HYDERABAD – How It Works DNS Fundamentals
Thursday, November 03, 2016 – 09:00 to 10:30 IST
ICANN57 | Hyderabad, India

UNIDENTIFIED MALE: Take a pull of the room. Is it bright enough in here you guys? Do you want more light? Is it calming and cool? You're okay over there? How about over there where it's darker, you guys okay or you want more light? We're okay? We're okay with light. We're good.

STEVE CONTE: All right, so we're going to get started. We're running in some last-minute technical challenges, which is the way of an ICANN meeting. So in the meantime, I will give my introduction and all that.

So welcome, this is a really basic DNS 101 Fundamentals. Do any of you here know DNS at all? All right, Wes, you're in the wrong room. Seriously, this is entry level DNS. We'll go through some history. We'll talk about resolution process. We'll talk about some of the players, the registrar, registrant and stuff like that but this is really meant to acquaint yourselves with what the DNS protocol is and what the DNS system is.

So I don't want to waste your morning if you've got breakfast or coffee or something that you'd rather do than sit here, then please – I want to just give you a heads up on that.

I am Steve Conte. I'm the Director of Programs for ICANN's Office of the CTO and I've kind of got a bit of a past back. This is my second time at ICANN. I started at ICANN first time back in 2003 working in the IT Department and working with John Crain on the l.root-server. And then, I left ICANN in 2008 and then went to Internet Society for five years and ran their Fellowship to the IETF. And then, I came back here about two and a half years ago.

So this is my second time around. It's kind of like – I don't know if you guys are familiar with the Eagles song, the Hotel California. Once you get into the ICANN community, you can try to leave but you really can't. You're here for good. So welcome for those who are new and we'll be seeing you over and over, and over and over, and over again.

I'm going to switch to manual mode here. Rules of the house is I'm hard of hearing, therefore, I have another microphone. If you have a question, please raise your hand, I will bring the microphone to you. We all also are streaming and recording the session, so I'd like to capture all questions on the recording. Please interrupt me at any time. I'm not a formal speaker. I like

the casualness. I like the interaction. So if there's a question, ask it right then and there, and we'll get you an answer.

With that, can I get you?

UNIDENTIFIED FEMALE: Why don't you just… You want to [inaudible].

STEVE CONTE: So DNS came because addresses are hard, numbers are easy for machines to know and remember and they're really hard for people to know and remember. And it was somewhat hard back in IPv4 days when you only had the dotted quads. You only had four numbers to remember. But when we moved on to IPv6 and we get this whole bizarre string of hexadecimal and decimal characters, it becomes very impossible for a human to remember an address for a machine. So we needed to use names.

So back in the early days of the Internet, names were simple. There were no domain names yet and so there was a HOSTS.TXT file, the hosts file and there were single labels, so again, there was really the very beginning of the Internet and people would submit their host name to a central place. Go ahead the next slide.

It's centrally maintained the NIC, Network Information Center at Stanford Research Institute. And it was good, it worked. But it was also clunky because it was essentially maintained database. The host, the other machines on the network had to go and download it regularly and it became quickly out of sync, so not everyone grabbed the latest file and so there wasn't always an update to it.

As we see, it was released once a week and it was downloadable via FTP but people had their stuff on their day job and stuff, so remembering to grab a single file once a week, may or may not have hit their to-do. Go ahead.

There was also naming contention. So because it was a central thing, people would submit their host names, again, a single string so there wasn't a .com and a .net and ways to differentiate that. So there was contention on who would get that host name. And, it's made by hand so there was the potential for typos and errors to be entered into the host file that way.

Like I said, there was never really a good method to synchronize this file, it relied on a human to come and download the file and install it on their machine. And, traffic load was pretty significant for that day. Don't forget this is the day of low bandwidth, dial-up and other methods that we don't have like today where we

have broadband and things like that. So, it was a pretty significant load to download this host file.

So it was decided among the community that essentially managed database or essentially managed file just wasn't a good way to move forward.

So back in the early '80s, they started discussing a replacement and the goals was to move the host file to another way to address it. So there were scaling issues that we just spoke about. They wanted to simplify e-mail routing, which we'll go into a little bit and the result of the decision by the community at that point, the technical community was to come up with the Domain Name System. And a number of RFCs request for comments. RFCs are just a side note, anyone not familiar with what an RFC is? Everyone is a pro, all right.

So RFC is maintained currently by the IETF. Early '80s, the IETF, mid-'80s was just starting up, so the RFCs actually proceeded IETF. But since then, the IETF took over management of the RFC series through the RFC Editor and a number of RFCs, the number of documents had been published about the DNS. Go ahead.

So DNS is a distributed database and that means that it's managed by a lot of different places and it's not centralized like before, so there are some root areas – no pun intended or pun intended – but then as we look at delegation and things like

that, we'll notice that the management of various pieces of the DNS are distributed throughout the globe.

Some definitions, when we talk about "resolvers," that's people who are sending queries. So resolvers will be [inaudible] from your telephone or your laptop that's open right now. It will be the server in which – we talk about a caching server, when you're asking a question, you'll be going to the next step in your machine in the system. That is going to both be a resolver and a name server and we'll look at that as well.

We talked about or we'll be talking about caching to improve performance and replication and redundancy, and the things that make DNS robust and strong that it is. Go ahead.

So, if we look at the components like I just said here, resolver could be your machine, your phone, whatever, when your program is Safari or anything or e-mail server, any of that server – the programs that you load up on your phone or on your device, it's going to make a call to a resolver that's on your device. And that resolver will then move to send the query out to the name server in whatever network you're in, be it your home network or here at the conference we have, DNS servers too that are providing resolution.

That name server or the recursive name server also acts as a resolver and that it asks on your behalf to various [inaudible] of

name servers and we'll go through the whole process down the line but this is kind of at a glance what the DNS components are.

DNS namespace structure, the whole inverted thing is called the namespace and each node has a label. So, if we look at the very top, we see the root and the dot, then we'll look at that dot in a little bit but that's the very top of the hierarchy of DNS. Beneath that, we have the top-level nodes, top-level domains and that could be anything from .com, .net, it could be a country code TLD, like here we are in India, it will be .in. It could be an IDN TLD, which we see off to the left here and this is .hk in its IDN form.

Then the next level below that is the second level. This is typically where you'll see the registered domains that happened, company names – nike.com or icann.org or any of that. That's typically where the registrant will come in and register a name.

And then below that, we have the third level of the nodes and those are typically the host names and machines themselves. So when you look at – go to the next page. Now, okay, you can stay there. When we look at a couple slides from there, you'll see the tree and as we walk through this, the tree and the structure how that all comes together.

Legal characters for the DNS that we called LDH, Letters, Digits and Hyphens. There's only a certain – and it's asking characters and zero through nine and then the hyphen, and those are the only characters that are allowed in the DNS itself. And so, when we look at IDN and we see this xn--jade6w193g, there's a conversion that takes place within the applications that converts the IDN, the Internationalized Domain Name to an ASCII subset that is recognized throughout the whole DNS system.

Maximum length is 63 characters and there's no case sensitivity within the DNS. So if people are putting their e-mails out with lower case and upper case or domains with upper case and lower case, the system doesn't care. You can type it all in lower case, all in upper case, a combination if you want to be that crazy person that does upper and lower case, the system will read it all as lower case. Go ahead.

All right, so here we look at the different nodes in the sequence of this. So we can build a domain name that everyone is familiar with by going – taken from the bottom of the structure, the bottom of the tree and go up, so we have host name www, we have domain name, example we have topleveldomain.com and then we have the root. There's always a dot at the end of whatever domain you're talking about. As a human, we don't always need to use that dot. Go ahead.

And speaking of the dot, putting the dot at the end is called a Fully Qualified Domain Name, an FQDN. That means that there's no ambiguity in that domain. So if you want to make sure you're going to the right domain DNSSEC removed from this point. But if you want to make sure that you're not… you're [accidentally] going to some place else, you put that dot at the end and all servers will look at and say, "This is fully qualified. This is exactly what I'm looking for." There's no ambiguity. Go ahead.

So domain is a node and everything below it and as the sentences part of that domain. So here we look at .com and everything within that space is the domain of .com. That means that .com is managing at least the pointers to everything beneath it recording the pointers to it. So if we look at this, the apex, the top of that domain space is com. Go ahead.

And I'm reading these slides to you. I've taken these slides from my colleague Matt Larson, so these are not my own slides, so my pause and my flow is because they're not mine but Matt knows this stuff, so these are good slides.

Administrative divisions are called zones. So if your .com or if I have conte.net as my own domain, when I manage the space within conte.net, that's called a zone. And when I'm creating and working within that zone, I create something called the zone file and that's where I manage the space for conte.net. The person

who's delegating that zone is the parent. So for my purpose as conte.net, .net delegates that zone to me and then I'm the child of .net. Go ahead.

So another look at the domain space on this. Go to the next slide. I think there might be a highlight. Go next.

So here we look at administrative boundaries of the namespace. So, root has its own zone, all the TLDs have their own zone and then underneath that, when we get down to the host level, we see that the zones are a little bit larger because they're including host files – not larger because .com is huge. They're a little bit more diverse in what they're putting in because they're including mail records or including host files and other types of information resource records. Go ahead.

So, name servers as we spoke before that answers queries, resolvers ask for queries. A name server is authoritative if it has complete knowledge of that zone and we'll go through this, the whole process in a couple of minutes here. They're going to provide a definitive answer to the query about that zone. And zones should have multiple authoritative servers, which provides redundancy. You want to have them in topographical diverse networks.

You want to put them in different spaces if possible so they're not all within one IP address and then the same network of each

other because if there isn't intact or something goes wrong and your network is unreachable, you want to make sure that this diversity is out there and there'll be other servers hosting your zone that can answer these queries.

And it spreads the query load out. If you have more than one server, it will make the queries a little bit less. We're going to turn on this mic here. Testing, testing.

UNIDENTIFIED FEMALE: So when you say in different locations, it doesn't have to be geographically spread around the world, it just has to be… can they be near one another?

STEVE CONTE: That's a great question. They can be near one another. There's no off. It's kind of a little bit of both. So in the network world, diversity doesn't necessarily mean geographic diversity. We can live next door to each other and I might have AT&T U-verse, you might have Time Warner. We're on different networks. So if I have a server in each one, we have a network diversity going on.

Geographic diversity helps when you have a website or a domain that's being hit by a bunch of different locations around the world, then it becomes a… We're here in India, it might be closer

to go – well, to India or to the UK versus going back to my DNS servers in California.

So it's kind of a balance between. You could certainly have a diversity within your local region with different providers and that would become almost completely sufficient. If you're a huge bank, you want to get a little bit more doors than that. Does that answer your question?

Go ahead. Next slide. Oh, the nice small slide. Always good small slides.

Synchronization – I'm going to just recap this. This is a complicated slide. So everything, every server, every domain has what's called a master server and then hopefully has a secondary or more secondary servers. The master server is typically as you as a network operator, as a DNS operator are going to push your changes out to your DNS system, to the DNS system.

And when it's talking to its collective, if you have more than one DNS server for your domain, if it's talking to the collective servers, there has to be one who is a master and the other ones are secondary. The secondaries will pull the query updates from the master to synchronize that automatically. Am I catching everything on that?

So they will do a zone transfer pretty regularly. It's up to the operator to determine how often that happens. It's up to the operator to determine whether they're going to ask, whether they can pull for the information or whether the master is going to push and say, "Hey, I've got an update. Come and get it." They'll never push the whole zone file out to the secondaries. They'll say, "I have the update. Come ask me for it." And it's up to the secondary server's job to go and get that. Go ahead.

All right, DNS standard specifies the format of DNS packet center of the Internet. They call it the master file. And again, we're talking about master servers, master DNS servers for your authoritative zone and the zone file detains all the data for that zone in the master file format. Go ahead in the next slide.

So what that means really is that inside the zone record, zone file, there's resource records on that. These resource records are different ways to identify objects within that domain space and you'll see them abbreviated as RR. There's a whole bunch of different types. We're going to talk about a couple of them.

Every zone has at least one zone file even if the zone is empty, there's going to be at least a zone file in there relating to that zone if it's registered telling you who's the source of authority and very basic information even if there's no hosts on that at all. Go to the next slide.

So some of the format of the resource records, we're going to show the owner, the domain name and the resource records associated with. So if I have conte.net, I would be the owner and there would be an association in that and then we'll look at SOA record on that.

There's a Time To Live records and the Internet changes constantly, so we want to make sure that there's something called a TTL, a Time To Live, which designates how long the information in this record is valid for it. At some point, it's going to time out and it's going to force DNS servers to go and ask the question again and this make sure that we don't stagnate our cache and then make sure that any kind of information you're getting or you're asking for is up-to-date as much as the last change that the zone owner made.

We have a class. This is mostly unused back when DNS was first formed. There was ways to look at using it in different methods. The Internet took over and became way more popular than I think was originally. Well, no, it became more popular than the other class types, therefore, the Internet class became the primary class for DNS records.

We have type, which is going to designate between whether it's a host, whether it's an alias, a CNAME, whether it's a mail record or an name server and there's some other types of data in that,

too. And then, there's the actual data that's put in there. So the resolution between a name and a number, the number would be the data on that. Go to the next slide.

So here's another way to show it. In a zone file, it will always show the owner, the Time To Live, if it's not inherited from the master, the SOA records, Source of Authority record, the class and then the type and the RDATA. And you notice that there's some brackets on that. The brackets are optional and a lot of that if you leave out the brackets, it will be inherited by the Source of Authority entry, the SOA entry. And so you can leave the brackets out and it will inherit all of that information and you just put in the type, the host name or the MX record or things like that and then the data itself. Go to the next slide.

Some common resource record types you'll see. If you operate a DNS, you'll see the A record, which is an IPv4 address. You'll see quad A, which is a v6 address. You have to designate the different types of IP within the record, so it knows which one to answer when a query comes in. If you're an IPv4 only network and if you're asking for an address and it gives you a v6 address, it's not going to do your machine any good because it couldn't get there. So by breaking apart and designating between a v4 address, an A record versus a v6 address, a quad A, it allows some more diversity on how to answer that.

We have NS records, Name Server records. These are authoritative records to show what servers are answering, what authority to this zone. We have the Start of Authority or Source of Authority record. This always appears at the very beginning of a zone file. We'll look into that a little bit more detail but that shows some basic default Time To Live. It will show an e-mail address for the manager of that zone and some other things.

We had aliases, CNAMEs, this is if you have one machine and we've already set up an A record for that machine and that machine does multiple things and we want to give it a couple different names, so if we have the same machine does e-mail and it does web mail, we might have www as an A record with the IP address. But we don't want to put it in again with an MX record or a mail.whatever, so we'll do a CNAME and show that it's also acting as that host as well.

We have MX, Mail Exchange servers. We'll go into the detail on this but you'll see that this is different. You still need to specify a name if the mail server that you're running is within your domain space. So you'll still set up an A record or a CNAME but then the MX record designates two SMTP servers, which servers are acting as mail servers. And so we'll go into a little bit more detail on that later on.

And then PTR records, these are associated with reverse mapping from – we're talking about primarily here as mapping human words to IP addresses, www.example.com to an IP address. PTR and reverse mapping is if you know an IP address and you want to see what the reverse, what the host name is or what the domain name is, you can set that up until you can do a lookup on IP address. Or if the PTR record is there, it will give you back a domain name for that record.

UNIDENTIFIED FEMALE: It's probably basic but can you have more than one CNAME?

STEVE CONTE: Absolutely. CNAMEs are – you can have as many as you want, the challenge comes in and I think some of the slides somewhere else's that you don't want to – you want to avoid doing a CNAME to a CNAME to a CNAME and avoid looping CNAMES in. So the more you can stay to one CNAME aliasing to in A record to an actual name, it's more efficient.

If I have steve.example.com and I alias shawn.example.com to steve.example.com and I alias [along].example.com to shawn.example.com, which goes back to Steve and it gets confusing and it can cause issues within the system. All right, go ahead.

Are there a lot of other types of resource records? As of August of this year, there were 84 types allocated and IANA is the repository where those are recorded. If you ever wanted to see all the resource types either go to IANA, go to this page, Google resource type records for DNS and you'll get all 84 of them.

This is a wonderful slide for big rooms. This is just an example of the IANA page that has all the different resource record types.

All right, so as I mentioned, most common types of use of the DNS is mapping a name to a number. That's what we're going to mostly focus on in here. So if we look at example.com where mapping an A record and an anchor record from example.com to an IP address. If we wanted to map that to an IPv6 address, then we would set it up as a quad A, as an AAAA record. Go ahead.

The NS record specifies the authoritative name server for a zone. This is the only type of record that will appear both in the parent delegation and the child, and it's important because – well, we'll go to the next slide. You always have to have NS records in your zone file.

And so, here's an example where delegating… We have NS records both in the parent in the root zone for .com and .com also has those NS records listed as well. And that's because from a .com perspective, it's saying these are my authoritative name

servers, these are the ones that I recognize that will give you good information about my zone.

It's also at the parent because the parent needs to know how to get to that next to the child record. So all the delegations of zone data are in the parent as well with the name server record, so it knows how to answer a query to say, "Go talk to the com servers and here's the addresses for the com servers." Go ahead.

And, this is exactly what I just said, so go ahead next slide.

So there's another thing called glue. So let's say you wanted to go to conte.net and so you went to .net and you said, "Hey, what's the address to conte.net." And it said, "Well, go ask ns1.conte.net." And you're like, "Okay, oh, but how do I get to conte.net?" And .net says, "Go ask ns1.conte.net." There's no IP address associated with it. So you're getting yourself into a loop because the parent doesn't know how to tell the query, the querying server how to get to that next step.

So it was solved by putting something called glue. So, at the parent level or at any level really, when you put an NS record, you'll also going to put in a record that defines that address or defines that record with an address. So for what's on the page here, for an example.com, we're also inserting the IP addresses to example.com and we're putting that up one level up. Well, that's actually should be down at .com. So when I go to .com

and I say, “Hey, I want to get to www.example.com,” .com says, “well, here’s the name server label. Oh, and here’s the IP address too because you’re going to need that.” So it gives that back to the querying server, so it knows how to take that next step.

Glue is going to be an A or a quad A record and it’s like – it says it’s included in the parent zone and the parent part of the delegation. And, it breaks that circular dependency, so to say it again, it gives you the information how to take that next step and this is super important if all your name servers are within the namespace that you’ve registered. So if you have your example.com in all your name servers in an example.com, you must put glue above that, otherwise, no one will know how to get there.

Only one SOA per zone, this is the top of the zone file. What this usually shows is – go to the next slide, maybe that breaks it out. No. Go back. Sorry. Back to now.

This will show Start of Authority. When you have a zone file and you’re managing a zone, this will show you the querying person, the querying machine that your Start of Authority that anything that you’re answering, you know everything that’s about the zone, you as the DNS server, the hosting server. I don’t know. It’s kind of hard to see.

What we're showing here is the zone itself. We have the primary name server and as one, that example.com, which would be a master in this case. Hostmaster.example.com is actually an e-mail.

What you normally do is you swap out the first dot and you'd replace that with an @ sign. @ signs mean different things to different machines, so in order to not break various parsing issues, they took the @ sign out on that one.

It also shows a serial number. Serial numbers are a way to designate when comparing zone files especially between parent and secondary, master and secondary, which one is more up-to-date. So if I'm a secondary server and I'm going to query, the master server, I'm going to query and compare the serial number and say, "Nope, mine is older. I need the newest one. Or if I have the same one, then I don't need to go and get the new data because there is no new data theoretically."

There's a refresh on this. There's a retry. The refresh is your TTL. Was that the refresh [inaudible]? Wes, refresh, TTL, was that the same? No. Wes is standing up.

WES HARDAKER: Now, the TTL is how often the clients should be caching your data. And typically, the things in the SOA are more designed for

the synchronizations, so the refresh is how often that the slave should synchronize [inaudible].

STEVE CONTE: Thank you for that. Wes Hardaker is a very heavily involved with DNS. He's in the wrong room and I'm glad he's here. So, thanks. Next.

Yeah, I'm sorry. Go back one.

So he was saying that the data in here is mostly about the communication between secondaries and the master. So in this case, the refresh specifies to the secondary, if don't get a message from the master, how often should I go back and ask them? Go ahead.

CNAME – we kind of talked about the CNAME with Shawn's question about how many you can have. You can have as many as you want but you want to avoid making long chains and loops of CNAMES. Again, they're just aliases. They're canonical names but in this case, we have an A record for some hosts example.com and we're making an alias to say mail.example.com is also that the IP address of this some host of example.com that's already defined within the zone record. Go ahead.

Mail routing – so in the old days, it was pretty easy. The address was a single string, so if you want to just send an e-mail, that single string, that host was pretty much running all the servers, services for that network. Once we got into a more complex Internet though, we wanted to make sure we detach the concept of e-mail from the host or from DNS. So even though we have, we might have put an A record in that says mail.example.com is 192.0.2.7, we need a way to tell SMTP, the Simple Mail Transfer Protocol in how to go and how to reach out to this server. So there's an MX record. Go ahead to the next slide.

MX record tells mail servers queries how and where to reach different mail servers. So you don't actually have to have all your mail servers or any of your mail servers within your domain space. You can have them hosted by – and I know Microsoft does Outlook.com. There's other services out there that will host your mail servers and you can then tell through your zone file, you can tell mail servers who to talk to in order to get your mail or their mail to you to your server.

In here, we show – so mail obviously, the owner name, the username is on the left side of the @ sign, the domain name is on the right side of the @ sign. So when it's going in, if I'm doing steve@example.com and they want to get an e-mail to me, the mail server will send the query out and will come back and be

asking for the MX records. That response will come back with the MX records.

And it also has a number there and that's wait, so a waiting of the servers. If you have more than one mail server, it's going to show you a preference on which one to use first and if it can get to it, it will go ahead and send that mail. Otherwise, it will go down the list by the lowest number as the most priority and then it works up from there and we'll try these different mail servers until finally it gets one that it will speak to and then it will route you an e-mail. Go ahead.

So we're not going to go into a whole lot about reverse mapping. Mostly to say that it exists that you can look up in the IP number and if there's an entry in the reverse mapping through PTR records, it will come back and give you a host name if it was set up right.

IPv4 addresses uses something called in-addr.arpa. IPv6 addresses uses something called ip6.arpa. It's the things that as an end user you'll never ever see. Most .arpa services are – they're core services that the Internet needs to use [inaudible]. So for reverse mapping, the in-addr space is in arpa and ip6 space is also in arpa. Here's the things that most normal user will never ever see but there's where the reverse stuff happens.

And much like an A record where you put in the name, example.com and you have an A record and you have an IP address, here you have kind of a reverse of that. You still see it's kind of a name but if we look at it backwards, it's 192.0.2.7.in-addr.arpa. This is kind of how using in-addr.arpa using the DNS system, it's how it reflects that. So in the in-addr space, they put that in, instead of an A record, we put a PTR record in and we say, "That space there results to example.com."

And again, we're not going to touch reverse mapping. Most people won't ever use reverse but it does exist. People do use it and machines do use it, and it's a way to see the IP address and say, "I wonder what domain name that is," and you can reverse mapping off of that. Go ahead.

And DNSSEC, if you guys are interested in DNS Security, DNSSEC, there is a tutorial tomorrow and I highly recommend you guys go to. This is I think my only slide on DNSSEC. It is a method of authentication. I always say that in DNSSEC in some ways should have been called DNS auth because it's more of an authentication model than it is a security model. It's not going to encrypt your data. It's not going to do anything else other than authenticate. The fact that if you're asking a question, it's authenticating that the answer is coming from a predictable unknown source and it's got keys and things like that to manage that.

But there's a whole session on this tomorrow and then there's a tutorial on it later on this week. So if you're interested in DNSSEC, I highly recommend going to that. I'm not going to try to reinvent the wheel here.

But what we look at here, it's a very basic one slide description is that DNS data can be digitally signed for authentication. The signatures and the key pairs are split up. Some of the key pairs are put into the parent of the delegating zone. Some are put into the zone itself. Some are not shown at all. Those are held by the zone manager and the way that that's managed like that, it builds a chain of trust in which the DNS queries can be authenticated through the steps of the DNS process.

There's no certificate authorities. Parent zone vouches towards child's public key. So if I have conte.net and I signed it, I'm giving those keys through some kind of trusted method to .net, .net will publish their parts to the keys and say I'm vouching that the authentication took place and that chain is built between my .net and the child from that.

There's a couple of new record types that were introduced with DNSSEC: the DNS key, the RRSIG, which is for the resource record set, NSEC, NSEC3, which are pointers and the DS, the Delegation Signer. These are all new record types that we're going to introduce along with DNSSEC. And again, if you have

any interest in DNSSEC and you haven't done it before, go to the Beginner's Guide tomorrow and then there's a tutorial, a hands-on tutorial later on this week that will show you much more in-depth on about DNSSEC and how to use it. Go ahead.

So more record types – txt, you can put text into the record. URI, TLSA, many of these aren't being used anymore or aren't being used as much. Like we said earlier, majority of what you'll see in a zone file are A, quad A, MX records and NS, Name Server records but there are 84 types total that you can put in there. Go ahead.

Oh, another great slide for a big room. This is a sample zone file and what this really is showing and we'll not spend too much time on this because you can't see as a top part is your SOA, your Start of Authority. And then, you have your Name Server records and then we have some glue at the very bottom, and then we have some resource record types. So we have an A record there, we have a quad A record, we have two MX records showing that there are two servers out there that will handle e-mail within the example.com domain. And then, the next slide, we have some glue.

That glue also goes as mentioned, it goes also into the parent delegation as well, so.com would have the NS records, the

example.com NS, NS1, NS2 and it will also have the glue listed in that as well. Go to the next.

So the resolution process, we have stub resolvers, we have recursive name servers, we have authoritative name servers and they're all going play a part in the resolution.

DNS query always has three parameters. It's asking for the domain name, the class and the type. So, example.com would be the domain name. IN, Internet would be the class and then the type is an A record in this case when you're doing a lookup.

Two types of queries, there's recursive queries and there's nonrecursive queries. Stub resolvers send recursive queries. I need a complete answer or an error. Give me an answer or say that I have an error or MX domain or something like that.

And nonrecursive queries says, "I could do some of the lookup work myself and looks up a referral." So we'll talk about the referral in like a side or two as we move forward on this. Go ahead.

So, high-level algorithm for processing query. It's going to different Domain Name Servers, will do things different ways. So there's no exact answer possible. It will go on and ask the namespace, the DNS server will answer that in some automagical form. Go ahead.

I'm sorry. I should have cut these slides out. Go ahead.

So, root zone administration. We'll do a sample recursion or a sample process for creating an answer in just a second. So there's two organizations now that cooperate to administer the root zone's contents. So if we look back at the hierarchy, we have root, we have top-level domains, we have domains and then we have hosts and the space underneath those domains.

As it stands now, we have ICANN. As the IANA functions operator, we have VeriSign as the root zone maintainer. Those are the two entities now that act on administrating the root zone, only the root zone data. And then, there's 12 organizations that operate root zones and these root zones are the top of the hierarchical tree when you're doing requests and queries and stuff like that. Go to the next slide.

UNIDENTIFIED FEMALE: I have a question.

STEVE CONTE: Oh, yeah.

UNIDENTIFIED FEMALE: Remotely, what is the class and type that was shown in the last slide?

STEVE CONTE: Can you go back to probably two slides? One more.

All right, so here on the second bullet, we're asking again that a query always has a domain name query, a class and a type. In this example, we're asking for the domain name of www.example.com. The class is IN, Internet, and the type was A, which is an anchor record, A record is an IP address. So what this query is actually asking is what's the IP address for www.example.com?

Again, IANA is – the class type is typically not as important as it used to be. So IANA is pretty much the de facto for most queries. What will change in most queries is the domain name, the www.example could be ftp.example. It could be example.com without a hostname and it could have a type of MX, and that could be an e-mail server asking for the Mail Exchange information for that domain.

I hope that answers the question on that. I think we are to forward three.

Yes, and one more, sorry.

So currently, the 12 operators, we have 13 root server names in the Internet. I'll say that carefully and we'll go into that a little

bit detail. We have 12 operators of the root servers. Even though there's 13, VeriSign runs two of them. They run A and J server.

A common misconception is that A root server is the master, that's not true or correct, A is just another e-mail or just another root server, just as equal as G or K or E. They all act and they all respond in an equal fashion. Go into the next slide.

They have a website, root-servers.org. It's a great website. Actually, what this is showing here, it shows the location of various root server instances. Now, if we go back one slide, sorry. I said there's 13 root server letters on the Internet. Back 15, 20 years ago, I could have said there was 13 root servers on the Internet but that's not true anymore.

I believe the root server operators are going to be talking about that this afternoon in the RSSAC presentation here this afternoon. But using various technologies, one of which is called Anycast, they are able to replicate root server instances. We have a question back there. We're able to replicate root server instances. So now we have well over 130 root server instances. So even though there's only 13 letters, there's much more than that as far as the surface go. Question.

UNIDENTIFIED FEMALE: Yes, so what's the difference between A-VeriSign and J-VeriSign?

STEVE CONTE: There's absolutely no difference between that. They're different machines. They're different instances. They're just operated by the same operator. There's history in the past, please come back for the RSSAC stuff and ask VeriSign why they have two but it was mostly a question of delegation in the beginning of the DNS process.

UNIDENTIFIED MALE: Is the other one becoming redundant?

STEVE CONTE: They're all redundant. The whole point of having all these instances is that you have a redundancy across the Internet. So the fact that there's 130, 140 different instances, all operating as a authoritative name server is a redundancy in itself. The fact that VeriSign runs 12 of them, it's really nothing more than they have two of them. There's no redundancy in their model at all. It was just a question of delegation in the beginning.

Like I said, RSSAC, the root server operators will be here this afternoon. They'll be talking about Anycast, they'll be talking about the root server system. Please come back. Ask them that question directly and let's see what they have to say.

You have another question?

UNIDENTIFIED MALE: What is AS112 name server?

STEVE CONTE: Sorry. Say that again.

UNIDENTIFIED MALE: AS112 name server.

STEVE CONTE: AS112 is – is it the black hole? Alain? Alain? Alain? Alain? AS112. Alain Durand from ICANN.

ALAIN DURAND: Hi, good morning. Sometimes when we get queries for – reverse DNS to match IP address to the names, sometimes what you have is a private IP address, RFC1918 addresses like 10.something or 192.168.something. And if you ask of a query, what name matches this IP address, one of the queries is kind of ambiguous because it says your local IP address.

So if you ask your local server, that's fine, we have correct name. But most of the time, the local server is not necessarily become [configured] with that and what happen is the query goes

outside. And there's a really high volume of queries that goes outside.

So AS112 is essentially a project to catch all of those things and instead of polluting the root with a bunch of queries that are meaningless and essentially we done immediately to you that that query makes no sense because you cannot ask the global system about the local name. So that's what AS112 is all about.

STEVE CONTE: [Inaudible] call it the black hole servers. As Alain just said, the private address space is not ever meant to [inaudible] see the light of day. So if you're in RFC1918 space, 192.168.whatever and it's giving out to the Internet, there needs to be ways to stop the queries and there needs to be ways to stop the proliferation of the data itself, and there's a couple different projects. PCP38, is that to stop BCP38? No?

There's BCP after this that is meant to help network operators understand to not let their private address space proliferate out to the network and to stop it at firewalls and gateways, and things like that. AS112 is the same thing. It's meant to stop queries about private address space to go out on the Internet. Go to the next slide. Next one after that.

All right, so, two entities right now, managing the administration of the root zone. A ccTLD manager or a TLD manager will submit a change through the root zone management system. That change goes to IANA functions operator and they verify the query, the change request. They verify that it's coming from the right person, the right entity. They verify that the information that they're asking to change is correct too but they don't want to put out data that's incorrect especially at the root level.

And then, once that's all verified, then they authorize the change to the root zone maintainer, which in this case is VeriSign. They go and implement the change into their records. They put it into the master and then all the root servers as secondaries will come and create and get the new root zone that has that change in it.

Seeing the diagram, you see the root zone database, the root zone file and then the root zone distribution. So it goes in, there's a database that is managed internally by VeriSign. That's where the changes are held, where the changes are made. Once the changes are ready to be published, then they push that out to a root zone file. That root zone file resides on the distribution server that sometimes they call the hidden master because the only entities that can touch that server are the root servers themselves.

And then the root servers are secondaries, they go and they query, they pull that information and they get the new updates from the distribution server. And then, the root servers are the publication method in which the general populous on the Internet can go and query the root data.

All right, so resolution process – this is a game really about who you know. Resolution is all about who you know and who you don't. So, imagine a party and I can go up to Alain and I'll say, "Alain, I'd like to Shawn White. Do you know Shawn White? Can you tell me where Shawn White is?"

ALAIN DURAND: "Hi Steve, yes. I know and you're asking me for a big favor here. You know that at some points, I may have to ask you another favor, right?"

STEVE CONTE: "Can you refer me at least?"

ALAIN DURAND: "So, here is his number. That's a business card and [at the] back of it there is a number."

STEVE CONTE: So it's really about understanding who you know and if you don't know – and this is Shawn White by the way here – if you don't know the server that you're getting to, you're going to ask the ones that you do know and eventually they're going to refer you to the right server.

So here we have in this case, my trusty iPhone 5 it looks like, I'm on Safari on my iPhone and I want to get some – go to the next page, please, next slide. I think I'm going to example.com. Old attention, yes, I'm going to example.com.

So I want to see what's going on in the latest [craze], example.com so I typed that in my browser on my phone and I hit Enter and through API, the browser talks to the DNS resolver, the stub resolver on my phone and it says, "Oh, I need the address to example.com but I don't know it, so I'm going to go and ask my networks resolver, name server." Go ahead.

And so, it goes and ask the question to the NS server on my network and says, "I want to know the address to www.example.com." For this purpose, we're going to make the assumption that the caches are empty, that there's been no queries made before and we'll go back to that in a second.

The name server, the recursive name server on my network says, "I don't know that www.example.com but I know that hidden

dot, I know the root servers, so let's go ask them." Go to the next slide.

So it picks a root server. There's a number of methods in which it will pick the root server. Some of them will be speed and latency. Some of them could be hardcoded. There's different methods for different roots or different server applications.

So in this example, we're asking our root server, which is managed by ICANN. We say, "Hey, what's the address for www.example.com?" Our root says, "Oh man, I don't know. But I know the answer to .com, so let me refer you to them and ask the question there." Go ahead to the next slide.

So here's the addresses for the .com servers. Go ahead. So now, my server in the network is now acting on my behalf as my proxy and it goes to the .com servers and says, "Hey, I want to know the answer. I want to know the IP address for www.example.com." And the .com server says, "Oh man, I don't know. But I know example.com. I know those name servers, so let me give you those name servers and go ask them." Go ahead. And go one more time.

So now, your name server goes down to NS.example.com and say, "Hey, I'm looking for www.example.com." The example.com name server says, "I know this. I know the answer to this. Here's the IP address for www because it's in my zone. It's in my

authoritative zone and my delegation space." So it gives the full address for www.example.com back to the name server, back to the recursive name server that's in your network. Go ahead.

That sends that information back to your stub resolver on your telephone or on your laptop or whatever device, your smart TV whatever or refrigerator. Go ahead.

It sends it back to the browser. And now, it will go and speak exactly directly to that IP address and they'll have the communication that way. And now, you can go to example.com and you can go and see what the latest [craze] information there is. Go to the next one.

So caching speeds things up. So now, in a space of a horrendous 24 milliseconds, we got all that information we ask the root servers, we ask the .com servers, we ask example, we got all that information back and we were able to go to example.com. Well, 24 milliseconds doesn't mean a whole lot to us but it actually – when there's a lot of people asking questions, a lot of queries going on at the same time, it starts to add up at different levels of the resolution.

So one of the ways to speed that up is to cache it, so when we ask that question, "What's the IP address for www.example.com?" And the name server went and ask each entity for the question. When it got that information back, the

referral back, it holds that referral. So now, go ahead to the next slide.

Now, we're going to go – instead of going to www.example.com, we're going to do a file transfer, so going to ftp.example.com and we sum the query up. Go ahead. And we say, "It was the address to ftp.example.com." Go ahead in the next slide.

So now, we've asked the question already of the root servers and we ask the .com servers, the top-level domain servers and the recursive name server on your network has already got those answers, those referrals. So because of that, it doesn't need to go talk to a root server again to get the .com address. It doesn't need to go to .com again to get the example address because that already has that information you cached within its system.

So now, it goes directly to example.com and says, "Hey, I'm looking for ftp.example.com this time and not www." And so, it can go lose two steps off that process and go directly to the cached data, using the cached data, it goes directly to the example server and ask the question for a different one.

If we were going to a different domain, let's say nike.com. I want to see the latest in the footwear. And we've already gone through this resolution process a couple of times. The recursive name server already knows the address to the .com server, so it doesn't need to go and ask the root anymore for .com. It will

start its query at the .com servers and say, “Okay, I don’t have nike.com cached yet but I do have .com, so I don’t need to go ask the root. I’m going to go ask .com and now I’m going to go and ask for the nike.com main server addresses.”

So caching is at a layer level of two. So the only time that you theoretically should hit a root server is if you’re asking for a domain, a fully qualified domain that you haven’t touched any piece of that before.

So if I went to nike.com and I saw that we’re there and now I want to go to, let me make one up, crocs.pr and I want to go to the Puerto Rican store website for Crocs to look at their footwear and I’ve never been to PR, now, I have to start the whole resolution process over again and start at the root because I don’t have the top-level domain cached.

Once that top-level domain has been queried from the root, I’m going to hold that in my cache until – do you guys remember the Time To Live part in the beginning of this, the TTL? The TTLs tells the caching server how long this data is valid for. So as a caching server, I’m going to hold that data until my TTL expires. At which point, I’m going to say, “I don’t trust this data anymore. It could have changed. I’m going to throw that away and I’m going to start to process over again.”

So at some point, even if you have .com and .net, .whatever are the most queried domains, top-level domains, even if your network server, recursive name server has queried those before, at some point, the TTL is going to expire and you will be getting – your recursive name server will have to go ask the question again to get new TTL, to get new fresh information.

Now, I'm going to pause here. Does that make sense? It's all about who you know and if you don't know, you're going to ask for referral. We do have a question over there.

UNIDENTIFIED MALE: All right, thanks. In terms of ownership in your diagram, who owns the recursive name server? Would that be the users ISP?

STEVE CONTE: In this case, yes, it's the easiest way to say that. If you're at home and you're doing this, your ISP is probably your recursive name server. If you're at the office and you're doing this, your company's infrastructure is probably your recursive name server. So it really depends on where you are. If you're here in the conference this week and you're on the ICANN network, ICANN meeting services have recursive name servers here on site and so they're your recursive name server.

[Inaudible] anything online? No? No questions? We're good? Any other questions about resolution? This is really the core of DNS. This is what it's about is really the who you know. Do I know how to get there? And if not, do I know those who can get me there at the end of the day?

I could have just done this one slide and then we could have gone and had coffee. Go ahead. Keep going.

The domain name players – in the process and especially in these meetings here, you'll hear three terms used a lot. You'll hear registry, you'll hear registrar and you'll hear registrant. Registry is the entity that manages that delegation, that zone delegation. So if we look at it from a top-level going down, top-level domains, .com is the .com registry managed by VeriSign. .org is the .org registry and it's managed – I'm spacing out – it's managed by another entity and I can't remember. Sorry?

UNIDENTIFIED MALE: PIR.

STEVE CONTE: Thank you. PIR. They're managed by someone else. If we look at .hk, they're managed by someone else. If we look at PR, Puerto Rico, it's managed by a registry. It's the PR registry managed by

a different entity. Whoever that delegation takes it to is the registry.

The registrar is the primary agent between the registrant and the registry, so let's get down to the registrant for a second and then we'll go back to registrar.

Registrant is the holder of domain name registration. So for me, for conte.net, when I registered that domain, unregistrant on the one who has registered that domain and managed that, that domain file or that zone. In some ways, I'm the conte.net registry because everything beneath me I'm managing but since I asked for that domain, I'm also the registrant on that.

Now, the registrar is the body in between. Most registries won't let the registrant – won't have a direct communication between the registrant and the registry data. So the registrar is the body in between that will manage the request and the data changes and everything else between the registrant and the registry. Typical registrars, GoDaddy, Network Solutions – there's a whole bunch of registrars out there.

Many registrars will also handle more than one registry. So if you go to one registrar, you can register a multitude of different registries, domain names registries.

Those are the three big players within the namespace. In here we have – we also have resellers. Resellers will work on behalf of a registrar to resell that domain. They have an agreement specifically with the registrar whereas registrar has an agreement with either the registries if they're with like a ccTLD. They also have agreements with ICANN itself and the registrant just goes to the registrar. Go to the next slide.

So here, we show different ways in which the registry speaks to different entities. We have the registrant, which is everyday user, my mom, me, whatever, will go through the registrar to request a domain or to put data into that registry.

We have recursive name servers, the registry will then publish that data or subset of that data out in its name servers. So, it holds things such as private data such as contact information if you decide to keep that private. It will hold some other things but it will publish some things such as the name server records for its child zones. It will also publish – sometimes will publish WHOIS information. Sometimes that's held at the registrar level. It depends on the type of registry it is.

WHOIS – all the stuff on the right side of the screen here is accessible at some level to the common Internet user or the common Internet application. So if you're going to do a DNS query, it's coming out of the cluster of databases and

information, which the registry holds the information. It publishes it out to the name server. If that registry is also one that does its own WHOIS or manages its own WHOIS, it will also publish WHOIS information out to the Internet in which applications or people can query that. There are different types and different agreements on who holds the WHOIS. This is about DNS. We're not going to talk about WHOIS.

So we see here also registries as defined by Merriam-Webster, is a place where official records are kept and a system of keeping official list or record of items. So really, the registries is just an official record keeper for the purpose of DNS only, the purpose of record keeper for domain and zone type data. There's other things that they do but for the purpose of this tutorial, that's what their goal is. Go to the next slide.

Oh, look, we're done. So, it's all about who you know. If you're going to take anything out of this today, who you know and it's all about referring. If you don't know where you're going like at the very top level, how to get there and that's going to go talk to a root server and the root server is going to take you to the next step, and the next step, the TLD is going to take you to the next step until you finally you get to the answer to what you're asking for.

Any questions? We've got about ten minutes. You have a question. All right, do we have any online? All right.

UNIDENTIFIED FEMALE: Does the resolution goes the same for the new TLDs like .solutions, .doctors? Is it a dedicated server for [inaudible]?

STEVE CONTE: So you're asking if the new TLDs, if they acts the same as this?

UNIDENTIFIED FEMALE: Yeah.

STEVE CONTE: This acts as the same any data that's in the DNS. The query and response cycle is exactly the same. It doesn't matter if it's .com or .museum or .beer or whatever the new TLDs are. It's all in the root zone and it's all in as the same type of information, so you're going to be asking the same question, you'll just get a different response depending on where the servers are.

Any other questions?

All right, well, I'm going to [tease] the next session at 10:30. We have Alain Durand who's going to be talking about Internetworking. He's going to talk a little bit about DNS still.

11:00? Thank you. 11:00, you've got 40 minutes now, Alain. He'll be talking about routing and addressing, put your own thing here.

ALAIN DURAND: It's all about addressing, naming and routing, and sometimes we will use all those things interchangeably but they're very different. So we will talk about IP and if you're a lawyer and you think that IP means Intellectual Property, well, actually, you'll be in the right place because we will tell you that IP is Internet Protocol. So it's something a little bit different. It's not Intellectual Property rights. It's different.

And so, we'll try to give a broad perspective on all this technology to put it together and see how it really works.

STEVE CONTE: Thank you, Alain. So, 11:00, please come back for that. No questions online?

I want to thank you all for joining me this morning for our session in the first day and we've got a really good turnout here, so I appreciate sharing your time with me. Thank you for those who know DNS and not heckling me from the back. You know who you are. And we hope to see you back here in 40 minutes, so thank you very much.

[END OF TRANSCRIPTION]