobe's connect? Where it does help is in this case for example, all the clients are going to ask for what's www.exam.com. Well, because the resolver already knew about com, it does not have to go ask where com is. It can actually start at com and go down from there, asking where exam is. So you can see that there's far less arrows on the screen because it had already had com in its cache, so it only has to add exam.com.

So, this is what I'm proposing to change with my project that you can all log into and put this in your own resolver if you want. What happens is instead of DNS Resolution happening where the resolver contains only the top level information for the root, I actually give you all the information for the root, so it's basically caching every single TLD out there that includes com and org and net and cx for the country, and then horses, which is one of the new gTLDs. So you end up with sort of a pseudo cache of all the information in the root.

What that means is that DNS resolution with LocalRoot in place, those red lines are no longer needed because the resolver already has all that information, so it doesn't need to go ask the root anything. It'll know where com is. It knows where org is. It knows where horses is, so it can actually start just after that.

The other thing that my LocalRoot project does is it keeps your local resolver and your resolver's root cache up to date. It does that by actually sending out DNS notifications just as if you were a slave to the root, you actually do become a slave to the root for your local infrastructure only. So, what happens is, when the root changes, the local root servers will send you a notification saying, "Hey, come grab a fresh copy. Something's changed," so that your data is always up to date and you never have to worry about it.

So the question is, "Why would people do this?" And there's a couple different answers. If you read RFC7706, it goes into a little bit more detail about why this might be useful, especially in distant parts of the world where there's a lot of delay between you and say the servers around the world. The primary benefits are again, pseudo caching of the root data. Everything is in your cache. You remove the need to contact the root at all. It actually takes the root almost completely out of the equation, and you get faster lookups for TLD lookups, so the response time is zero milliseconds for something in the local infrastructure.

Another important thing is it's not that you just get faster TLD lookups. You get faster lookups for things that don't exist. When you type in eBay without any other information into your browser, most browsers go off and they go, "Did you mean eBay this? Did you mean eBay that? Did you mean ebay.com?" So it actually is searching for a lot of stuff. And if you're doing a lot of those, the negative resolution, the fact that dot eBay itself doesn't exist -- maybe it does now, I don't know -- that takes a while to come back, so those negative answers are actually almost a bigger benefit than the positive ones, because the positive ones are cached, the negative ones often aren't. So if you ask for eBay five minutes later, it's going to go ask again and it doesn't remember that it doesn't exist.

`Anyway, so again, you always get an up to date copy. And then, this is designed for me to be a research project to a large extent so you can do research of your own when you get DNS notifications, maybe you're working on DNS software where you might actually want to know when the root zone changes. This is a mechanism for you to actually subscribe to in-band DNS notifications of change.

So, a couple of things with respect to security. The best thing about DNSSEC, and this is one of the reasons that I'm here today, is that because it's signed, you actually don't care where you get data from because DNSSEC data wants it signed. It's signed by IANA. I could give it to you incorrectly and you'd be able to know that I gave it to you incorrectly, so the wonderful thing is that this is actually now possible, this whole concept of a local root is actually possible because that data is actually signed and everybody can trust it.

We do add TSIG security on top of it. The notifications and the transfers are also protected by a TSIG key, which you'll see in screenshots in a little bit. Right now, that's mandatory. Some people have said they don't need TSIG. It really doesn't buy you that much security. I won't go into the nitty gritty details of whether that's true or not, but I may turn it as an optional feature in the future.

So next, I'm going to go through a demo; due to Adobe Connect fun, I'm going to show you the screenshots instead of the live demo. But, this is the main page. Unfortunately, the text on the screen is a little bit small, but that's okay, most of the fields that are in them are big. So, if you just go to LocalRoot.isi.edu, which will be on the final slide but you are more than welcome to go there now if you want to type that in. It's localroot.isi.edu.

There's some starting links about local root, it gives you a whole bunch of information. There's a getting started thing that shows you how to walk through four steps to get a resolver up, but the first thing you have to do is register. So, you'll type in your email address and a password, and do captcha processing if needed.

Once you log in, you'll have to go through a registration process and get a registration code in your email and stuff like that, but once you're logged in, there's only a couple of things that you can do because it's actually a fairly simple system. The first thing is, as I mentioned, there is a getting started link. That's a good place to read documentation to get started. But the first thing you have to do is create a TSIG key. Actually, here's a screenshot of the getting started document. So, it walks you through all of the four steps in order to get a resolver up and running.

So, this is the list of TSIG keys. There's nothing there right now. The little green text says “No TSIG keys are generated yet”, and there's a little “Create New TSIG” button; and when you do that, there is a form to fill out and you can put in anything you want as an administrative label. What do you want to call your key? It can be anything you want. In this case I put in “my cool TSIG key”, and when you click next, it shows you the key, the administrative name on the left, and it automatically gives you the algorithm and the value.

You actually don't need to do anything with this. This is all automatic and it will spit out some config in a little bit, but it's a list of the keys that you have. You don't need to actually copy the value down as I'll show you in a second.

So then you can go over to your servers list. This is a list of your resolvers. I should probably change that word actually. It's a list of the recursive resolvers that you want to deploy your local route structure into, and again, there's a button for “Add a New Server”, and then underneath that, there's only three fields to really fill out. The first one is an administrative name for your recursive resolver. You can call it anything you want. I think most people would put in the real hostname of their server; that's how we think about our machines. And then you have to add the IP address for where your server is. It supports both V4 and I don't think I have V6 on the box yet, but that's coming.

Then, which TSIG key do you want to use. In this case, because I only had one, it’s going to give me the right one, but if you had multiple TSIG keys, you can sort of granulate them out into different regions if you want. And when you're done with that, you push the Create Server button and you get a list of your servers.

And the text is small, but basically it's the same information that we had filled in: the name of your server, the address and the TSIG key, and then the three right-hand things are enabled, so there's a checkbox next to Enabled, an X next to Active, and a Config button, so we're going to go through those in a second. Enabled means you can actually click on it and toggle it on and off. If you don't want notifications sent to you anymore, you just turn it off.

The active one is, one thing I realized is I didn't want people to log into the system and then start sending DNS notifications to anywhere under the sun. So, I added a little bit of security protection that you have to go do a manual AXFR transfer from your address to the local root server before it will become active.

So it's just a security step to make sure you're not trying to mess with somebody else's address. It's actually quite easy to do and there's literally a command you can cut and paste out of the getting started text in order to do it, even if you don't know how.

And then finally, that Config button actually spits out everything that you need to do to put into -- right now, the only config is for ISCs bind. I hope to add some others in the future. Not all recursive resolvers support AXFR transfers, to actually get a copy like unbound unfortunately. Specifically, a recursive resolver can't implement LocalRoot, but ISCs bind can.

So all you have to do is copy and paste that into your configuration file and you're done. It gives you everything you need, and you'll note that in that bottom list, there's a whole list of IP addresses you can pull Root Zone data from. The LocalRoot server is the first one. As time goes on, I'll probably add more copies of the LocalRoot master server, but you can also pull from B, C, F, G, and K, and a couple of ICANN sites too, so if any of these actually went offline, you'd actually still be operational.

So, it's actually a very safe thing to do, because even if my university decided to go belly up and that top server went off, you'll still pull data from the rest of them. The only thing that would happen is you wouldn't get notifications, but the servers already do SOA pulling on a regular basis, so you'll never fall out of operation. It's a very safe thing to do.

So, let's talk about real world effects. This is me, from in my house. I run a recursive resolver in my house because, well, I'm a geek. And the traffic on the left with the lots of spikes is basically queries per second. You can see that some of the peaks go up to 40, 35, somewhere in there, and then all of a sudden, they go flat. That actually happened when I turned the LocalRoot service on.

So, those are all the queries to B root only from my house and then all of a sudden, there's very very few, and that very very few, I'll talk about those reasons in a second. It went down to flat because I no longer had to talk to the root. The root was now locally in my house. And then I turned it off a while later to watch it come back on again. So, now, it's just running straight and I get a whole lot less requests to the root.

An important thing is that the reason why there's little tiny dots still on the bottom and it's not totally flat is that slave zones for even the root, or any zone, does go out once in a while and it queries an SOA record to make sure that it's still fresh and it didn't miss a notification. The other reason is, I have a RIPE Atlas probe in my house, and the RIPE Atlas probe is bypassing my recursive resolver entirely because it's measuring B root by itself, so there's a little bit more traffic for me than there might be for you if you don't have a RIPE Atlas probe.

So, next up is just questions. I'd love for you to try it and see what you think. I did notice that there's a bug on the webpage a few minutes ago, that I need to go fix; so the login button is kind of -- I think if you go through register, then the log-in button will work, but I think I broke something in the last hour. My bad.

I'd love feedback. I'd love to know a) if you're interested in it. I'd love to know if you're going to use it and try it. If you do any analysis, I'd love to see how much it actually dropped your traffic. Note that because you're pulling the entire Root Zone data down, it may actually increase your traffic, but it will still increase your speed because you probably don't go through every TLD out there right now anyway.

If you're research focused, if you're doing research projects based on it, the data or notifications at all, I'd love to hear what you're doing, and additionally, which other features would you like to see. In the FAQ, there's actually a list of things that I'll do in the future. Right now, I literally just finished this last week, so I'm still expanding on it, like there's no Delete Server button yet, kind of missing. There's no I lost my password button yet. That'll come at some point, and it depends on popularity, so if a lot of people find this useful and really want to put it in their local environment and it's popular, then I'll put a lot more effort into it. So, any questions about the service at all? Please.

CRISTIAN HESSELMAN: Hi, I'm Cristian. Would you also be able to extend this concept to the TLD level?

WES HARDAKER: Unfortunately, no, because I don't have any connections to TLDs. Now, whether TLDs would be willing to do that for you, I don't know. We'd have to go talk to every single one of them. The reality is that there's a lot of TLDs with, I don't want to say proprietary information, but restricted information, so I'm sure some won't let you do it. It'd be interesting to have that discussion with them. I don't know. I don't think you want all of com in your house for example, that's a little too much bandwidth, but some of the other ones I don't know. Good question.

UNKNOWN SPEAKER: All the gTLDs are available, albeit only once a day.

WES HARDAKER: There is a -- I shouldn't turn mine off -- there is a get repo where actually you can pull down TLD information with get Deltus as well from GitHub. So that's an interesting idea as well. How much information can we cache locally? That's a good idea.

UNKNOWN SPEAKER: Also, my impression is that although the root nominally changes like once a day, in fact, if you're a little stale, the chances of problems are low and I'm wondering if --

WES HARDAKER: Yeah, so one of the other things that I was thinking about in the future is, in order to decrease bandwidth, the reality is, is the root doesn't change very often. It gets reassigned twice a day generally. Sometimes you'll see a change three times a day, but it's pretty rare. I operate the USC infrastructure for the root, so I watch this on a frequent basis. The data itself doesn't change very much. It's the signatures that are changing most of the time.

So one of the things that I might do, and this is in the FAQ, is to only send you a notification once every one third of the signature expiration time, and then there would have to be some code and bind as well just to not do the SOA pulling and pulling. Because you don't need the data that frequently, you're absolutely right. So, that's a potential research project for me in the future, and if someone actually wants that, let me know so that I can make that happen.

UNKNOWN SPEAKER: Keep us posted.

WES HARDAKER: Yeah, will do. Yes?

RAED AL-FAYEZ: Yes, hi. This is Raed. Is this concept, is it still a concept or is it like best practice or is it stable? Can I put this in an operational network? Can I ask my let’s say ISP or something bigger, like Telecom provider, to use this concept? And is it safe? Is it stable?

WES HARDAKER: Very good question. So, two things, or a few things at least. This project that I created where you can actually get notifications and sign up and it will spit out config, that's very new, so it's listed actually on the slides it says alpha but I upgraded it to beta, so now it's beta. So it's still in development.

That being said, the way I've designed the config that it gives you is that even if my service completely fails, you'll still get all that data from the other roots. So, RFC 7706, which is this whole concept of caching the root data ahead of time, is a published standard, and it's not experimental I don't think. Anyway, the config I give you will not break, even if I break. So, I would say, yes you can probably put it in production from that point of view, from the local root and getting notifications. I need to see how much interest there is. I have no intention of turning it off. How much development I put into it depends on how many people want to use it.

So, if you're interested, please write me and we can have a correspondence about that. Okay. My address is on the website in the bottom of the FAQ, too.

So, any other questions? How many people want to turn it on? Yay. You can put your hand halfway up, I'll take a halfway answer. That sounds good. Just a minute. Alright.

UNKNOWN SPEAKER: Yeah, I've got a question. So if over time, more and more people do this, more recursive do this, what does it mean for the infrastructure for the root?

WES HARDAKER: It'll be slightly less used. One of the long term questions is what's the best way to distribute initial bootstrapping information for the root zone and that's being discussed in many walks. So this is like one option for better ways to get the initial information out to everybody, the prime inquiries especially. Alright, my timer went to red right now.

JACQUES LATOUR: Any other questions? Cristian.

CRISTIAN HESSELMAN: It's more of a remark. I think that if you would extend this concept, if that’s possible, if you would extend the concept to the TLD level, you would also increase your DDoS resilience because you would push the information further down into the edges of the network. So let's say a certain registry could be DDoS off the planet but things would still work in terms of resolution.

WES HARDAKER: Now that's a really good point. The problem is, the TLD information is often much larger. The Root Zone is actually a very small zone. So you don't want to do it with com because it's just not worth it, but it might be worth it for some other domains that are smaller, so I'm going to look into that. I think that's actually a really good idea, and I appreciate that feedback because I hadn't thought of that.

RUSS MUNDY: 7706 is informational.

WES HARDAKER: That's a really odd classification for that. Okay.

MOHAMMAD: Yes, it's Mohammad for the record. What I see from the presentation is it’s a TSIG transaction between your recursive server and the root servers, so from what I know the root administrator should config it's servers with a TSIG key, so all of the root administrators are operators?

WES HARDAKER: No. So, TSIG is a shared key between two DNS servers or a client. So, that just means that it's like having a shared password, kind of like when you zip up a file and you can encrypt it with a password and the other person has to know it to decrypt it. Same kind of concept for a single DNS transaction, so you share the key and that just makes sure that you’re -- it's not actually encrypting, it's actually just protecting it so that you know it hasn't been modified in transit.

So the TSIG key that you generate is only good between my local root server and your recursive resolver. It has nothing to do with the roots. I get notifications from the roots into the local root and then I'm basically sending that notification out to you with TSIG so it's protected in multiple ways. But no, you don't need to deal with the other roots at all.

MOHAMMAD: So, your server acts like a midpoint between me and between the root servers.

WES HARDAKER: Exactly. Exactly, yes.

MOHAMMAD: And in this case, if your server is down, so how can I reach the other root servers?

WES HARDAKER: The other root servers; your server will still pull them on a regular basis, like once an hour just to make sure that they haven't missed data, so even if you don't get a notification, the way recursive resolver software works is that they will go pull the SOA to see, "Oh look, it's changed," and then they'll go pull a new copy anyway, so it's not a problem. It's safe. Thank you very much, everybody.

JACQUES LATOUR: So I’ve got one last observation question.

WES HARDAKER: You're running the show, so.

JACQUES LATOUR: I'm running the show, so I'm thinking, if we replicate this for dotCA for example, where we can serve the dotCA zone on to a large Canadian ISP, those are the main kinds that went for resolving dotCA. With the same framework, you pick and choose who you want and that means potentially the requirement for CIRA for any cache infrastructure. I don't need to be super extra big. I can have a good enough infrastructure, but that draws a line where how big is big in terms of infrastructure.

WES HARDAKER: Are you able to tell me how many records are in dotCA?

JACQUES LATOUR: Two million.

WES HARDAKER: So, large ISPs probably don't care. The smaller ones probably don't want to cache all of CA, but the larger ones I could see them doing that. I hope at some point that I will open sources. Right now, it's on my local servers just because I haven't put it out, but I would love to speak with you. I would be happy to give you the infrastructure I've developed.

JACQUES LATOUR: I think there is something to look at there. Thank you.

Next is Vittorio Bertola from Open-Xchange and you're going to do a presentation about DomainID using DNSSEC for a secure, federated digital identity system. There you go.

VITTORIO BERTOLA: Thank you. This is a big project by Open-Xchange. For people who don't know it, it's a European Software company making webmail platforms and also a couple of very ubiquitous software applications like [inaudible] and Power DNS, but it's a joint project with DNIC so the dotTLD registry, and with 1&1.

So, one of the side effects, and actually one of the reasons we are doing this is to promote DNS and DNSSEC since the idea is to one day use DNSSEC for the [inaudible] that are in this project. So, I'll go pretty quickly because this a general PowerPoint presentation. I think we're aware that there is a problem in how the identities are managed over the internet, so we all have too many user names, too many passwords and it's hard to track them.

Some people use password managers, but most people just reuse passwords if they can. If they cannot, due to different password requirements often you cannot reuse passwords so you have more than one and then maybe you write them down, so this creates all sorts of users. So, everyone is looking for a solution to this. And of course, the solution is a Single sign-on system. There are already many of them, so that you only have a single account and you can use it to log into every website over the internet.

But there are already several global-scale single sign on systems, but we have a problem with them. The first one, the first type of systems is the government-ran ones. In Europe we have the eIDAS project for eID cards mandated by government, but the problem is that they are really heavy and also you might not want to give your true name and address into each and every website you want to log into. So, in some cases, maybe if you want to log in to your bank account, you might want to use your official ID, but for most websites, that's overkill. That's too much in terms of protecting your privacy.

So, what's really taking off and everyone is starting to use today is this one. So, the OTT run global identity service that lets you log into websites with your Google account or your Facebook account, your Twitter account, which is very handy so most people are starting to do this, but we think this is not really the way it should be done because it's owned by a single company and more often than not, these are companies that monetize your identity, so they actually offer the service so they can track you around all the services you use, so you don't have real privacy guarantees.

And you don't really have a choice, you should be able to choose your identity provider and not just to use one of the two or three that are supported everywhere. So, we were wondering whether we can find a way to make all these identity systems interoperated and create a public identity system which, for the moment, we are calling it DomainID. We are looking for a better name.

But, the idea is basically to build a global single sign on system which is public, so it should be a public open standard that everyone can implement. You can have a million a different entities offering identities and all of them would interoperate. So any website accepting logins from the system would be able to use and support any identity given by any company, any provider, and you could even install the server and run your identity yourself.

So, based on public standards, it would also try to empower the user, so let the user choose which data he or she wants to share with every website and have different layers and control on which individual data items are shared with every website. So basically, from the technical standpoint, this is based on OpenID Connect, which is the same protocol that everyone is using, including the OTTs, but we are adding several things, and actually, what we are adding is building over the DNS.

So, the problem is that you need that discovery process. The idea is that you can use any hostname or email address as an identifier so you would be able to put your identity, your identifier inside your own domain name, or inside any domain name you want to use, but then you need some mechanism to map your identity to the entity which is managing it. So, the idea is that you would use a DNS record that we're showing to map the identity to the manager.

So, what we are adding is not just a discovery process, we're also making it so that for website owners, you'd only have to implement one log in form and one library and you can accept identities from anyone.

We also offer portability. The idea is that if you're not happy with a company or an entity or whoever is managing your identity and you don't trust them anymore, you can move your identifier to another provider and then you continue using it. So, it just gives the user some power by gaining power in respect to the identity providers and you're not forced to stay with the company that gave you the original account.

Also, we're trying to mimic the architecture of the domain name system in terms also of separation of roles, so you should not have a single company that knows everything about you. We're separating it into two parts. We have an identity authority which is the equivalent of the TLD registry which is the one that's actually managing the actual authorization and authentication, so it's checking your password or whatever credentials you use. Then, we have an identity agent which is the equivalent of the registrar which is the part that's actually having the relationship with the user, so as a user, you would go to one of these identity agents and get the service, and they would then create the identity of the identity authority.

Then we also want to give you, the user, the possibility to give and withdraw consent to each individual piece of data that you want to share. We edit them in a single place, so that if you change your address, you don't have to go and change it everywhere, a hundred times in a hundred different websites, and you can add more information about you, as long as you trust that the only party that gets to know it, which is the agent. Then it can be shared individually to different websites under your control. So, this is how it works technically.

The idea is that you use basically any valid DNS hostname. It might be the same host name that you find in the computer, but it may also be any string which is in a domain name that you control. Basically the point is that you would go to this identity agent and ask for the service. They would possibly get you a personal domain name, because you have this in your Telco domain name, your provider, so you can get this from a company, but the real value for the user is if you get it into your own domain name. So, maybe you buy a domain name if you don't have it already, and then the only thing that you need to do, and the agent is doing this for you, is setting up the DNS records.

So actually, it's one DNS record. We are using a TXT record in d-mark style so it’s a series of name and value couples. And for the moment, it does very basic information apart from one version, the identifier basically does just two pointers, say which is your identity authority, which is your identity agent. So, anyone can just make the DNS query and from the identifier, learn which servers are to be contacted if you want to authenticate the user. That's it.

This is the creation part, so of course the identity agent would then go to the authority. The authority would verify that everything has been set up correctly, and then they would confirm the identifier. In the end, the user would be directed to go directly to the identity authority because the point of the entire system, one of the points is that the only entity that actually gets to know your password, to see your password, is the identity authority.

So, there's just one place where your password needs to be secured, and you can remember even a difficult password because it's the only one you need to remember, so it's sort of a password manager, but it's online. The agent and also the websites never get to see your password. They cannot steal your password if they are not good people and they cannot pretend to be you somewhere else. Or they could not even leak the password if they are cracked or all this kind of problems.

So, this is what happens when you actually have to log in. You go on a website, you enter your identifier and the relying party, the website, only has to do this DNS query and get to know which is your authority and which is your agent. This is the part that is also secured with DNSSEC and we are mandating the use of DNSSEC with policy of course because this needs to be secure, otherwise you can be hijacked.

Then, the rest of the process is actually the standard open ID connect process that is already implemented. There's many, even free, implementations available of the different server parts, so I don't know if people here are familiar with OAuth and OpenID Connect, but basically there's a procedure; in the end, the user gets redirected to the authority which asks the user for the password, if they need to do so, because you might already have an open session so in this case, you don't really need to enter your password anymore.

If the authority wants, they can implement two-factor authentication or any additional check, and as soon as they implement that, it's immediately available for any log in in the entire world. So, it's also easier to make logins more secure.

In the end, after the log in, possibly the user will have consented to share some information, especially if it's the first time they log in with the website. So, the website will get an access doc and that authorizes them to go to the identity agent and retrieve some data. The idea is that the authority actually shows the user list of all the data that the website is asking and the user gets to say, “Okay, I'm sharing this, I'm not sharing this, I don't want and so on.” The user has full control over which pieces of information are shared with the website.

So, there are several reasons why we think this is good for the users. First of all, you choose your identity. So, you can use your personal domain name, you can choose a company, you're not forced to go with the domain name of the company that is providing you identity, be it Google or Facebook or whatever. Also, you can pick the provider, and this is really important. It's a way to create an even, fair relationship between users and identity providers, so if you can port your identifier, your online identity to a different provider, then it's easier that providers will compete in a positive way to give good services, trusted services, and in the end, you also can choose to use as a provider someone who is not monetizing your data.

In this initial setup, in our proof of concept, then it is running the authority and these are not for a profit company so they don't have really a reason why they should want to sell their information about you and all your log ins to someone else. This is really an increase in the trust, even for users, in the trust of when using the internet.

Then, it's more secure because it's true that now you have a single place that needs to be kept very secure because it's where all your logins go through, but you don't have passwords everywhere. You don't need to have 1,000 different passwords to remember them or to write them down or to put them on one device or somewhere. In the end, even if your manager loses trust or they are hacked or cracked or whatever, you can just change it. Change the password, change the identity authority and you're back online in five minutes or so.

And also, it's meant to be more private. There's no reason why you should not have multiple identities. Even today, people use different email addresses to sign up for different services, so you could have multiple identities, maybe even run by different providers. It also helps you because there would not really be the need to register anymore on websites because you just log in and your information is already there. It just needs to be communicated to the website. You can choose which information you share, so you are in control of your information.

But then, more importantly here I think is what is the strategic value of the domain name industry and the domain name world. This is also of course meant as a way to sell more domain names to promote the sales of domain names, which is always nice, but in the end, it's meant also to promote the DNS and to keep the DNS relevant because this is the real thing for us.

We are really scared, I know how many people in this room share this feeling, but we are really scared by the direction that the world is taking in the terms of online identity, tracking and privacy, and from one point of view, the ability to track the identity, to check the identity of users is important because if you have good authentication over the internet you could attack several abuse and security issues which in today's world are harder to deal with, but on the other hand, we don't want to build an internet in which everyone is constantly being tracked or monetized.

Unfortunately, identity tracking is really the cornerstone of the world gardens that are developing over the internet, and so we think that the identity management layer should, like email for example, get back to being a public standard interoperable run by many different providers and even self run by users, and not just be the private ground of a few big companies and providers.

This is also strategic for DNS. So, the idea is that nowadays the DNS is the internet's public directory for hostnames and services and so forth, technical identifiers. But there's no reason why also information about people should be in the DNS. We think that the DNS is a wonderful, public shared and distributed directory. Now it's also secure if you use DNSSEC, so it should really be strategically the place where also the information about people is stored rather than private databases or whatever.

So at this point in time, there's really no way to do what we are imagining so this is why we've been working on it. We want to build a public open standard for everyone to be able to provide identities and to get identities, and to log in with one identity everywhere and share their data in a controlled, secure manner.

This is what we are building, so at this point in time, we have a proof of concept which is working. We've written the first couple of internet drafts which have been submitted in the last couple of weeks, so they are already online, and we look forward to seeing, if there is interest, where this can be standardized if it can become an idea or standard in some way or also partly should be within the open idea foundation, which is also dealing with the Open ID standard.

But more importantly, we're trying to get feedback so we're now in a phase in which we are presenting the project in different places and trying to see whether people see a need for this and want to maybe participate; even participate in developing the standard and working on it. We've got some pretty good feedback from Telco because of course Telco have a problem with the fact that people now are logging in with Google identities rather than with their own, but also from the software community because there's many people in the free-software world that are worried by what is happening in terms of identity management.

So, that's pretty it. I'm happy to take questions. Sorry if I did it pretty quickly. If there's anything you want to ask, I can explain it a little better. Thank you.

UNKNOWN SPEAKER: Hi, Vittorio. [Inaudible] from CZ.NIC. I like the idea. Maybe you know that in CZ.NIC we are already running the identity provider on top of our registry for more than a half a million registered users, so anybody who has a domain can have also identity in our OpenID Connect provider. I've already registered a domain ID and it works. It's nice. I think that we've already talked about this with Marcos from DNIC that how it is designed it could be quite easy to add our half a million registered users, the DomainID record in the DNS and extend your user base.

VITTORIO BERTOLA: That would be really great. Of course we're looking for adopters to promote the standard and the idea is really that anyone already having an OpenID Connect based authentication system just needs to put the DNS records in and then that's it.

Of course, then there's the other part of the job, which is convincing websites and online services to adopt this, but if we get some user base from registries and from some big Telcos and so on, we can have a critical master plan and invite people to support it.

JACQUES LATOUR: Russ, questions?

RUSS MUNDY: Thank you, Vittorio. Very interesting approach and how to leverage off at DNSSEC. One of the questions that came to my mind watching your presentation is clearly some amount of change would be required up of the sites that one was going to. Any sense at this point how easy or hard? It sounded like you're really focused initially on websites, but things like social media things and so forth. How much work would it be for them to participate in this new way of doing authentication?

VITTORIO BERTOLA: Well, you really need to know how the platforms are implemented. If they are using OpenID Connect, I don't think it's too much of an effort. It's more like if they want to do this or not because for some of these companies, it's really strategic to keep control of the identity of the users. Technically speaking, we tried to choose the path which creates less work for people to join. If you have a website and want to accept these identifiers, of course, we're working on the libraries and so on, but in the end, it's an OpenID Connect line plus one DNS query, so even if you have to program the DNS query to add that part yourself, it's not that difficult. I see it more like is there interest in pushing this and making this happen, that's the real problem. I don't think it's a technical problem.

JACQUES LATOUR: Alright. Any other questions? No, that’s it. Oh, we have one more.

UNKNOWN SPEAKER: Hi, this is [inaudible], the German registrar. Yesterday, Jack had the talk where he wanted to have an identity power household and the nice thing was the root itself makes the domain automatically. If I look at my 17 year-old twins, they like to log in with Facebook because it's so fast and so easy, and the first step is to have your own domain registered within a few seconds and then run other identities with that, so do we have any ideas how to make this first attempt, the most easiest way and most convenient way to get the domain running? Any ideas to get my kids using that?

JACQUES LATOUR: If I may respond, I added that as a feature in my spec, so already. Yeah.

VITTORIO BERTOLA: All you need is to basically register a domain name, then there's the issue of who pays for the domain name, but if it's included in your service from the router, the router can just create a domain name. So, I don't see a problem.

JACQUES LATOUR: So who does the actual identity verification?

VITTORIO BERTOLA: If you mean who checks the password, it's identity authority. If you mean who checks the IMIs, so the identity is real, that's outside the scope of the protocol. This is an authentication protocol and the assumption is that all the data is self declared, so the only assumption you can do about the data is that I am declaring that this is my name which is exactly like it works today for any registration you make online. Then, we could have some hooks in the protocol so that you can add some third party signature or something that verifies --

JACQUES LATOUR: Like verified by whatever?

VITTORIO BERTOLA: Yeah. That could be the approach, but for most websites, I don't think you really need to prove that you are you, for most things you do online. In some cases you don't really want to prove that you are you. You want to have pseudonymous identities and this gives you the flexibility to do that.

JACQUES LATOUR: Thank you. That's a good presentation. Up next is Ondrej Filip from CZ.NIC and you're going to talk about Automated KeySet Management. [AUDIO BREAK]

ONDREJ FILIP: Good afternoon, everybody. My name is Ondrej Filip. I'm from CZNIC domain name .cz, Czech Republic, and I wanted to show you how we try to help DNSSEC grow in a new country. The current situation is not bad. We have roughly more than half of domains signed currently in the registry, but the true thing is that those are mainly the domains that are hosted on the same DNS servers that are run by the registrars because there is a very short path between DNS provider and registrar. Zero communication. Basically, it's just one entity.

We figured out that there's roughly 1% or 2% of other domains that are signed and they never publish DS records in the registry and there might be several reasons for that. Maybe they cannot just submit the key material, we just don't know. But we've talked to several DNS providers and they told us [inaudible] for example, they told us, "We cannot submit the material because the domain name holder is not able to do that. He doesn't know how to make it. They have all different registrars," and so on.

So, there is sub-optimal support from the registrars sometimes or the domain name holders do not understand DNSSEC at all so it's complicated for them and the DNS providers have no relationship with the registry.

Another motivation, which I praise, is to make DNS great again, and I apologize for abusing this old campaign slogan from the great President, Ronald Reagan. In old times when you created a DNS domain, you put it to name service, you just register domain, it'd just run forever. There was no need to touch it.

Now, with DNSSEC, you need to like resign it for all of our keys and everything. There's a lot of work you need to do and all of it is all very complicated. We wanted to get back and make DNSSEC very simple.

And last but not least, we are not just talking about the Czech Republic. There are 11 other countries using FRED and that's why we need to keep our registry feature regional. We are doing open source registry, many other countries download it and install to deploy it. We have a responsibility to help those countries as well, so also thinking about those other countries. So we are trying to make this as soon as possible.

So how to do it. There are [inaudible] standards, mainly two of them, RFC 7344 which introduced a new resource records like CDS and CDNSKEY and also the way how to work with them, and very recently this year, RFC 8078 which exactly describes how to [inaudible] and also how to do remove DS records if you really don't want to get rid of DNSSEC at all. So, those are very important RFCs for this game.

There is also a draft done by, I think you are author, right, Jacques Latour? Thank you for that. Which is for pushing the DS keys into registry. We are doing the opposite way [inaudible] which is mentioned in 8078.

But let me talk about two sides of the problem. We have a registry and we have a DNS signing software, signers. Here's a list of open source [inaudible]. Maybe there are some others. The key rollover and the whole system of publishing DNS keys and stuff like that, it's not supported fully in some of them. I think Open DNSSEC has planned for next year, if I'm not mistaken. I was told that from the [inaudible] Labs.

PowerDNS and Bind have some semi-manual publishing system. You usually need to run some scripts from chron or something like that so it's not fully automated, but quite close, and also Knot DNS has full support. This full support has been there since version 2.5. Now we have version 2.6, which is even better of course. This version is really recommended for that. On the other hand, the registry software is, as I mentioned, FRED and again, it has full support.

A little bit about Know DNS. It uses double signature KSK rollover. It has optional KSK submission via CDS and CDNSKEY. And also, it periodically checks for the existence of DS records via a set of configured nameservers. So, basically it can be either authoritative nameservers or validating resolvers. The nameserver publishes the CDSKEY and then it checks whether this DS record is in the parental or whether it is visible through the validating resolver, and if it is, it can perform the K rollover basically.

This is a configuration example. As you can see, you define the lifetime of the key, who checks the submission of the CDS record or the CDNSKEY record, validating resolver, and then you configure the IP address or the validating resolver, and that's it. Everything is automated. So that's the signer side.

Other features that are in Knot DNS. You can use CSK just single key for signing the rule zone, so it doesn't have to be split between CSK KSK. Also, you can share keys among more domains and Know also supports algorithm rollover which requires a little more steps than the regular rollover.

One thing which I need to mention explicitly. If you want to delete DS records, there are special types of CDS and CDNSKEYs, but this is not automated of course. This has to be done manually as well. If you publish those keys, then it means that the DS records should be removed from the registry.

Before we started, we discussed with the registrars of course and we showed them three options: either do not implement it at all, of course. You have always the option not to do anything, right? Or the Registrar will take care of it, so they will scan the old domains and publish the DS records, or the registry, us, will take care of it. Not very surprisingly, the registrars were not very interested in doing that, implementing something new and they rather said, "Yeah, please go ahead and do it."

So, now we manage the key sets, which is a little bit a change of the concept how the registry is managed, but we agreed on that and it seems to be working well.

Here is the architecture of the whole thing in the registry. So, we have something which is called cdnskey scanner, which is a C++ program implemented in C++ that uses the DNS library from [inaudible] Labs. Thank you very much guys; and it just reads a list of domains and then, you know, it checks how the [inaudible] would be distributed, not to overload a single name server, and then tries to create all the nameservers for the existence of cdnskey in this case.

Then there is the main manager which is called fred-akm which is a common line tool invoked from cron that does the whole logic. So invokes the CND scanner, collects the results and then, if necessary, does something in the right layer of Fred which is called fred-akmd.

The two upper parts are not register specific, so they can be used in the other registries. The last, fred-akmd, is just our specific, and actually, we publish everything as open source and the good news that somebody started to use the scanner. So, it's proof that this can be universally used.

`Now, how the scanner is working. We scan all domains in zonefile for CDNSKEY records. It takes about three hours, and while we're scanning, there are three categories of domains: those who had no KeySet before, like no DS records in the zone before, those that already had some automatically generated KeySetS, they were already in the system, and those that that legacy KeySetS, so A key set by the registrar and the reaction of [inaudible].

If the domain had no KeySet before, that's the most complicated and tricky part because you're boot strapping the secure process from an unsecure environment. So, we scan all the authoritative nameservers. We use TCP queries, and when CDNSKEY is found, we inform the technical contact that we found the CDNSKEY, that we believe that this is your signal to create DS records, and then we keep scanning for seven more days.

This is something that was described in the ISCs and we also believe that if someone has control over your zone for more than seven days using TCP then there is probably something fundamentally wrong, so we hope that this is safe enough to bootstrap the DNSSEC process. If the CDNSKEY record is the same for seven days, we create DS records and record in our database, and inform the domain holder via email and also the registrar via EPP. That's the first case.

The second case is that we find a domain with automatic KeySet. So, if you find a new CDNSKEY, and that's like a new key, which is quite updated and do nothing else, very completely automated. Nobody needs to be informed. If the KeySet is that empty one for deletion, of course we are going to notify the domain holder and the registrar that there is some change and a technical contact is informed anyway.

And the last option of that there is called the legacy KeySet, the old one, but again that means that we got the CDNSKEY record, we are secure there so there is no need to deny the process. If it's the regular key, we just create a new automatic key set and swap those. If this is the deletion key, we just remove the key set and we, again, inform the technical contact via email and also the domain name holder and registrar via EPP. That's kind of change in the domain because before they had statistic of key set and now they have this automated, so they need to be informed that something was change, but hopefully that's okay.

So, some statistics. We started roughly in June and since that time, we've had more than 600 domains managed this way, so it's at the beginning. We know there's much greater potential.

There were several peaks in the process when it started, then when we allowed to replace the manual set keys with the automated. Also, we signed some domains of ours in October this way. Of course, they were signed before.

So, there are several peaks, but you know, it had some user base at the beginning and we hope it's going to grow. But you can see that there were even some rollovers, so this system is live. Basically, again, if you are a domain name holder, you just install the DNS, just tell the domain name, sign my domain, and that's all. The system will find you. The system will bootstrap the DNSSEC, and then we'll just publish the DS records and you'll have a secure path to the registry, so pretty easy thing, and there's not much to do.

We are currently discussing with some of the registrars that they would probably use this system for the domains they have because it would remove some complexity in their software. They would just keep us doing everything, so they just run mainly not DNS with all domains and just let it go, no additional [inaudible]. Something like they used to before DNS came.

We have some other discussions around it. I would say, personally, I'm a little against, but there are some people who think that there should be some sort of opt-out mechanism. I was a little bit arguing internally so probably it’s going to need more discussion because if somebody publishes CDNSKEY record then it's a pretty clear signal that he or she wants DNSSEC. It's under debate.

One more thing that will probably increase the trust in the process is adding some more location for scanning. We currently have a single location which is just sub-optimal, we're thinking to extend it to more sites. Again, just to increase the security of the projects. We need to tune some of the notifications of the contacts. The first emails were very technical, which scared some of the domain name holders, so something that needs to be tuned of course.

Also, we would like to implement the PUSH model that was mentioned in the draft written by Jacques. Currently we have the pull model and we would like also the push model, so if Knot DNS or any other DNS implementation [inaudible] the keys, it should just push it through the registries and having some secure channel of how to do it, and it will happen immediately; and the DNS software would not need to wait one day for our scanner of course.

And also, do some kind of marketing, talking to DNS providers that this option is available and trying to explain that there's this easy way that will not increase the operation or complexity. That's all. Thank you very much. Are there any questions?

JACQUES LATOUR: Thank you, Ondrej. Well, you know I'm a big fan of all of this so, all in. Any questions? Yes?

UNKNOWN SPEAKER: Thank you, my name is [inaudible. I'm from the dot ID registry. I think we are interested in using something like FRED with the DNS second implementation. But sometimes I cannot find the source or I [inaudible] site for the FRED itself. Can I know the URL or something?

ONDREJ FILIP: I have very good news. The main architect of FRED is sitting in that part of the room [inaudible] and he is responsible for the site called fred.nic.cz where everything should be available. If it's not, just take something hard and try to convince him any other way to make it happen. You are very free to do that. I think that should be all the actual versions, but if there is something missing, let us know and I'm sure [inaudible], who loves this project, he will fix it quickly.

JACQUES LATOUR: Any other questions? [AUDIO BREAK]

I had a question myself. When did you expect to turn that feature on by default? So as soon as somebody installed KnotDNS, it publishes a CDS and the cycle starts like that?

ONDREJ FILIP: It is by default. If you register such a domain name currently and just run KnotDNS with the domain with the signing, after seven days, DNSSEC is on.

JACQUES LATOUR: So if all vendors supported this, then the adoption would increase pretty fast?

ONDREJ FILIP: Yeah I hope the other vendors will join us in this in [inaudible] and the authoritative domain name servers will support this feature because I think that would help the implementation of DNSSEC a lot because as I said, it doesn't bring any additional operation complexity, so that's something that will be very interesting for the DNS providers, so even every company that drafts their own, it's on the DNS way.

JACQUES LATOUR: [Inaudible]?

UNKNOWN SPEAKER: Just a small comment that you might see that there is about 600 or 700 domains. Maybe 90% of those domains are from [inaudible] because [inaudible] is one of the biggest supporter of this new way how to manage DNSSEC and in [inaudible] has several thousands, much more domains than this, but those people haven't yet clicked on the button, "I want DNSSEC," so right now we are negotiating with [inaudible] two things: how to market this feature against their customers that now they don't need to do anything else than to click the button, and the second thing was if they would be able to switch from opt-in to opt-out so anybody that will register [inaudible] domains will immediately have DNSSEC enabled. There is some negotiation right now that can be positive, but we will see.

JACQUES LATOUR: Okay, thank you. Probably a topic for the next DNSSEC Workshop. Any other questions? Thank you. Thank you, Ondrej. And Russ is going to talk to us about DNSSEC. How can I help?

RUSS MUNDY: Well, this is our ending session and I would first of all like to extend additional thanks to all of our presenters today and all of those that were asking questions and interacting. This is one of the reasons that this community has been so incredibly helpful and I think it's one of the reasons that DNSSEC has succeeded as much as it has. Thank you, thank you very much everybody. I appreciate it very much.

Can I get the clicker? Thank you. This probably will not take all the available time, but it's in a way, just a quick walkthrough of the various incendiary things that anybody in this room should be able to find themselves on at least one or more of these slides in terms of what you can do because there is a lot of cooperation in the community and we had a brand new research project described today by Wes, but there are a bunch of other on-going research things that are very useful and you'll see statistics show up a number of times, so in this case for TLD operators, which there's often quite a few TLD operators at ICANN meetings, we started off the session to hear 90% of the TLDs are signed, which is wonderful, but there is still more to do.

So if any TLD operator has not signed their zone yet, please look at doing so, and if you need help or input from other people, this is a great place to get it because there are a lot of people that are happy to share what they've learned. Then of course, be sure to look over your overall plan and be ready to accept DS records and DNSKEY records because that's how you move DNSSEC down through the hierarchy.

The next one, registrars. There are some areas where registrars have really stepped up and done an excellent job, but in many TLDs, they still remain a challenge. The registrars just simply aren't prepared, for various reasons, to do things with DNSSEC, so one of the reasons that some of these newer initiatives like we just heard about here from Ondrej, are very helpful but there's many places that just getting the registrars to engage and participate will make a big difference.

Statistics, statistics, statistics. There's a lot of people that really need and want to count and identify what is going on so that we can see how much progress is being made, and also a great help in terms of they key rollover things that are going on, especially in respect to the root. So, for Zone operators, just not a TLD operating any zones, whether it's in your house, Jeff Houston.

I would guess that this room may contain a higher percentage of people who are operating recursive resolvers in their own houses than maybe any other room in the world, but there's other places. Look at both professionally what you might be doing with your job or what you know of people in operating zones. Get them signed. Do your DNSSEC verification and tell your registrars you want support because many of them still don't, again statistics.

Enterprises. There are lots of things that can be done in an enterprise to get DNSSEC expanded and extended. At the enterprise level, we haven't got anything useful really to measure yet much yet at the enterprise level, but the sense is there's still an exceedingly small amount of DNSSEC used at the enterprise level. Look at how that would be something that could be done in your places of work and talk to your people in the security area and the areas that are worried about risk and risk assessment, how DNSSEC can help improve that.

And, when you're talking to ISPs, this is where the ISPs and their validation becomes very very important because most people don't run resolvers in their house. They rely on their ISPs and if they're not running one of the open DNSSEC validating ISPs, Google has been mentioned a lot and Verisign has some and there's other open validating resolvers that people will use, but it's even better if you can get your local ISP to do that. Every ISP has got zones themselves. They need to sign their own zones.

Get them fully engaged. Everyone of us here can do something to do more for yourself for DNSSEC and a large portion of the people that come to our workshop session love to get their fingers, if not into code, into operations and at least into configuring your own machines, building your own home environment and experimenting with it there.

I think a lot of times, lessons that end up being used and incorporated on a global basis, end up going through the IETF, but they start with ideas that come out of individuals heads. Some of them are when they are playing with their own networks in their own houses, and experimenting that way. We love to get more lessons learned, we are over 10 years on doing our DNSSEC workshops. We still have a lot of interest from a lot of people. We want to see this continue so we're planning on doing another workshop at the next ICANN session and we'll be announcing a call for participation on that probably late in December or early next year, and we want to hear lessons from everybody and see how much other help that provides to more people.

So, workshops, other meetings, RIPE is doing a lot in this space, there's a lot going on in Europe. I know Japan and JPRS are doing a lot. So in your local areas, look at organizing things and we heard some today that Rick Lamb has been in this area teaching courses and so forth, so that's a very good way to learn and participate. So thanks to everybody that was here and that was participating, and one more thanks to our lunch sponsors

Afilias, CIRA, SIDN. Let's give them one more round of applause. So we've had good support for this.

Like I've said, we've been doing these workshops now for over 10 years. They would not be possible without support from the SSAC which is one of the advisory committees of ICANN and the Internet Society's Deploy360 program. So, we have some pointers and URLs on here for people who want more information that they want to get, and I'd like to just give a special tanks, oh Julie has gone to another meeting, Andrew and Cathy who make all of this possible for us to do this. Thank you very much.

So, one more opportunity. An open floor for any questions, any comments, any thoughts anyone might have. Yes, Wes?

WES HARDAKER: The log-in button on my demo that I mentioned was broken is now fixed.

RUSS MUNDY: Thank you. Okay, everybody. Go get the count on Wes's machine now.

UNKNOWN SPEAKER: While we've got Wes, you said you thought informational was an odd status for 7706. What would you think it should be?

WES HARDAKER: That's a really good question. The reality is when I actually stopped to think about it after saying that I thought, "Well, maybe that is actually right," because the reality is it could be experimental. That would be a reasonable guess, but it's not really a protocol change, right? It's really just how you might deploy something in a recursive resolver, which you could have done before, this is just documenting the fact that it's an available solution to especially low-bandwidth environments. That's really why 7706 was written.

RUSS MUNDY: Okay, anybody else have any last minute thoughts or comments? Okay, thanks again to everybody for coming. We really appreciate it and hope to see many of you next time.

[END OF TRANSCRIPTION]