KOBE – How It Works: RDAP
Saturday, March 09, 2019 – 17:00 to 18:30 JST
ICANN64 | Kobe, Japan

CATHY PETERSEN: Good afternoon, everyone. Welcome to How It Works on RDAP. We will be starting in a couple of minutes. Thank you.

Good afternoon, again, everyone. Welcome to our How It Works Tutorials on Registration Data Access Protocol or what we call as RDAP. Our presenter today is Eduardo Alvarez from ICANN's GDD Technical Services Department. Eduardo, take it away.

EDUARDO ALVAREZ: Thanks. Hi, everyone. Welcome. We're going to be reviewing a simple presentation on what RDAP is and how it looks, how it works and then at the end we're going to be having some time for questions, so let's start.

As Cathy mentioned, my name is Eduardo Alvarez. I am part of the Technical Services Team at ICANN. Gustavo, my colleague, also listed there, he's going to be giving us this presentation tomorrow morning in the 8:30 AM slot, so just so you know, there's a second session of this.

This is the agenda. We're going to start with the introduction, then we're going to cover some of the basics of RDAP. Then we're

going to see how the RDAP queries and responses look like, the elements that they include. Some of the features that this protocol has, making a special mention of differentiated access and then just a little bit on the ongoing work that has been going on around the RDAP topic.

So, why RDAP? As you know, WHOIS has been the go-to protocol for registration data on domain names. However, we noted that it has a lot of shortcomings that are listed here. Many of the registries have different formats that they use when displaying the registration data. Some of them don't support internationalization, so some of the characters don't display correctly, if there's foreign characters, for example.

There's also not support for any authentication, so there's only public queries for the general users. It includes lookup only which is when you're searching for a specific domain name in the WHOIS [inaudible]. You cannot really do search for a group of results that you're interested in. It's not standardized. It would refer with the reference to redirection which means, for example, if TLD registry says that specific domain name is sponsored by a registrar, they can only provide the registrar WHOIS URL for you to go query, but again, it's not standard or it may not appear at all.

It's also considered to be insecure because there's really not support for any authentication of the servers. Also, communication is not encrypted.

So, taking it back for the chronology of RDAP implementation to where we are today, we can look at these key dates. In September 19, we got SSAC's communication saying that we should have a replacement for the WHOIS protocol. Then on October 28, the board resolution adopted the communication from SSAC. Then until June 2012, we got the road map to implement published in SAC 3051 and then the development started within the IETF for RDAP which is something that we're going to look at a little bit closer further in the presentation.

Then, more recently, in March 2015, RFCs got published in the IETF related to the RDAP protocol. In June 2015, the work on the RDAP profile started which intended to map policy requirements and contractual requirements from WHOIS into how RDAP should behave.

Then, on 2016, on July, the first version of the RDAP profile got published. Then in August the Registry Stakeholder Group submitted a request for reconsideration regarding the inclusion of RDAP in this policy document called the consistent labeling and displayed policy.

We can kind of move a little bit forward. Last year, in May, we got a temporary specification for gTLD registration data where there is a requirement to implement RDAP following a [inaudible] profile, service-level agreements, and the reporting of registry statistics to ICANN.

On August 1, 2017 that same year, ICANN received the proposal from the Registry Stakeholder Group with support from the [Registrar] Stakeholder Group to implement RDAP and then much more recently in September 2017 the RDAP pilot started. Then, in August, a few months ago, 2018, a proposed gTLD RDAP profile was published for comment that was created by the contracted parties, meaning registries, registrars with ICANN. Then, a few days ago, February 27th, the official or the final version of the gTLD RDAP file got published and the ICANN Organization issued a legal notice to contracted parties to implement RDAP and that includes a deadline that by August 26th, this requirement of RDAP implementation has to be made, at least in the gTLD space.

So, moving on to a little bit on what is RDAP now that there is this requirement to implement it. So, this is just a protocol for registration data access. It's [inaudible] the IETF as we saw on the timeline. It's comprised of a series of RFCs. It provides a standardized query and response and error messages which is some of the issues or shortcomings that we saw that WHOIS had.

It supports secure access to data, meaning it's via [inaudible] or Web. It has extensive [inaudible] features which makes it easy to add more output elements to the RDAP responses and it also enables differentiated access which we'll see a little bit more in detail going forward.

There is a bootstrapping mechanism to find authoritative server for [inaudible] domain name entity IP network. Again, we're going to get more into detail once we move forward in the presentation.

Then, also, internationalization support is quite … It's included as part of the features. Now, the implementation status right now, as we saw on the timeline, the temporary specification for gTLD registration data includes a requirement for implementing RDAP.

Now that we have the gTLD profile available and published, we still have work that is ongoing on financing the SLA portion of the registry reporting requirements.

Moving on to the basics of [inaudible]. RDAP has a [inaudible] architecture which means that HTTP requests get sent to the server with the methods that we see on the right hand, [get post, hit], among others. It's an API for accessing information.

In regards of RDAP, we're interested only – this is a read-only protocol, so we're interested only on retrieving information for which we just get [methods] and [inaudible].

What does this mean? So, the RDAP, as I was saying, when you send us a request to the RDAP server looking for a specific domain name, you're going to receive a response with an HTTP status like the ones listed here just to provide some examples where it will tell you whether the object was found and that it's present or that it needs to be found in a different location which is the redirection that it's in [inaudible] or two or whether there was an error among other examples.

So, part of the queries that RDAP support as opposed to WHOIS which only supports the lookup search where you ask for the specific information about a domain that you're providing, RDAP supports lookup for that same functionality as well as search mechanisms. Lookups, like we see in the example, allow you to get the registration – well, to verify the existence of a domain name that you were providing and then get the registration data for that, whereas the search you can use patterns, for example, to retrieve the list of domains that match that pattern. For example, domain names that start with ICANN.

How do you ask RDAP for this information with regards to lookup paths? You can ask for a specific domain which we see in the first

example. So, you just [inaudible] base URL for the RDAP server to just add the path of domains, slash, and then the domain name you're looking for and you'll get the response whether the domain does not exist or the registration data for that domain.

The same works for the other objects that are supported in the RDAP protocol which are the name servers entities, which in this case – or in the domain name registries to represent [contacts] or registrars.

Then, IP networks and autonomous system numbers. These are also supported. However, they fall a bit more in the area of regional Internet registries, not really on domain name registries, so we're not going to go too much into those.

Then, segment for help which is just basic information of the RDAP server.

Then, on the other side, we have search paths. So, the same way you have a base URL, for example, like rdap.example and then if you add domains and then the query parameters which are following this question mark, you can search for name pattern for domains that match a specific IP for their name servers as well as in the second group we see search of name server objects by name, by IP address. Again, these are patterns which means you can get multiple results for those. Then, at the end, search for

entity objects which means you can search registrars or contacts by name or by identifier which is the handle here.

Now, a little bit more into detail. So, how does an RDAP response look? So, RDAP responses are in [inaudible] or java script object notation which is very similar to XML notation. I'm going to show some examples. It's defined in RFC 7159. It's just a text about name and value pairs which, as we can see in the second bullet, it's between this bracket you'll have a string stating the name of an element and a colon character and then a value that can take the form of another string of a number, of a bullion, an array of values or an object which is again kind of like another object or another [adjacent] structure within the value.

So, here's an example which will hopefully make it a lot easier to understand for those of you that are not familiar. If we wanted to represent just an everyday object like a book, for example, we would start by opening the bracket indicating that the [Jason] structure is beginning and then we have a series of name and values. So, the name here would be the title and then the value would be [introduction to Jason] then a comment to indicate that the next element is starting and then we can have a next name of an element which could be ISBN, the author, pages. We can see the different types of objects that we can have, strings, number, bullion value. Then, the last element, table of contents, we see it's an array of objects which is another [Jason] object. Again, we

have the bracket that's opening. We have a name and a value and then followed by a second name and a value which is a number and so on. With this structure, we can have a very organized way of displaying a book object, just to cover that example.

Now, this, to map it as how it would look on RDAP, let's start to take a look at some of the common data structures that RDAP would have, the name elements and what they mean on different RDAP objects.

So, when we're looking at an RDAP response, the object class name data structure is what's going to tell us what are we looking at. Is it a domain name? Is it a name server? Is it an entity? So, this is required element that needs to be included, such as, in the example before, as we saw the title, if we're looking at an RDAP object, we're always expecting these object class name element to be present. And these are the values on the right that are supported by RDAP protocol.

The RDAP [inaudible] data structure is the one that we rely on to understand what specifications the object is supporting, like I mentioned before, because RDAP supports [inaudible]. There are different specifications that can be created over time and the way this protocol tells the user which of the specifications it's supporting is by including an identifier in these elements. So, we'll always have this at the top-most level in the [inaudible]

object, we'll always have this RDAP conformance element and then an array of strings telling us that this object complies with maybe the standard RDAP RFCs or the specifications and then additional extensions can be added there, so that we know what is being supported and what is not.

The [inaudible] data structure allows us to add reference to other objects. Again, we have a static structure of information here where we can have a value, the relation type here, from the main object to this link in the URL that's going to click. For example, if we're looking at the result of RDAP domain search or lookup of test.example in this example here, a common practice to always include a link to the same object. So, basically, we're saying that the URL listed here in this link object, it's where the RDAP server is providing this response.

Then, additionally, because this is an array, we can add further link elements to other objects, for example, to the registry homepage, terms of service, or any other type of link that is required, is supported there.

The notices and remarks are very similar. It's also an array of values that will have a title, a type of description and links. These are used to provide general information of the RDAP service or the object.

The type, however, the values are based on an IANA registry, so we have a catalog of values that can be used for different scenarios. Then, in the description, you can have the text that you want to include as part of the notice or the remark on the RDAP object.

The difference here is that the notices element may only appear once in the main object of the RDAP response whereas the remarks can be added to include more information on objects that are included within objects.

For example, if we have a domain name, we know that inside it's going to have more entity objects to represent the domain name contacts and response in registrars. Each of these objects may have their own remarks element within.

Again, in regards of the internationalization feature, we have a common structure which is the language identifier. It will allow you to specify in the response whether the contents are in a specific language and this is [inaudible] in other RFCs to indicate the language, the script, and it also supports the regional aspect of it. In this example, for example, is Mongolian language, Cyrillic script, in the region of Mongolia and then you can add different combinations of this to reference any of the languages and regions as dictated in this RFC.

Then, the events data structure. This is included as part of the object to show different important dates on that object. So, looking at this object, at this example, if we are looking at the domain name, we will have two – well, this would represent that there are two dates included here if we're trying to look at the WHOIS values. We'd have a creation date of the domain name. It would be represented as the first [inaudible] of the example. We have the event action labeled as registration. They have an actor which means who triggered this event and when it happens. So, we'll have a registration date of this date in 1999 and then, as a second element of this array, we'll have the expiration date of the domain name 20 years in the future in 2019. Then we can have additional dates here as more events like update dates, transfer dates, and so on.

Again, the event action is also included. There is an IANA registry that dictates possible values for this element, so that again, RDAP [inaudible] will always use a standard way of representing the registries and the expiration date of domain names and so on.

Status object is also coming across RDAP object classes. It will indicate from a list of possible statuses that apply to, for example, in this text to a domain name that is active and has a client transfer prohibited EPP status assigned. But you can also use it in different objects such as entities for [contact] statuses or for name server statuses and so on.

This is a reference. Port 43 is a common data structure to include a reference to the original WHOIS server where this object may also be found, so in the case of domain names, it would point to the TLD registry, where the TLD registry is offering the WHOIS server.

Public IDs. This is used to show the attributes of the object classes in the case of the – or the most common case in domain names is the IANA registrar ID for registrar entities. Because identifiers are handles of entity object classes may not be standard or uniform to what we identified. For example, in the case of registrars, a registry may have a specific handle or identifier for each of the registrars that does not necessarily match the IANA ID. So, with using this common data structure, there's a way to provide this information in a way that is standard and people in the gTLD area would be able to access this information.

So, these are the common data structures, the ones that we saw. We put in this a little bit more together, specifically to the object types that are supported in the RDAP protocols. If we remember, a couple of slides ago, we have these five object classes. Each of these can include some of the common data structure that we just saw, but [let's see the] specific elements of each of these can include, starting with domain names.

So, if you query an RDAP server and provide the specific domain for the lookup, you will be able to retrieve the information related to that domain name or confirm that it's not found in RDAP server.

Some examples. A domain query would be just the domain name. Again, it has internationalization support so you can also have the Unicode characters if you're looking at a registry that supports IDNs. Or you can use the A labels. It should yield the same behavior.

In the response, however, you're going to have this [Jason] structure and if we go back to the example of the book object, a domain object would have these attributes. It would have a handle which is the identifier that is included in the domain name registry. The LDH name which stands for letter digit hyphen form, that's the label of the domain name that you would see in the zone file. The Unicode name in the case of IDNs, you would see the U labels or special characters in other scripts that are not in ASCII.

There could be the variance member. In the case of IDNs that support variance, there's a way to indicate these attributes as well. And I'd like to stress that these elements in these object classes, not all of them are required so they may not really appear in the [Jason] structure. For example, if IDNs are not supported,

you're not going to expect to see a [various] element in the [Jason] response.

So, the name servers, it's an object array where you're going to find each of the name servers associated to the domain name entities. This would include the domain name contacts and the sponsoring registrars. Additionally, there can be other roles for entities. We're going to see a little bit more of how these entities object look. DNS. Then, a network object which is not very common but the IP network for [inaudible] DNS.

Secure DNS. So, this is a separate element that can appear within a domain name. It supports having a lot of … Well, a lot of members or elements related to DNSSEC for domain names. If we look at WHOIS the way [inaudible] now in the gTLD area, the one that we would expect to see is the delegation signed to say if a domain name has a signed delegation or not.

So, for name server queries, again we have the lookup segment. We use this to identify specific name servers within a registry. To execute this search, we have to provide the name server name. It's a fully qualified host name. Both A labels or Unicode format is supported for search, for this lookup search, and then we can see some examples of how this values would look when querying a registry for a name server.

The response is not … Doesn't have as many elements as the domain name so we have again an identifier, the LDH name and the Unicode name for internationalization support. The IP addresses, for example, when you have a name server that is a [glue] record, you will have different IP addresses associated to the object. RDAP has a way to structure them, separated for IPv6 version of IP addresses and IPv4 and entities as well to include sponsoring – for example, sponsoring registrars.

Moving to entity queries, which is what would represent contacts or registrars in the domain name space, we are also able to search or perform lookups for this object by means of the handle or identifier. These identifiers just takes the form of a regular string. It can be numbers. It can be just the text. This is something that the registry defines. And in the response – again, apart from the common data structures that we saw before, we can also expect to see some of these members. The identifier or handle, [inaudible] which I'm going to show an example next which contains the contact information with, for example, phone, address, etc.

The roles object which is an array of strings, this is the one that's going to dictate whether this entity, how it's related to its enclosing object, the [inaudible] scenario is the domain name, which type of relationship it has, if it's the technical contact, if it's

the billing contact, if it's a registrant, the sponsoring registrar and so on.

An entity may also have additional entities within. For example, the common example that comes to my mind is if we have an entity representing a registrar, inside it has an additional entity with the role of the abuse contact so that you can have the contact of the registrar for abuse reports.

As [inaudible] actor. So, when you're looking at an entity, you can also have a list of events as we saw in the common data structure, that these entities are responsible for or has performed.

These last two objects are more related to IP networks and autonomous system numbers. So, we don't really want to focus right now, but you can also list these types of objects that aren't related to the entity that we're looking.

So, going back to the [inaudible] and the contact information for entities. It's supposed to go … Yeah, it went [inaudible]. Okay, we're back.

So, V card array. A V card is a digital representation of a business card. That's a definition that we can use. So, it's defined in RFC 650. It's a format that is [inaudible] in different types of software. It's a representation that includes contact name, organization, address, telephone, e-mail, a lot of contact attributes can be

added and are supported, but it looks something like what we see in the orange square here on the screen.

J card is the [Jason] representation, like a mapping or translation of the V card format into [Jason] which is the use of the brackets. Arrays end in – always have name, colon, and value.

The way it would look in the J card to represent addresses and contact information, it's a very structured format in which you always have this array of four elements where the first element will always tell you what information is being included. Then you have other parameters to include additional data. And in the last field, the fourth field of the array, you're going to have the actual value.

For example, in this text right here, we have the contact which the full name stands for. The text attribute and the name is [inaudible]. The organization value, you can have ICANN as the value. Then the telephone you can see it as the type is [inaudible] because it's a clickable representation of a telephone number. And then in the address, you will always have this structure of seven elements within an array that are always expected to be in this order stating with the Post Office box, [inaudible] address, street address, and so on and then whatever is not applicable, to just leave it empty but it needs to be in this format. This is how RDAP contact information gets represented.

I guess for the sake time, I'm not going to go over IP network and autonomous system objects, because as I mentioned before, these are related to regional Internet registries. However, they are supporting RDAP so there are ways of supporting these objects. They're available in the session materials if you want to look more into it, but for now, we're just going to stick to domain name registries.

These are the elements that are supported in IP network responses. This is how you do queries for [inaudible] and the elements that you can expect to find in the response.

Another of the helpful responses is the help which is also standard. This is a way for registries to provide information about their service to users, so whenever you have an RDAP service and know the URL, you can just go to /help and you should see a notices update as we saw before with the relevant information that the registry wishes to share, which can have links and then any relevant text. That's also good to have. It wasn't really available before with WHOIS unless they added it in standard responses by the server.

And error responses. So, whenever something goes wrong, there's also a standard way of reporting it. You'll have an HTTP response code object which are also standard but they would be displayed in this format that we see on the screen, an error code,

a title of the error, and then a description. In this example, we're seeing a bad request or an error but we could have a rate-limit error or other types of error here.

The point of this is having a machine readable response that can be processed by clients, for example, where at the same time, if a person encounters this error response, it's also easy to interpret.

So, features and concepts [inaudible] RDAP, going away with more. We listed some of them at the beginning, but I want to go a little bit more into some of these concepts.

Extenseability. It's one of the major features in RDAP. IANA registries are easy to [extend] its use to host a catalog, for example, of different types of roles an entity can have when related to other objects in RDAP, different types of notices, links, and so on. So, this doesn't really require that the protocol is updated. We can just extend or add more values in this registry that define which values can be used. For this, [inaudible] review process, which is some of the – these are the steps that are required to add more values.

So, this is the list of catalogs that I was referring to. For example, types of notice and remarks, statuses, event actions that we saw for registration and expiration dates of domain names, roles for contacts and other relations of entities.

This is the link. For anyone that's interested, can go to this catalog or registry in IANA on the values that are supported by RDAP for the different types of elements that are listed here.

For documenting RDAP extensions, there is also a separate registry that we can see here. Again, this also allows the implementors of different extensions to reference this registry, the IANA registry that is found here, and it allows to document all of these extensions that are available for RDAP implementations which can be used also for RDAP clients to reference, to know about this existing extension, so that they can be processed efficiently.

We spoke about the RDAP conformance element before. There is an identifier that illustrates the different types of specifications that can be supported by clients. This registry has also a reference to some of these concessions.

Different concept, bootstrapping. This addresses also some of the shortcomings of WHOIS which is knowing the authoritative server of an object. Before, to query for a domain name, you have to know where the TLD registry is offering their WHOIS service so that you can query, unless there's some of the most common top-level domains which may be including some WHOIS [inaudible] but otherwise you would have to know where the registry is publishing their WHOIS service. With RDAP, this is not really the

case. There is mechanism – again, based on an IANA registry, where you can list or find listed the TLDs and where the RDAP server is published. This, however, requires a registry to add their top-level domains there. In this case, it's for the top-level domains, but there's also other files that allow bootstrapping for other object types such as IP networks or autonomous system numbers.

In the case of registrar s, however, this is not the case, for registrars offering RDAP servers. Sometimes you would need to query the registrar, not the registry. For example, in the case of thin registries where the contact information or the registration data lies within the registrar and not the registry, you're going to want to know where this registrar is offering this service.

So, RDAP offers the capability of providing the URLs to the users. So, if you're requiring, for example, a dot-com domain name, you will get the reference to the sponsoring registrar's RDAP server so that you know where to query and get the registration data from.

There is no bootstrapping mechanism right now for registrars. However, the RDAP Pilot Working Group that has been working with ICANN has asked ICANN to come up with this temporary repository for RDAP [inaudible] for gTLD registrars. That way, just by knowing a registrar's IANA ID, you should be able to look at the

central repository and ask for the server where the RDAP service is being provided, from that particular registrar.

This is, in turn, being done to help address the requirement from the RDAP [inaudible] Implementation Guide which is part of the gTLD RDAP profile where the gTLD registries should include a link to the sponsoring registrar's RDAP server so that additional information can be found from the registrar.

RDAP object tagging. This has separate feature where we have this bootstrapping mechanism for domain names and other objects. It is not the same for entities. If you're looking for a contact, for example, RDAP does not have a way of – initially does not have a way of knowing which RDAP server to ask for the contact information or for a domain name contact or for a registrar, so the object tagging extension would be … It has been defined to address this issue. It's defined in RFC 8521 and it basically … It's a best practice to append to entities or objects handles a suffix with an identifier of the authoritative source of that object.

For example, if you're looking for a specific registrar under that … If you have a registrar handle and this handle is including the object tagging practice of defining this RFC, you would see an identifier separated by a hyphen and then a prefix indicating the

source of that object of that entity which could be the TLD registry or some other server.

Again, this is also relying on an IANA registry which the link is provided here. But it helps address this issue of knowing where – for [inaudible], which server needs to be queried for a specific object and information.

Internationalization. We saw in some of the examples, it's supported both in queries and responses. This also applies using the language element to contact information. It's better supported than a WHOIS because this is [Jason] formatted and by default is required [inaudible]. As a Unicode encoding, it supports the characters that are included in U labels and internationalized addresses and so on.

So, rate limiting. This is more on the security side or protection of abuse. This is also existing in WHOIS. RDAP also supports adding rate limiting which is including a pre-defined number of queries that a same user or source can do to an RDAP server. Once that number of queries is exceeded, then you're going to stop getting the responses for queries. Instead, you're just going to get an error for a pre-defined amount of time. Again, this is also similar to what we have today in WHOIS.

In particular, differentiated access is an interesting topic right now. To start, differentiated access to registration data, I want to start by going for the two elements that [inaudible] concept.

So, first, is authentication or verifying that a user is who they say they are. Then, the prime example is a username and a password. You have to check that these are correct in existing users and that the password matches that account and that's how you authenticate a user.

Authorization. Once a user has been authenticated, then there's the second step of deciding what this user has access to, what type of information. In the case of RDAP, the registration data elements that whether they have access to those or not.

So, in summary, this is like the difference or the key questions that each of these processes respond. The authentication is verifying identity and authorization is verifying that this person has access to these resources.

In RDAP, because this is a [restful] architecture and it's over HTTPS, we have authentication of the server supported. With HTTPS and the use of certificates or TLS, we can have authentication of the server. We can see it in the image here. Most browsers or the main browsers have an indication of whether HTTPS and a valid certificate is being used. This is something that we don't really have with WHOIS today.

So, for those that don't know, TLS relies on digital certificates. These are issued by CAs or Certification Authorities that are trusted entities. And identify a specific domain name when users are accessing that webpage. Again, a common indication is this kind of [inaudible] or green check we see when accessing some of these URLs.

In this case, when using data certificates, clients as well can use data certificates to identify themselves with servers by sending this information to the server which kind of replaces the need of having a username and a password. With that certificate, you're already including the information of who you are when sending a request to a server. This is also supported by the RDAP protocol today and offers an added element of security to the protocol as compared with WHOIS.

So, in combining these two elements, differentiated access in RDAP, it means having mechanisms supported that allow users to authenticate themselves and then getting different access to different elements of registration data. For example, some users may only have access to public information, whereas other users need access to the full registration data for domain name or a subset of it. That's where the differentiated access comes into play and the features that RDAP has of being capable of supporting these use cases.

So, this is part … To kind of put this all together, we're trying to … I want to show a little bit of how an actual RDAP query has. This is a web client that we have hosted, so that we can see how similar the behavior can be when compared to WHOIS. I have a couple of test domain names here. Again, this is a client. This is all in java script. It's running on the browser which is … It prevents the URL here where this client is hosted from knowing what queries I am making. So, for example, I know we have a test registry, assuming, at dot-test TLD. If I type a domain name that I'm looking for and then do a lookup, I will get … I don't know if this is big enough. I will get a response on basic public domain name information. Of course, this is all fake examples. This is how a client would – one way a client would format this information. However, this is what the … If I click down here, this is how the server actually returns the [Jason] response, which is a combination of all the elements that we were looking at earlier.

So, have an RDAP conformance element that's saying this complies with the basic RDAP specification. We have an [inaudible] telling us it's a domain object. The identifier in the dot-test registry of this domain. Some remarks. Links to this object and related objects. The registration date, expiration date, etc. This is all compliant with the examples that we were looking at before and up here we can have a summary of these objects.

Now, I'm going to do the same query again because I know that dot-test registry, which is part of the RDAP Pilot Working Group supports certificates, [inaudible] certificates to authenticate myself. But because this is the presentation laptop, I need to install it real quick. So, let's see. I hope there's no issue with permissions here.

But by adding the digital certificate that I downloaded from this test page – and this is available for public. Feel free to go to the URL that's in the presentation if you want to try it. You can also get the detailed certificate and do it.

By adding this to the detailed certificates of my browser here, I can import the one that I just … It should be there. Then, by checking the client certificate now that I have it in my … Hold on. I need to create an [inaudible] probably to start this.

So, this is the URL. [inaudible] starting the browser now that I imported the certificate. I need to see why it's not asking for the certificate here. Maybe it wasn't really [imported]. It would be easier if there weren't so many. But I don't see the one that I just imported. It should be here. There it is. It wasn't letting me. By adding the password, now it should be supported. I need to restart the browser here.

So, now that the browser recognizes the digital certificate I just added, it will tell me who issued it and other information. This

was all made for the purposes of the pilot. But when I confirm, now I get a much more complete response, including contacts. So, this allows the server to validate the certificate against what they have in the server, and once it deems that it's correct, it authenticates that I am a user that's authorized to access this information and the RDAP object includes all of the object. This is the V card arrays that we were talking about earlier. I can see the roles which identify the relationship to the contact and we can map it to each of the roles of the contact types here. Registrant, technical, administrative. The sponsoring registrar includes more data.

So, this is a very simple example of how RDAP offers support of these differentiated access to registration data. If I choose not to send my DDoS certificate again, I'll get the limited response in contrast to what we were looking at.

So, this is using detailed certificates. So, this is another TLD that I know participated in the RDAP pilot. So, if we look at nic.career as a public query, we'll see almost no information here that the RDAP server is returning. However, they do support test integrations with open ID which is a different mechanism of authenticating. Not with detailed certificates but with third-party authentication providers. In this case, for the purpose of the pilot, this TLD is supporting authenticating with Gmail and Hotmail or Microsoft type e-mail accounts so we can try one of this. This is

just validating the authentication part and not the authorization. It will give me different responses, although some of the elements are going to be redacted. As long as I provide a valid Gmail account.

So, for purpose of the test, I'm going to use this one. And now the RDAP server actually redirected me because it identified that I provided a Gmail account, so it just forwarded me and now I'm in Google's domain name authenticating myself, so the RDAP server or the RDAP client does not know my credentials and I'm just going to Google so that they can identify me as a user.

So, if I provide valid credentials here, I'll get the results of an authenticated response from the registrar – registry RDAP server. Again, this is back to the registry RDAP server that's providing the RDAP service for dot-career domain names and we get [inaudible] more complete. I'm going to collapse this because this is just a server response. We're not really seeing a visual formatting. It's just the [Jason] elements of this. But it helps identify how before we used to have a really small amount of elements here in the [Jason] [inaudible] and when we provide an authenticated query, we get a lot more information, although obviously because of authorization issues, while now I'm able to see the entities which are the object representations, I still see data that is mostly redacted here. But before all of these elements were not being included in the response. So, this is another alternative or

mechanism on how RDAP servers can support this authentication and authorization mechanisms, again, that WHOIS they didn't used to have.

So, going back to the presentation, just to wrap it up, we have some future and ongoing work that I wanted to talk about. The RDAP profile that came up earlier. So, this is a document or two documents, actually, that got created by contracted parties with the goal of standardizing how RDAP should behave.

As we saw, some elements or objects that can be included in an RDAP response can vary. Not all the time you're going to get all of the elements that we saw.

So, with the profile, the intent is to standardize which elements should always be included, which ones can be truly optional or as applicable and so on.

Again, this is for standardization interopability between gTLD registrars, registries, and registrars on how the service should be implemented, considering requirements dictated by their contracts and policy.

These are the two documents that got created as part of the RDAP profile. If you're interested in looking at more in depth, you can go to the link in the slide. They're available there. This profile got published, as we saw on the timeline on February 27th and it's

available for people to see. It's a recommendation that ICANN gave on how gTLDs, registries, and registrars can implement this service.

SLA, service-level agreements, and registry reporting does not consider part of the RDAP profile. It's just an ongoing work that is going on, how to finalize these two elements that are part of the temporary specification.

EPDP. So, this is a topic that is also [inaudible] a lot. There are plenty of sessions on EPDP this meeting. So, this is meant to define what's going to happen with the registration data directory services requirements while aligning or complying with the requirements that this GDPR European regulation implicates on this service.

For more information, I encourage you to go through the EPDP sessions. The last … Just recently, the final report was published, submitted to the GNSO Council for the review with a list of recommendations and then the GNSO Council approved this report and I believe it right now is submitted to the board for approval or it's an ongoing process, like I said, and more updates on this are probably available or will be available in the sessions related to EPDP. But this will definitely have implications on RDAP as they affect which contact information will be available and

who is it going to be moving from – to be either collected or transferred between these contracted parties.

The ICANN Technical Study Group. This is another group that got created to address the differentiated access element on RDAP implementations. Their goal is to propose a model of this unified access that will also [support] the requirements that will be dictated by this policy. In this case, EPDP on how to access public information and then how to grant access to non-public registration data to other users that do have a legitimate basis for accessing that information.

The TSG published a draft of this model, actually, two days ago. So, if you go to the link in the slide, you'll be able to see the draft of this model as well as minutes of their meetings and the work that they've been doing. So, that's also available for public.

Another source of ongoing work, the IETF. For those of you not familiar, this is the organization where technical people, the technical community, works forward to create different standards, RFCs, and ways of implementing different services, or look, for example, the RDAP specifications that were published in this organization. It's open to the public. So, anyone can really participate in this work. It's not a company that has employees or anything. It just comes of technical people and engineers just

working to find the best way of implementing processes and protocols in this case.

Within IETF, the registration protocols extension working group. That's where the RDAP work has been going on. Along with other stuff, but this is where RDAP is also being discussed. And there are some drafts going on. We can see a list of documents that are proposed to extend behaviors in RDAP. These are the ones that are currently being discussed in the group. Some of this is related to authentication as we see in the first one. The next three are related to search mechanisms for RDAP and the last one as well for search with regular expression. So, this is also a place where a lot of work is going into RDAP. It's worth mentioning.

For reference, finally, we have just links to open source projects here at the top. These are RDAP servers for engineers interested in looking at how others have implemented RDAP. These are some examples that we found.

DNS [inaudible] makes registry [inaudible]. Then, RDAP had client projects, so this is also a list of some RDAP clients that are already implemented. The first one is the one that we just saw and then others are actual clients that are in production.

With that, that's all I had. So, we have a couple of minutes for questions.

CATHY PETERSEN: If you have any questions in the room, please feel free to go to the microphone and kindly state your name and affiliation please. Thank you.

WERNER STAUB: My name is Werner Staub. I work for CORE Association. I have a question about the representation of contact data. You explained this is J card. And when I see the address in there, it's just a list of items. At some point, you find the postal code. You find the country. But we don't know where. How do we know where the country is, where the postal code is? Is there a standard for this to be a country code and so on? I should see a list. Which one is what?

EDUARDO ALVAREZ: So, with regards of the address, this is an ordered array and this is how a J card is defined. So, the order of the elements is always static. So, the country will always be the seventh element in the array for an ADR element.

WERNER STAUB: And then do we say that the country must be a code in the case of ICANN? Can we write any country?

EDUARDO ALVAREZ: Well, that's an ongoing issue as well. So, the B card and J card indicate that these seven elements needs to be the country name. In order to do the country code, there's ongoing work as to adding an additional information attribute to include just a country code and not the name here. But as of now, it's not really supported in the J card, to my knowledge.

WERNER STAUB: Okay. A second question is if you want to have a contact handle, is it possible to specify a contact handle as itself, a domain name or URL?

EDAURDO ALVAREZ: To define a contact handle?

WERNER STAUB: Which itself is a URL.

EDUARDAO ALVAREZ: Well, for contact handle, that's just a string.

WERNER STAUB: So, any string will be okay. There's no restrictions on … So, if you happen to use a URL that will be a string, it will be okay.

EDUARDO ALVAREZ: I will have to get back to you on that, but I have not seen that before. I don't know if there is really a restriction.

WERNER STAUB: Okay.

RICK WILHELM: Hi. Rick Wilhelm, Verisign. In regard's to Werner's question about the country code, the modification to the RFC to allow, to add country code to the V card is underway and is going to go last call. It's past last call and it's going to go to ISG after Prague, after the IETF Prague meeting coming up. So, there will be a change to the standard to allow to put in a country code element. The RDAP profile is written to allow the country code to be put into the seventh element because right now, none of the registries have the country. They only have the country code.

EDUARDO ALVAREZ: Instead of?

RICK WILHELM: Instead of the country, so you can put the country code which you have into the country element and be compliant. Then, of course, it would not be completely interoperable with other stuff.

Instead, the RFC is being rewritten to add the country code element to that specification. So, there will be country and country code. And then the RFC is also getting updated to add the contact URI because the temporary specification says that you can either use e-mail address or provide a contact URI. And of course there's no spot in there for a contact URI that would be compliant. So, that RFC is being updated as we speak and that's going to go in front of the ISG after Prague.

GAVIN BROWN: This is Gavin Brown from CentralNIC. A couple of things. First one is we published an open source RDAP client which is [inaudible] designed or use of service and [providing our programs]. I just recently in the last few days put up a web client that may be of interest. It's a little bit different to the one that [inaudible] actually has written. The address for that is client.rdap.org if you want to check it out and send us feedback.

The second thing is as registries and registrars are starting to implement RDAP, we're starting to see some of the pain points in the way that the specification is written and V card, or J card I should say, is the one that causes most of the pain. V card is a complex protocol in its own right and J card as a [Jason] representation of V card is also complex and annoying as well.

I put together a little strawman draft RFC saying let's just come up with our own representation of contact information. A few days later, someone else published something called J Contact which is another way of representing contact information in [Jason] without using V card.

What I would like to encourage is all the registries and registrars in the room who are currently implementing or thinking about implementing RDAP in [their systems], if you come across things that are annoying and painful, the [inaudible] working group would really like to hear your feedback because the more feedback we get from implementors, the more likely it is that there's going to be enough momentum in that group to fix those things by producing a new version of the standard or an extension that takes away V card and replaces it with something a little bit less painful to implement.

EDUARDO ALVAREZ: Thank you for those contributions.

ANGELA: Hi, my name is Angela from Botswana Communications Regulatory Authority. I have a few questions for you. First one, would you say that you observed a smooth transition regarding the RDAP in the [inaudible] to generate TLDs? Second one is

should also ccTLDs expect a similar transition and when do we foresee this? What has appeared to be the most frequent queries or challenges regarding this transition from WHOIS to RDAP? The final one, since RDAP allows object tagging, does this mean that I can query from different registries or registrars within a single query line? Thank you.

EDUARDO ALVAREZ: That was a big list of questions. I'm going to start by the ones I remember and then I may ask you to come back. For ccTLDs requirement to do RDAP, there's no such thing. They're free to do so, but there's not really any requirement that ICANN can ask of them to do. Some ccTLDs already have their RDAP services available, [inaudible] IANA registry but there's really no guarantee of who else is going to do it or when. That's completely up to them.

For gTLDs. So, the transition, if we can call it that, it's just really in the very early stages. Right now, we only know of the requirement that ICANN issued of implementing RDAP by August 26th this year. However, that does not mean that WHOIS is going to go away. It's a separate situation. So, that's something that we're going to have to observe as time goes. Right now, the only requirement that's there is just to have an RDAP service implemented by the deadline of August 26th.

Object tagging. So, it doesn't mean that a registry will have information of other registries or redirect you to another server. It's more targeted to a client to identify. For example, the client that I was using as an example as a very basic one and doesn't really have support for entities right now. But if we were to add one, that client would have means of identifying by the handle of an object which RDAP server to query. RDAP servers could in turn add some logic to process object tagging, but I guess the purpose is just providing the mechanism of finding who the authoritative server for that obvious is. That's what is basically being supported by this practice. I don't know if I left anything out or if … Okay.

So, the question was queries, inquiries, or challenges that ICANN has received on this transition to RDAP. I am not aware, but this wouldn't really go to me, so I'm not the best person to respond to that but I have not heard of issues.

The legal notice to implement RDAP was issued just a couple of days ago on February 27th I think, so it's a really a short timeline for questions and concerns to come back, I guess.

GAVIN BROWN: Hi, this is Gavin Brown from CentralNIC again. Maybe I misinterpreted the previous question about the transition. I thought it might be useful for people in the room … I have a little anecdote that I got from ARIN. It wasn't RDAP then. It was called

RESTful WHOIS. It was the predecessor to RDAP. When the US, the North American RIR, deployed their version of RDAP they say something like 50% of all their query traffic transition from Port 43 to RDAP within a fairly short timeframe, within about six months to a year. So, I think registries and registrars who are looking at deploying RDAP for their systems, you might see a very rapid upswing in traffic going to your RDAP system but they'll be a very, very long tail of Port 43 traffic as well.

So, some people have said to me, "Is WHOIS going away on the day that RDAP goes live?" The answer is no. The two are going to live together a bit like IPv4 and IPv6. They're going to live together for a very long time because there's going to be this long tail of shell scripts that are written and put in a [inaudible] job on a server and they've been left for years and years. And those will continue to generate traffic to Port 43 WHOIS. RDAP is the future but WHOIS is the present and will be the present for a long time to come.

EDUARDO ALVAREZ; Thank you.

CATHY PETERSEN: Do we have any other questions still from the room? I think that will conclude this session for our How It Works on RDAP.

Tomorrow we will have another session for RDAP starting at 8:45 AM here in the same room. So, if you want to catch another version of this tomorrow at 8:45 AM, please feel free to come back. Thank you so much.

EDUARDO ALVAREZ: Thank you.

[END OF TRANSCRIPTION]