ICANN84 | AGM – How it Works: DNS
Sunday, October 26, 2025 – 10:30 to 12:00 IST

ULRICH WISSER

Hello and welcome to How It Works DNS. My name is Ulrich Wisser and I'm the participation manager for this session. Please note that this session is being recorded and is governed by the ICANN Community Participant Code of Conduct, ICANN Expected Standards of Behavior and the ICANN Community Anti-Harassment Policy.

Please observe the following guidelines to participate in this session. I will also post them in the chat for your reference. Only questions posted in the Q and A pod will be allowed during this session as time permits and when directed by the chair of this session. If you wish to speak, please raise your hand in the Zoom or otherwise as directed. When speaking, please state your name for the record and speak clearly at a moderate pace. I will now hand over the floor to Champika.

CHAMPIKA WIJAYATUNGA

Yeah, good morning, all, just for the mic. Yeah, so my name is Champika and I am part of the technical engagement team at ICANN. So this session, we are going to discuss about DNS, basically How It Works. And so, feel free to ask questions and we have remote participation people from remote sessions as well. You're welcome to ask questions as well. So, we will try to go

through the slides first and then probably we'll have some time for the Q&A later on.

Now, the assumption is that all of you are kind of new to DNS or do want to understand about DNS. So, we will start with the namespace. Now, when you speak of DNS, the domain name system, we have this terminology, the namespace. So, you can call it as a database if you like. And DNS is more of a hierarchical namespace. So, this is what you see over there.

So, you have the root on the top and then you have the Top-Level Domains. So, currently there are like 1,500 or so Top-Level Domains. Now, these Top-Level Domains can be, there are generic Top-Level Domains or what we call gTLDs. And then we also have ccTLDs, the country called Top-Level Domains.

And then in more recent times, oh, the last 10 years or 10 plus years, you've been having New Generic Top-Level Domains. For example, here you see things like .coffee or things like that. And as you may know that we are also having a new round of these New gTLDs as well. And then you may also see something here, which is actually more of a ASCII character set. This is basically, these are IDNs, the internationalized domain names.

So, you will have Top-Level Domains in IDNs as well. So, these are all delegated from the root. And then we have the second level nodes and then the third levels and so on. So, it's pretty hierarchical.

Now, going further, there are some guidelines when it comes to the domain names. Now we say that when it comes to the, say the Top-Level Domains, we can only have letters, digits, sorry, letters. In the Top-Level Domains, we can have letters from zero, sorry, from A to Z. And then in second levels, we can have letters, digits, and hyphen. We can't have digits and hyphen in the top levels because there can be some confusions with IP addresses and so on if you try to have some numbers or digits in the top levels. But in the second levels, you can still have letters, digits, and hyphen.

And then to the third levels and so on, you could do the same. And we can also have 63 characters. That's the sort of the length. Now, you can also see some definitions here. You know, when you try to define something called a domain, if you take a certain point, say in this case, you can see .com, and then there is also this dotted line that you can see.

So, anything below that particular node is what we call a domain. And even another example here, you can see anything below this co is where, co.uk domain. So, if you take a certain point, anything below that, that's what we define as a domain.

Now, then we also have this other term called zones. Now, there is some difference between the domain and the zone. So, domain is, that's what we defined earlier, the namespaces, whereas zones are more of administrative spaces. Now here, if you see all these dotted lines, they are like zones. So, they are more administrative spaces. So, when we speak of zones, we also have to think about

the delegation, concept of delegation, because when it comes to delegation, you are giving part of your responsibility to another party.

So, once you have given that responsibility to another party, then you are left with a space where you really manage or administratively you manage that. So, this is what you call the zone. And that's what we highlight here using this dotted line. So, you will see all the zones. So, you have the root zone, we have the .com zone in this example, we have the UK zone and so on. So, this one is example.com zone and so on.

Now there is also actually this terminology called FQDN. Now FQDN stands for Fully Qualified Domain Names. Especially if you are dealing with configurations and so on, zone files and so on, these Fully Qualified Domain Names would, you'll be dealing with it, especially when you define a name, you have to sort of, put the absolute path or you would end up with a dot.

So, this is actually, this last dot really refers to the root. So, in this example, this is a Fully Qualified Domain Name, www.example.com dot. So, this last dot is really referring to the root and that ends with a dot. Always, the other important thing is that in this hierarchical delegation process, we have parent to child relationship.

So, that's what you see all the time. So, root would be the parent and .com would be the child or .com would be the parent and then

example.com would be the child and so on. That goes all the way down. So, we have parent-child relationships.

Now then we get to the other important thing in DNS, the zone files and resource records. So, we have zone files and what's inside the zone file is basically DNS data. Now when we speak of DNS data, we have to have some certain format, how we are going to actually put this DNS data. This is where the resource records comes into the picture.

So, we have this thing called resource records or simply we call it SRRs. Now these are, some of the sort of most common resource records you'll find but there are quite a lot of these resource records. If you want to know more about the resource records, you could go to IANA website and IANA website would list all the resource records.

These are some of the popular ones. A record is to define IPv4 addresses. AAAA records, we have it to define IPv6 addresses. We have NS records. This is to define name servers. Then SOA resource record, this is what we call start of authority. We use the start of authority or SOA record to define the administrative information about a certain zone. So, this is sort of also called the zone apex because it goes to the top of the zone and that's the first resource record that we would define in a zone.

MX record, we use to define the mail exchanges or basically mail service and so on. So, these, as I said, now these are some of the important resource records that we see in a zone file.

Okay, going further. Now, earlier I was talking about delegation. Now, when it comes to delegation, you're giving part of your responsibility to another party. This is what we are doing. So, in the case of in this example, the root has delegated .com. So, .com has been delegated. And then we could go further.

So, when we do this delegation, say from root to TLD, you have to define these sorts of statements or these resource records or these delegation records in the root zone, so in the parent zone in this case. So, in the parent zone, you would find com zone. So, this is the administrative space that we were talking about, com, and that's been delegated to these name servers.

So, com is running, the com space is being kept in these name servers. So, there are 13 of those in this case. We have a.gtldservers.net. Dot, if you see always these are fully qualified domain names that ends with a dot. And then so a.gtldservers.net, b.gtldservers.net and so on. So, the .com zone has got 13 of these name servers. So, the com zone is being delegated to those. So, these kind of statements or records has to be there in the root zone.

So, when you speak from root's perspective, what root would be doing? Root would be delegating all these Top-Level Domains. Now earlier we spoke about thousands of Top-Level Domains that

exist today. So, all those delegations have to exist in the root zone file.

So, if you go down through the root zone file, you will see delegations for .com, .net, edu and all the gTLDS and then the country code Top-Level Domains and so on and so on. So, that's what you say or you see in the root zone file. And we go further down. Now the next level, we should do the same thing. So, if you want to delegate, say example.com from com zone, you should do the same thing in the com zone.

So, you will have to have a delegation for example.com, again, we have the fully qualified domain name here and that is delegated to these name servers. So, these are the name servers that keep the name space of example.com, which is in our example, that's ns1.example.com, dot, that's a fully qualified domain name, ns2.example.com and then there are two other name servers, a.gtldservers.net and b.gtldservers.net.

Now there are some certain concepts would come in here. There's something called Glue Records. So, if your name servers are in the same hierarchy. So, in this case, you can see that these two name servers, out of those four name servers, ns1.example.com and ns2.example.com, they're in the same hierarchy. So, you will need what we call Glue Records.

So, basically you have to have A records, or IP address records, something like this. Now these are IPv4 addresses of ns1.example.com and ns, this should be actually

ns2.example.com. So, these IP addresses have to be glued in. So, these are what we call Glue Records. Okay, and going further.

So, what about the root? Now we spoke about the delegations. Now let's go through this hierarchy. So, earlier we discussed top of this hierarchy is the root. So, what about the root zone? Now when we speak from the root zone management, now first of all, you have to understand, okay, we talked about the root zone. What is there in the root zone? These are the delegations for the Top-Level Domains.

So, we have all these Top-Level Domains, .com, .net, and so on. So, all those delegations, if you remember, these ns records and the glue and so on, these has to be in the root zone. Now, who keeps the root zone? Where do we keep this root zone? So, the root zone is kept in servers called root servers. So, that's the concept of root servers. So, we have root servers that keeps the root zone.

Now, who is going to manage the root zone? So, that's where the root zone management comes in. So, we have this IANA functions operator. So, we have IANA. So, IANA is a function within ICANN. And that is managed by an entity called PTI, or Public Technical Identifier. So, that's a function within ICANN. And PTI is an affiliation to ICANN.

So, if we have to modify the root zone, IANA would do that. And how that process happens, say, for example, if TLD wants to do some change to the root zone, say, if they want to make a change to their name server, or if they want to do some change to their IP

address, or something like that, a change request has to be sent to IANA.

So, the TLD manager would submit a change request to IANA. And then IANA would actually do all the checks. They would validate this, whether it's coming from the right TLD, and right party, and the right contact, and all that. So, the IANA would do all that validation process. And then once it is all okay, this will be actually, to update the root zone, this is sent to something called a root zone maintainer.

So, this root zone maintainer is the VeriSign. Currently, it is VeriSign. So, VeriSign would do all those updates, and actually update the root zone. So, what would they do? They would update the root zone database, and then they would create the root zone file, and then publish the root zone file, and then transfer this root zone file to root server operators. Then we have this term called root server operators.

So, who are these root server operators? So, initially we spoke about the root zone file. Root zone file is managed by IANA, and then send it to the maintainer, which is VeriSign, and then send it to the root server operators. So, why do we need root server operators? So, root server operators are the ones, actually, which would give this root zone file to the resolvers out there, or basically to the users, the whole internet, basically. And root server operators will put the root zone file into the root servers that they operate.

So, that's the sort of job of the root server operator. So, they would distribute the root zone to all these root servers that they operate. Now speaking from the root servers' point of view, there is a certain naming convention of the root name servers. So, because every of these root server has to have a certain name. Now there is a naming convention for that. So, the naming convention is there is a letter, and then you also have a certain domain, or a naming convention for that root name service itself.

So, the naming convention is the letter.rootservers.net, and the fully qualified domain name would be the letter.rootservers.net. So, that's how you would see a name of a root name server. And then this letter would be something like A, B, C, and so on. So, currently we have these 13 root names, so starting from A to M. So, we have A.rootservers.net, B.rootservers.net, and C.rootservers.net, and so on. So, altogether we have 13 of those.

And these are the names. And then who would operate these names? They are the root server operators. So, we have A.rootservers.net is managed by a certain operator, B is managed by another operator, C is managed by another operator, and so on. So, currently there are 12 operators because one operator, basically VeriSign, so they manage two of these letters.

So, A and J is managed by, or operated by, VeriSign. And then if you take, say, for example, B, B is operated by ISI, right, and so on. And C is operated by Quotient and ISP. Similarly, if you take ICANN, ICANN is also a root server operator. ICANN would operate L. So,

there are different operators who would operate all these different letters you see here.

So, this is the sort of, how the root zone management is being done. And as I said earlier, if there is a change request comes to IANA, IANA would do that change, do all the verification checks, do that change, and then this change root zone file will be basically, the maintenance, root zone maintainer is VeriSign. And once they have done that, they would actually distribute it to all these root server operators.

And then root server operators would then actually distribute to all their instances. Because every of these root server operators, they also have their specific instances, basically, having that same letter. So, if you take, say, L, the ICANN, so L has got hundreds of instances scattered around the world, and that ICANN operates. And similarly, if you take F, the ISC, they have, again, hundreds of instances, they run around the world, and they carry the same letter F.

So, all these root servers, they run on AnyCast, based on AnyCast technology, right, in the routing. So, that's how the root servers would operate, and the root zone is managed. And then we have all these resolvers out there. There are millions of resolvers out there in the internet, and all these resolvers would talk to these root servers based on the AnyCast. So, basically, the most optimal root server will be contacted, most optimal root server instance will be contacted, and get the root zone file, or basically, the

appropriate response from the root. So, that's how the root zone management would happen.

Then we get to the DNS as a service. So, what are the components that we have to think about here? Now, if you think of, say, from client's perspective, now client would have different devices. It could be a laptop, it could be a mobile device, it could be, any IOT device, your refrigerator, whatever. Behind the scene, we do all these DNS queries.

So if the client device wants to, say, send DNS request, or you could be sending an email, or you could be doing some browsing, or something like that, then the application, right, the client application needs to know the IP address, because, when we speak of the protocol, going to the protocol level itself, we are dealing with the TCP IP protocol. So, in TCP IP, we have IP layer. So, if you want to get the packets from one host to another host, that happens in the IP layer.

So, the IP address, we have to know the IP address. And DNS is there as the translation. So, DNS is in the application layer, and so we have to have some translation there. So, from application's point of view, application has to know the IP address. So, application would talk to the operating system to ask, asking for the IP address itself. So, that's where we have to have what we call a stub resolver in your operating system. So, the application, we will have the API calls to the stub resolver, and then the stub resolver would send a request to what we call a recursive resolver,

asking, basically, it has to send that question that the client ask to the recursive resolver.

So, the stub resolver sends the request to the recursive resolver, and asking for the same question that the client ask. Say, if you want to browse www.example.com, we have to find the IP address of www.example.com. So, this query goes to the recursive resolver. And then, the recursive resolver has to now send this query out to various other servers out there on the internet, and then find the answer so that this answer can be given to the client.

Now, when it comes to the recursive resolver service, now, this service can be run by different operators, different parties as well. So, say, for example, mostly the ISPs, the internet service providers, they provide this service. They run the recursive resolvers for their clients, for their customers. Then we also have, like, mobile operators. They run recursive resolvers. Then we have public DNS resolvers. For example, Google, Cloudflare, and so on, they run public resolvers. And lots of organizations, they do run public resolvers.

There could be some corporate resolvers as well. Universities, they run their resolvers for their users, and so on. So, there are lots of parties who run these recursive resolvers. So, whatever the case, the client sends the request to the recursive resolver, and then the recursive resolver will have to then find this answer, and then the recursive resolver, of course, can also forward this request to another set of recursive resolvers as well. So, this is where, things

like caching forwarders would also come into the picture. And whatever the case, the question has, or the response has to come from servers called authoritative servers, authoritative DNS servers.

Now, if you relate back to, earlier we spoke about, our zones. We spoke about zones, the resource records, and all that stuff. So, all those zones are kept inside the authoritative servers. So, even if you think of root servers, root servers are authoritative servers for the root zone. And similarly, if you're talking about TLD authoritative servers, they are authoritative for the TLD zone, and so on.

And if you are running a second level authoritative server, say example.com, you are authoritative for example.com zone. So, you would have to run authoritative server for that. So, those are the ones that we call as authoritative servers. So, the recursive servers will have to talk to these authoritative servers, and get a response from the authoritative servers, because, they are the ones who are authorized to give reliable responses for that particular zone. Then get the response from the authoritative server, and then this response has to be then returned to the client, because, client is the one who asked for that question.

So, that's basically, what happens in DNS, basically in a query resolution or query response. Now, when this query response happened, let's see how it happens. So, what's the process? So, we have the client. You can see the client here, and then client want

to visit, say, www.example.com. That's this web server that the client wants to visit. Now, this web server could be out somewhere in the internet.

At this point of time, the client doesn't know where it is, and that's why the client has to know the IP address of that web server, right, so that the client could get to that web server and download the web pages and so on. So, client has to know that IP address, so this IP address has to be then sent to, as we discussed, to the recursive resolver.

So, the first step would be, we have a query going from stub resolver to the recursive resolver, and then next would be, okay, how the recursive resolver is going to find that answer. Now, recursive resolver knows that DNS is a hierarchical system. We have a hierarchical database, and top of this hierarchy is the root. So, we also spoke about, the root and where these root servers are and how many of those and where it's scattered and so on. So, there are like nearly 2,000, close to about 2,000 root server instances.

Earlier, we also spoke about those, naming conventions and so on. So, the query would go to root zone. I mean, basically, we have to get to a root zone. We have to get the nearest root server, the instance. So, this is where we can find it through the AnyCast. So, the query would go to the nearest authoritative server for the root zone, and then we ask the same question, give the IP address of www.example.com. So, that is the question that the client has.

So, recursive resolver can send this question to the root, and then, but the root wouldn't really interest about, answering the whole query itself because, root wouldn't know that because root only knows where the Top-Level Domains are because that's what root has really delegated. If you see the root zone, the contents of the root zone file, it's all the delegations, that's what we have.

So, the root would respond saying that, okay, I don't know where is www.example.com, but I know where is .com. So, the root would respond with the name server details of .com. So, it will return the name servers, if you remember those Glue Records. So, it'll return those NS records for .com.

And then the recursive resolver has got the NS records for the .com. So, it knows the IP addresses now, the Glue Records. So, it can go and talk to the name servers of .com. So, it will send the same request now. So, the query is basically give the IP address for www.example.com. So, the query would go to the .com servers and then .com will say, okay, I don't know the IP address of www.example.com, but I know where is example.com because, example.com is a child zone of .com. So, there's a child, parent-child relation there.

And then .com would respond. If you remember this earlier, right, we had these records, the Glue Records as well. So, these records will be returned to the recursive resolver. And then the recursive resolver gets this response, and then it can go and talk to the

example.com because it got the IP addresses, send the same request, give me the IP address of www.example.com.

Now when the example.com name service contacted for this, now this information is in the zone of example.com because, that's where this particular data is. So, the example.com name service can respond with that IP address. So, it'll respond with that IP address, and this IP address will now get to the recursive resolver. And then the recursive resolver will do something, it's going to keep this answer in the memory or in the cache.

In DNS terminology, we call that the cache. So, it'll keep this answer in the cache, and then it'll send this answer to the client. Now client got the answer. Once the client got the answer, client has got the IP address, now once you have the IP address, you can establish a connection between yourself and the web server. You can get to the web server, and then you can download the webpage and so on. You can, once the connection is established, they can have that communication.

So, the DNS is there, the resolver service is there to find that IP address and give you that IP address or give that client the IP address. So, that's the service that we are trying to expect from the DNS here, okay? So, in this service, the important components that we have been dealing, we have the stub resolver that is in the client, we have the recursive resolver. That's, again, an operator would provide this. And then we have the root servers, the authoritative servers, the top of the DNS hierarchy. Then we have the TLD

servers, these are also authoritative servers. Then we have the second level servers, they are also authoritative servers who would provide those answers.

Now one of the other important things that you also have to know, the recursive resolvers, when they give the answers to the clients, this answer is a non-authoritative response. The answer is a non-authoritative answer. Because, recursive resolver is just, someone who is in the middle and giving this answer, getting it from someone else and giving it to the client. So, the answer is not really authoritative. If you want to get an authoritative answer, you have to go and directly talk to an authoritative server. And directly get it from an authoritative server itself.

The answers what you get it from resolvers are non-authoritative answers, okay? All right, so that's how the resolution happens.

Now earlier we spoke about then caching. So, why caching is important? Now I mentioned you that when the recursive resolver got the answer, now all these answers are kept in the memory of the recursive resolver, or in other words, in the cache. Now for how long? This is where we have something called TTL, so Time To Live. Now this TTL can be decided by the authoritative server administrator.

If you are administrating the authoritative name service, you can decide, okay, what sort of TTL that you want to come up with, right, for that particular record. Remember, we have the resource records. For those resource records, you can define the TTL. Now

say, for example, from root's point of view, we have all these delegations. We have NS records. We have delegation nsrecords.com. We have nsrecords.net and edu and all that, country codes and so on.

So, in the root zone, the TTL is defined as two days. So, when the particular NS record for that TTL gets into a resolver, it will be kept there for two days. So, for two days, this answer will, you know, you really, the recursive resolver doesn't have to go and talk to the root server if it is from the same TLD. If the client's asking about .com always, that will be there for the two days. Resolver knows about .com for the next two days because it got that information from the root.

So, the root servers are not the other important thing. What you have to remember here is that for every DNS query, we don't have to go and talk to root because the resolvers know that. Same goes with the other responses as well, other TTL values as well. I mean, I'm not talking that every record will be there for two days. It's up to the DNS administrator how you can define your TTL values.

Okay, so caching will improve the DNS response time because if you have caches, you can improve the DNS response time. But also, from the other side of it, say from a different perspective, if you have very large caches, say if you have very large TTL times, you could be having say one week or one month, that means there is some possibility of having some errors there. Because if the resolver is going to keep that cache for one month, and within that

one month, if you want to do some change to your DNS data, that will not come into a resolver.

So, for the whole one month, then the resolver would be responding with that old information. So, this is where you have to have some balance between the accuracy and then the response time as well. So, you have to actually work it out and see what's the best optimal TTL values for your particular zone.

Okay, so that's what we say from a DNS service point of view. The clients will always ask questions from the DNS servers, from the resolvers, and the resolvers would send these questions to authoritative servers and get the answers. And these answers will be given to the clients, and these answers will be kept in the caches, okay?

Okay, next question is, is this all secure. Now we've been going through all these queries. We know we can ask the resolver for a question, DNS query, and then we can get a response from the resolver as well. But are we secure enough? Okay, if I ask you, is it secure? Yeah, I mean, you might say yes, you might. But then, yeah, maybe. Because sometimes there could be some problems. So, what sort of problems that we could have here?

Now, some of the problems could be maybe someone could stay in the middle and capture these DNS packets and change those DNS packets. Change the contents, the information, the metadata.

Now, by the way, all these DNS queries are going in plain text. That's how the protocol is designed.

So, if you capture all these DNS packets, you can see all that metadata related to that DNS packet, we can work it out. And that can be changed also. So, this is where there are few things that we have to really think from a security point of view. Of course, from a security point of view, there are things that we have to think, say, the confidentiality, the privacy aspect of the DNS data. Then there is the integrity. Whether the data is changed in the middle, whether someone can modify that in the middle. And then there is also the authenticity aspect.

So, authenticity is basically, are we getting the response from the right server? Because this recursive resolver is sending the queries to the root server. So, we have to know whether are we really getting the response from the right root server? Or is it really the root or someone else impersonating as the root? Or are we talking to the right TLD? Or are we going and talking to fake TLD, for example, or for that matter, for a fake authoritative name server? So, these are the questions that we have to ask. So, these are the problems that we have.

Now, so the weakness, as we discussed, there can be always some bad actors. So, they can stay in the middle, they can change, they can impersonate. They can pretend to be an authoritative server. They can give wrong responses. They can even pollute these caches. They can send some wrong responses to these caches. So,

if some wrong response is going to a cache, the caching servers, the resolvers, they can keep this wrong information in the caches, and then they can respond to a client with the wrong information.

So, the client would end up in going to a wrong place, wrong website. So, these are some of the weaknesses that we'll find in this DNS ecosystem. So, you'll find all these weaknesses. So, in this case, this is just an example where a bad guy who is capturing this data and then changing this information, say, in this case, the IP address. So, the right IP address is supposed to actually not have a different one, but then it's actually sending a wrong IP address to the recursive resolver, and then the recursive resolver, in turn, sending this wrong IP address to the client. And then client is going to go to a phishing site in this case. So, it could be your bank website, and it looks like the same as your bank website, but it's not really. It's a different IP address.

Now, this can be done in various ways. You could have a malware. Maybe the client could be directed to a malware site. So, the client would click on a link, and client would get to a malware site, and client would then download that malware into your device, and then the malware could change your DNS settings, for example. Or even malware could change the resolver service.

So, typically, we are talking to our ISP resolver or mobile operator resolver, but then these DNS settings can be changed by the malware itself, and then you could go into a resolver that is managed by the attacker as well.

So, these are all some weaknesses that we find in this DNS ecosystem, and also in this resolution process. So, this is where we need some security. This is where we need some solution, some protection there. So, this is where we have to protect our DNS data.

Now, we have DNSSEC. There's a security mechanism called DNSSEC that stands for DNS Security Extensions. Now, DNSSEC would protect your DNS data. It doesn't mean that deploying DNSSEC will protect the whole DNS ecosystem, ecosystem security you can achieve with DNSSEC. That's not the case, because when you think of the whole DNS ecosystem, there are many components in it. Earlier, we discussed about it.

So, there are resolvers, there are authoritative servers, there are transactions happen between these authoritative servers, but DNSSEC is very specific. Now, DNSSEC, what it would provide you is the protection from data perspective. So, you can protect the DNS data. You can secure your DNS data. And also, you can achieve two important security objectives, like what we discussed before, the authenticity and the integrity.

So, authenticity is, are you talking to the right DNS server? And then the integrity is that these answers that you're getting from the DNS server is not modified in the middle. So, these two things we can achieve from DNSSEC or DNS Security Extensions. Now, the other aspect we mentioned about is the confidentiality aspect, which is the privacy. I also told that, DNS data goes in the plain text, but DNSSEC will not, of course, provide that confidentiality aspect.

If you want confidentiality, if you want DNS privacy and things like that, then you have to encrypt.

So, you have to have DNS encryption from client to the resolver, but DNSSEC will not provide that. So, DNSSEC will provide basically the authenticity and integrity of the DNS data. So, now you can also have some comparison here. For example, say without DNSSEC, the recursive resolver will directly go and talk to the authoritative server and get the response from the authoritative server, but recursive resolver is not really, no, it doesn't know whether this data is valid or not. It could have got some fake answer, and this answer could go into the client and client could go and talk to a wrong web server. But with DNSSEC, the recursive resolver will do some validation, will do a check whether this data can be validated.

Now, this is the sort of conceptual view of DNSSEC. Now, there is quite a lot of technicalities goes into the DNSSEC deployment, the implementation, because what we are trying to do here is to digitally sign the DNS data. So, earlier we spoke about authoritative servers, the zones, right, where we keep all those resource records, the zone files. So, we digitally sign those resource records. So, earlier, I mean, without DNSSEC, we only get a resource record. So, you go and talk, recursive resolver goes and talk to authoritative server, get a A record or get a COD A record. And then give this COD A record or A record to the client. So, that's it. But now when you digitally sign that, there are signatures that comes along with these A records or COD A records.

So, if I'm a recursive resolver, when I go and talk to an authoritative server, authoritative server would also send me not only the A records and COD A records, but also it will send me the signatures or the digital signatures of that. So, once I get these signatures, I can validate whether these signatures really belong to the right party. When I say right party, that means the right authoritative server.

So, for me to validate that, I will need the Public Keys associated with that zone. So, if I want to actually get that Public Keys, whoever the zone owner, zone administrator has to publish those Public Keys in the zone. So, there is this whole Public Key cryptography will come into the picture now. And you have to have private Keys and Public Keys because we use asymmetric Keys in DNSSEC. And then you use your asymmetric Keys and basically you publish your Public Keys, you keep your private Keys safe. And then whoever the resolvers, once they get the signatures, they can use those Public Keys that are published to validate those signatures.

So, that's the sort of concept behind DNSSEC. But once you implement DNSSEC, the resolvers, they can do validation. They can give the correct answers to the clients.

Now, when it comes to the DNSSEC, there are two important aspects of DNSSEC. One is the signing aspect and the other is the validation aspect. Now signing is done at the authoritative server level. If you remember that whole resolver diagram we had, all those authoritative servers, the root servers, the TLDs and second

level servers and so on, all those authoritative servers, they have to do the signing part because they are running their own authoritative data. They have to do all the signing. Whereas the validation, it is happening in the resolver level. The resolver operators, they have to do the validations.

So, if you, earlier also I was giving you some examples of different type of resolve operators. We have public resolve operators and so on. So, most of the public resolvers now, they enable DNSSEC validation. So, for example, if you're using Google or Cloudflare and so on, your DNS queries or the responses that they get, they have been validated. They will only give the answer to the client if the response can be validated.

If the response cannot be validated, the answer will not be given to the client. The response will be dropped. So, client will not get response. So, basically the whole idea is that the client will not go to a wrong place. Okay. So, that's some very quick overview of DNS security extensions or DNSSEC.

Now, this is more of a broader view of security challenges we have in the DNS ecosystem. I would call it as a DNS ecosystem because there are so many parties who are involved in managing different components of the DNS. Now, you see many of these things what we discuss now. So, we have stub resolvers running in our devices and operating systems. Then we have recursive resolvers running in ISPs and public resolvers and corporate resolvers and so on. And then we have authoritative servers.

And also, actually in that authoritative servers in that layer, we also have multiples of authoritative servers. That's where, you have to think about the redundancy aspect because you can't just have only one authoritative server and just forget everything. You can't do that. You have to have redundancy. You have to have multiples of authoritative server. You have to have minimum of two authoritative servers.

So, this is where you have primary name servers, secondary name servers, and multiples of those secondaries and so on. You can also have any cast servers as well. So, you have redundancy built in there.

Now, between those authoritative servers, say if you have your primaries and secondaries, you also have to actually make sure your data is all synced, synchronized between your primaries and secondaries. If I'm the primary and if Ulrich is my secondary, both, we have to have the same zone data. We should be in sync.

If I do any update to my zone, that update should go to Ulrich as well. So, basically, I have to have some synchronization between myself and Ulrich and Ulrich could, whenever I do some update, Ulrich should come to me and get my updated zone data into his zone file. So, his zone file should also be updated because, ultimately, we could be in different places. He could be in one network, I could be in another network.

We should not be in the same network, same subnet, same data center, same rack and so on. If all, if our name servers are in the

same place, then this is not a good design. It's not a best practice because there could be some natural disasters. We could be doing some maintenance for the same subnet. If you're going to have some name, same name servers in the same subnet, then that could go offline at a certain time.

So, that's where we have to actually have some diversified infrastructure. And then these diversifications can be achieved in different levels. It could be in subnet levels, it could be in different, data center level, different networks level, different ACE levels, different even geographical level as well. It could be in different countries as well.

So, these are all security aspects that we have to think about. Otherwise, if there is some failure, you are prone to have much more bigger failures actually if you don't consider these aspects. Yeah, so we have authority servers, secondary servers, and then we also have this whole ecosystem where we have registries and registrars. The ecosystem.

So, registries, registrars, they should also be synchronized. There are some protocols that we run to make those synchronized as well. So, there is this EPP, the Extensible Provisioning Protocol, to make sure that they are synchronized and secure. Now, from client or the registrant perspective, say for example, if someone gets a domain name, if you get your own domain name, you would go to a domain name registrar, could be a reseller, but again, reseller

would be dealing with a registrar. So, ultimately, the registrar has to update the registry database and so on.

So, we also have to make sure that we provide a very secure interface to our clients or the registrants. Secure portal access to the registrants because if that portal access is not secure enough, there are always possibility of credential thefts and credential compromises and so on.

So, that's where many clients, many registrants would lose their credentials and then the attackers, they could get into the portals and if they can get into their portal, they could change the name servers or they could change the mail server information in the registrar portal. And if these things are changed, much more harm could happen to your domains. And to make it even worse, if the attacker managed to transfer the domain from one registrar to another registrar, it could be even worse.

So, the registrars, they should also have some things like, registry locks and so on, mechanisms so that they could protect that infrastructure as well. So, these are all where you find all these red crosses in this whole diagram. That's where your DNS can really be, attacked or you could have some threats. So, this is where you'll find security challenges and this is where you can actually try to protect your DNS infrastructure.

Now, again, this is why we are also trying to say that this is more of an ecosystem because one person cannot do everything. Or one party cannot do everything because these are all different

responsibilities. There are resolvers, who has to actually have protection there. They have to have things like, proper, mechanisms we discussed before, even DNS second enable, they have to have rate limiting, they have to have proper DDoS controls and things like that, all that.

And then from authoritative server levels, they have to have proper mechanisms, make sure that you have your transactions are protected between your primaries and secondaries. You have to have some authentications. It could be IP-based authentications. It could be Key-based authentications. You have to have those proper access controls, basically.

And then zone files has to be protected as well. And then credential management is also another big area because when these registrants, their credentials, as we discussed, it can also have quite a lot of compromises. So, credential management is something that is very important. We have to have some two-factor authentications. Also, we have to manage, if you are registrar or registry, you have to manage the credentials of your registrants in a proper way. So, these are all important things that we have to think about in securing our DNS ecosystem.

Now, this is a very quick sort of snapshot overview of various security mechanisms we can implement, a summary of those things that we discussed. There are protocols that we can deploy as well. Earlier, I said that, DNS traffic that goes in the plain text itself. So, if you want encryption, DNS encryption, there are

protocols available. There is DOT, that is DNS over TLS. There is DNS DOH, DNS over HTTPS. So, these would help you to encrypt your DNS at this level. Then there is DNSSEC validation. There is also QNAME minimization. This is also a good practice.

QNAME minimization is, when I was going through that DNS query resolution process, I said that, okay, you're sending this DNS query to root name servers. But actually, root name servers do not need to know the whole DNS query itself. Root name servers only need to respond about the Top-Level Domains.

So, we can take out the other parts. You can only send the Top-Level Domains, the query that is relevant to the root name servers. So, that's what we do through the QNAME minimization. And then DNSSEC deployment, registry locks, even the TLS certificates, and all these things are important things in terms of protecting your DNS infrastructure.

I should also mention at this point, we also have this best practice or a framework that you can look into and follow that is called KINDNS. I don't have any slides here on the KINDNS, but you could always go to kindns.org. It's K-I-N-D-N-S.Org. We pronounce it as kindness.

So, KINDNS is a framework or a tool that you can use to evaluate whether you are doing the right best practice in your DNS ecosystem, in your infrastructure, and you can follow these KINDNS practices and deploy those things in your infrastructure.

So, there are different KINDNS categories that you can, if you're operating, say, authoritative name servers or if you're operating recursive name servers. Again, these categories also have some subcategories as well because authoritative server operators also can be critical. You could be managing some critical zones.

You could be managing some TLDs, which are also critical zones, or you could be managing second-level domains as well. So, in these cases, sometimes you may not need to have all the practices because, of course, the critical zones has to have more rigorous sort of checks in terms of security practices that you apply, whereas second-level domains you may not need to go into that level, that extent, so you could have some certain other practices that you can apply in your second-level domain zones.

So, KINDNS will give you guidance of all those practices that you can apply. And in a similar way, if you're managing a resolve operator, if you're managing resolvers, you could also fall into different resolve operation categories. You could be running private resolvers like corporate resolvers. You could be running some ISP-type resolvers or more kind of shared resolvers, or you could be also operating some public resolvers as well.

Depending on these, you could apply some different best current practices as well. So, I would, at this point of time, maybe I would also encourage you to actually follow KINDNS best practices and apply those in your DNS ecosystem so that you can protect your

DNS as well. I think this is what mainly we wanted to mainly highlight in this session.

So, we spoke about DNS in general, and we spoke about the DNS hierarchy and zones, the resource records. We also spoke about the root servers and also the root zone management aspect as well, how the root zone is managed and IANA functions and so on. Also, we spoke about the DNS security aspects as well and the query resolution process, the caching, and the authoritative servers, how the queries resolve, caching happens and so on, and then also the securing aspect as well, like DNSSEC and other security mechanisms available and looking at the whole DNS picture in the high level and see how we could protect the whole ecosystem and then, of course, applying the best practices through KINDNS.

So, I think at this point of time, probably we can open up for any questions or discussions or even any feedback. Thank you.

ULRICH WISSER Any questions? Yes. Mic, please.

YANNICK [ph] Thank you for the explanations. I'm Yannick from Japan. One of the difficulties we've got on our side, it's the impact of the IDNs on the system. On the first slide, you had a TLD which was coded, and the coding in itself means nothing. So, what is the difficulty to use an actual foreign character for the system? Thank you.

CHAMPIKA WIJAYATUNGA Yeah, so let me get to that slide first. Yeah, so this is what you were talking about, right? Yeah, so okay, now stepping back to a few more pointers. Now, when the DNS was designed, it was all designed in a more -- the queries were going in ASCII format. So, if you think of a DNS query, it goes in ASCII, and the DNS servers understand all the ASCII stuff.

Now, that's when, of course, we could use our usual ASCII characters to send the data and store data and so on. But the thing is that that's where you have, if you think of your keyboard, we use ASCII. So, in ASCII, we can only have, say, two to the power of seven, I mean, because it's one byte. And the first bit is more of a control bit. So, we can only use seven bits, which is actually, we can have two to the power of seven combinations, which is 128. We have 128 characters.

Now, if you try to assign, say, English alphabet, simple letters, capital letters, and all these other characters in your keyboard and everything, we have pretty much used all those 128 characters. Now, when you try to, say, bring in more characters, say, Japanese or Korean or Chinese and all the other different characters and so on, we don't have enough space over there. You know, it's more than 128 characters.

So, this is where we need another different type of encoding, right, more than ASCII. But we can't also actually just get rid of ASCII because, we have been using ASCII for all these years. So, that's

when we have to have some new or a different encoding scheme. That's when the Unicode came through, and there are different encoding schemes we use based on Unicode.

So, we have an encoding scheme called UTF, Unicode Transformation Format. And again, in UTF, we have UTF-8, 16, 32, and so on. But we are using UTF-8 as the standard sort of code. So, what you see here is actually, if you take any characters that we were talking about from Japanese point of view, right, if it's in Japanese, DNS would not understand all those Japanese characters. So, it has to convert that into an ASCII, basically ASCII character.

So, what you see here is that conversion from the Japanese to the ASCII. And this conversion is what we call the Punycode. That conversion is called the Punycode. So, always the Punycode would start from xn-hyphen-hyphen. That's what you see over there, xn-hyphen-hyphen.

Now, ultimately, this would convert this, let's say a Japanese word. So, that Japanese word, it may not be exactly in terms of storage and in terms of byte. It may not exactly ASCII because, ASCII character could be one byte, but Japanese character would be two bytes, three bytes, and four bytes, and so on. But then once it is converted, you will see this whole string, which is what you see in ASCII, basically. So, that's the equivalent ASCII or the Punycode for that top-level domain, which is in a different script.

So, this is the sort of, explanation behind that IDNs. Now, we have IDNs in Top-Level Domains, and currently about 60-odd country-code Top-Level Domains have been delegated in UTF characters or in IDNs, basically, 60-odd country-code Top-Level Domains. And then in the New Next Round and so on, you can still get the gTLDs also in UTF-based characters as well. So, but the DNS query still resolution happens all based on ASCII. It was a long response, but yeah.

ULRICH WISSER I just wanted to add that Unicode at the UTF-8 encoding is not a single encoding for the same character always. So, you can, for example, you have the, I'm a German, we have these umlauts, the A with the double dots, and there is actually a Unicode code point for that. But there's also, you could have only the two dots and the A and combine them. And I mean, for a human, there's no difference, but for a computer, so there's even steps involved in this to make it a defined encoding for every of these steps. And so just using UTF-8 encoding wouldn't work.

CHAMPIKA WIJAYATUNGA Any other questions?

PAULO MILLIET ROQUE Yes. For the records, Paulo Roque from the Brazilian Software Association. Abyss has about 2,000 companies, $20 billion revenues, and 260,000 employees direct jobs. And I'm very happy

to see this presentation. I would like to be able to use it in Brazil. We can translate it and do the same kind of training in Brazil, because I think our companies, they need this a lot. I have a question that is, where can we have a list of all available tools to protect the DNS ecosystem?

CHAMPIKA WIJAYATUNGA Okay, good question. All available tools.

PAULO MILLIET ROQUE Or at least some of them, some important.

CHAMPIKA WIJAYATUNGA Okay. I mean, I would actually, I mean, as a single point of visiting, I would also tell you to go to that KINDNS website that I was talking about, KINDNS.org, because, KINDNS talks about all the practices that you can use to protect your DNS ecosystem. But then, of course, it wouldn't specifically talk about any of the tools as such. But then overall, best practices, you could follow KINDNS.

Yeah, but then when it comes to tools, there are plenty of different tools that you can use for different purposes, actually.

PAULO MILLIET ROQUE Okay, thank you.

CHAMPIKA WIJAYATUNGA Yeah, we have a comment.

ADIEL AKPLOGAN Yeah, just interesting. From Brazil, specifically, we have somebody in the region that probably has the translation already in Portuguese for this. Nicolas Antonio works in the same team, so he can provide you with that translation. We have also started some work with NIC Beer and CGI Beer to translate KINDNS and also make KINDNS available for the Brazilian community. So we can further discuss that, and I will put you in touch with Nicolas to help. Thank you.

PAULO MILLIET ROQUE Thank you very much.

ULRICH WISSER Yeah.

THOMAS I have a number of questions for the record, Thomas, from ENS. I was just curious, in the example given about the Punycode representation with the German Umlaut, is that to say there's a canonical Punycode representation? So, whilst there could be two representations in Unicode, there's only one Punycode representation?

CHAMPIKA WIJAYATUNGA: It should be only one because we should do something called normalization. So, even if you have multiples of code points, it should normalize it to a single.

THOMAS: Okay, perfect. And then my second question pertains to DNSSEC. Quite simple. Why is DNSSEC not enforced?

CHAMPIKA WIJAYATUNGA: Now, for gTLDs, actually, it is in the contract because gTLDs are ICANN contracted parties. So, it's in the contract. So, all the gTLDs, they have deployed DNSSEC. They have signed. 100%, they have deployed DNSSEC. But for ccTLDs, they are not contracted parties as such. So, it's more a voluntary thing for ccTLDs. But if you speak from Top-Level Domains perspective, currently about more than 90% or so, really about 93, 92, 93% of the Top-Level Domains, they have deployed DNSSEC. They have signed their zones.

So it's only about, certain amount of maybe about 50, 60 ccTLDs who are yet to sign their zones, not the gTLDs. So, the Top-Level Domains, they have over 90% or so. It's more of the second levels. That's where we have to do more work in terms of DNSSEC. Currently, it's still around 10 % or so if you consider the second levels.

THOMAS Okay. So that's likely just a cost and effort constraint on an individual basis whereby it's not contractually enforced so they just can't be bothered.

CHAMPIKA WIJAYATUNGA I mean, I wouldn't really say it's only the cost as such because it's more about awareness. It's more about confidence that, they should get to understand about DNSSEC and then do that. That's also why we do a lot of capacity building, a lot of creating a lot of awareness about DNSSEC as well so that, I mean, frankly, over the years, a lot of these ccTLDs and so on, they have come to our capacity building sessions and so on and they have gone and deployed DNSSEC in the TLD level, in the ccTLD level and so on.

So, same goes to the second levels as well, the registrars and so on. So, it's more about awareness and they're getting, more confidence about doing the thing. Cost-wise, I wouldn't say it's a big cost as such.

ULRICH WISSER Okay, perfect.

THOMAS Thank you. Just quickly as well, I was just curious how is the Public Key exchange done securely?

CHAMPIKA WIJAYATUNGA Okay. Now, when it comes to Keys, of course, root zone was signed back in 2010, a long time ago. And then we have also done a Key rollover later in 2018. And then next year, there are going to be another Key rollover. So, from root's perspective, it's all done for the last 15 years or so. And TLDs, a lot of TLDs, ccTLDs and so on, they do very good Key management.

If you want to actually see even how the Keys are managed at root level, you can go to IANA website. It's all been recorded. You can see all the root Keys signing ceremonies and Key materials and everything is out there, the process and all that at the root level. And then the TLDs, they do, I mean, not to that extent, but then to the TLDs extent, they do this Key management stuff.

And for the second levels, we do, again, we talk about best practices in Key management as well during even our training courses and so on. So, a lot of people who do manage Keys, they could do it in different ways. I mean, you can do it in a very cheaper way as well. You know, you could do it by your own. For example, a lot of these name servers, they have automated functions, utilities that you can do Key management. Or there are, if you want to go for some more automated, more hardware type solutions, also there are some solutions as well in terms of Key management.

THOMAS Thank you.

CHAMPIKA WIJAYATUNGA Yes.

HAYDER HAMZOZ Thank you so much for your efforts. I'm Hayder Hamzoz. I'm from Iraq. I'm representing NSIM here. It's a digital rights organization. And my question related definitely to digital rights perspective, which is do you think the DNS traffic analysis could be used to analyze and target vulnerable groups or human rights defenders online? Is there any technical mitigation exist here for preventing that action?

CHAMPIKA WIJAYATUNGA Oh, that's, I mean, when it comes to, of course, you're talking, I think, more from a privacy or confidentiality perspective. Someone would capture your DNS queries and then analyse it and use it for various purposes. Okay. Now, if you want, if you go into more sort of, say, in that level, I mean, even now, for example, say your network operator, the ISPs, mobile operators, they do that for various good purposes. Say, for example, if you're going to go to some malware sites or, say some parental controls and things like that, the service providers, they do that. They do filtering from their side.

I mean, they do that based on the metadata of the DNS traffic. Now, bad side is also actually, yeah, as you said, sometimes, you might be worried whether, you'll be tracked and things like that. If you

want DNS privacy, if you want confidentiality and so on, that's where the DNS encryption would come into the picture.

You have to actually have DNS encryption. But then DNS encryption will also have its own challenges. Say, for example, if you use something like DOH, then you'll be, your DNS traffic would be all encrypted and your service provider may not be able to actually take any actions based on those queries. And also, you could be talking to a DoH server, maybe in a different jurisdiction, and then there could be, different laws applicable and things like that. And also, if, say, your users, sometimes, if the DoH server is having some issue, they might think it's a problem with the operator and they could be complaining to the operator. Whereas the problem is really in the other end, in the application or whoever the application provider is in.

So, there could be a lot of other complications could also come in with that, with the encryption. So, I would say it's more of a balanced approach that probably there has to be some policies, even from community's perspective, even from regulatory perspective, you may want to consider some policies and probably use policies that can be good for the community and then probably, take it from there.

HAYDER HAMZOZ I mean, this will open a lot of questions. If we're talking about policies, then, is it a Human Rights by design, as we see here? You know, because I'm talking from conflict zones, not like stable

countries, not democratic government we're talking about. That's why there is a lot of DNS Abuse, and it's easy to see that Minister of Communication in Iraq, she wants to block the Google DNS right away, you know. So, that's why I think it's very important that to see more voices like us, especially from our region, talk more about human rights aspect by design, especially I think business-wise, ICANN should follow the, business and Human Rights, UN guidance, I think, for that.

CHAMPIKA WIJAYATUNGA Do you have any comments there, Rich?

ADIEL AKPLOGAN Well, ICANN has no policies for your ISPs, so we're clear on that. We don't do that. We don't do it on that level. So, I agree on you that, that DNS privacy is important, and I would agree with you that I think an open access to the Internet should be a Human Right. I think most people here at ICANN would at least agree with you, but we are not the place to argue about it. That's a UN question, and that's not us. So, and just because you asked about the security and surveillance, even protecting your DNS traffic doesn't make you secure. So, you have to, there's a lot of more steps you have to make your communications private and not trackable before you, it's not only your DNS.

THOMAS Yeah.

ADIEL AKPLOGAN

Yeah. Rich just mentioned it very, very, very well. ICANN does not define policy for that. But what we can do and what we do is what is happening here, raising awareness about the standard, about what is being done to reinforce those levels of security, those level of encryption. And being here helps operators to implement them. But because at the end of the day, it's for the operators, for those who run, actually run the network, to be able to implement those standards that protect users, that add encryption, that add privacy.

ICANN does not have the mandate to set those kinds of policies. And ICANN doesn't develop standards either. It's the IETF that does. And the IETF is doing a lot of work to reinforce privacy in the protocol usage in general. Privacy by design, security by design. So, what we do at ICANN is to raise awareness where it's needed, agreeing on the fact that we need to be safe online. But operators have an important role to play there. So, they have to watch what is going on and implement what's needed to be done within, of course, the local regulation framework, which we cannot go around, unfortunately.

ULRICH WISSER

Adiel, just for the record, could you tell on the mic who you are?

ADIEL AKPLOGAN

So, Adiel, I'm from ICANN, in charge of Technical Engagement.

CHAMPIKA WIJAYATUNGA Any other questions?

LUTZ DONNERHACKE It's Lutz Donnerhacke from At-Large. I'd like to add another answer for you. Especially to the ISP level, there's an ISP constituency at ICANN. There are some really tough guys there and they can give you an advice how to deal with the not so easy situation you are in. That's the advice I would give you because the people are here. And this is a short walk to get some answers outside of the scope of ICANN, thanks.

CHAMPIKA WIJAYATUNGA Okay, are there any questions from the remote?

ULRICH WISSER No, no questions.

CHAMPIKA WIJAYATUNGA All right, then thank you very much for joining us in this session and we will see you around. Thank you.

ULRICH WISSER Please stop the recording.

**[END OF TRANSCRIPTION]**