ICANN74 | Policy Forum – GNSO: IDNs EPDP Working Session (2 of 2)
Wednesday, June 15, 2022 – 10:30 to 12:00 AMS

JULIE BISLAND: Hello, and welcome to the IDNs EPDP Working Group Meeting Session 2 of 2.

Please note this session is being recorded and is governed by the ICANN Expected Standards of Behavior. During this session, questions or comments submitted in chat will be read aloud if put in the proper form as noted in the chat.

Taking part via audio, if you are remote, please wait until you're called upon and unmute your Zoom microphone. For those of you in the main room, please raise your hand in Zoom, and when called upon, unmute your table mic. In the secondary room, please raise your hand in Zoom and go to the standalone mic when called upon.

For the benefit of other participants, please state your name for the record and speak at a reasonable pace. You may access all available features for the session in the Zoom toolbar. With that, I will hand the floor over to Donna Austin.

DONNA AUSTIN: Thank you, Julie. Welcome, everybody, to our second IDN EPDP session. Today Sarmad is going to provide some foundational information for the EPDP team that we hope is going to help set a baseline for our discussions as we move into the management of IDN variants at the second level and the IDN table implementation. So IDNs and their management at the top level and at the second level is not the same. And so as we move into the discussions around treatment of variants at the second level, we thought it would be helpful to level set. So that's primarily what we will run through today. So Sarmad will provide a foundational presentation.

Then what we've asked our contracted party, members of this working group who are Registry and Registrar team members, is just to share with us the practicalities of how it works at the moment for how they operate IDNs and they use at the IDN table and its implementation. So that's what we hope will be the second part of this session.

So we do have a 90-minute session. What I thought might be helpful—we'll see how we're going and I'll get a temperature of the room, but perhaps after we get through the first hour, we might just take a quick break for two or three minutes, and then come back in just to give everybody a little bit of a breather and just to get away for a second or two. So I'll see how we're going an hour in.

One thing I also would like our members of the EPDP team to do, if they can, is in the participant list, if you can actually identify whether you are an EPDP work team member, just so that you know that others that are attending today can see those that are members of the EPDP team and following this and then we can get a sense of who else is also in the room. So if folks wouldn't mind doing that, that will be greatly appreciated.

So with that, Sarmad, I assume that you are in the room there somewhere. So could I ask you to take it from here, please?

SARMAD HUSSAIN: Sure, Donna. Thank you. This is Sarmad Hussain with ICANN. Thank you for the opportunity to share an overview of the IDN tables. So we'll take you through some of the basics, and then also some details which may be useful for all of you to know as you, of course, start deliberating the policy around this work. So let's get started.

I think there has been many ways that people think about IDN tables. We've heard in our conversations even within the working group, also the EPDP Working Group, that these are referred sometimes as IDN tables, sometimes as variant tables, and sometimes as LGRs. And just to, I think, clarify a bit, when we talk about any of these, we are really talking about the same set of rules and data. So, when we talk about IDN table, it's the same thing as variant table or LGR. So, that's just a bit of a background just to clarify the terminology.

When we are defining a label in a domain name, it can be done at multiple levels. You can define a label at the top level or the second level or the third level and so on. And there are certain rules which one has to follow to define these labels. The rules are generally the same for second or third level, for example, but rules for the top level are slightly more conservative.

We start with the ASCII domain names, which I guess all of us are very familiar with. The ASCII domain names at second or third level, for example, can be formulated by letters, digits, and hyphened letters A through Z, also capital A through Z. So, uppercase/lowercase. This part of it, just listing which letters are allowed, this part is normally referred to as repertoire of an IDN table.

Then there are certain rules which can also apply to a label. For second level or third level, for example, there can be additional rules which apply on a whole label. So for example, the label length is constricted to 63 octets. There can be other rules on, for example, the use of hyphen. So for example, there could be a rule which says that a label cannot start with a hyphen. Those are some examples of, basically, how an IDN table or a set of rules can be formulated to develop a label.

For the top level, it's a similar process, but from RFC 1123, it says that top level labels must be alphabetic. Therefore, they're constrained to letters A through Z and digits, and hyphen are not allowed in ASCII top-level domains. The length remains 63 at max.

So when we look at the complete ASCII table, you would realize that there are many more characters or letters or symbols which are available in an ASCII table. So, one of the things which the label rules do is define a subset of characters which can be allowed in forming domain label. And this is, as I was sharing earlier, normally referred to as the repertoire or the character set which is allowed for labels. And even in ASCII, you can see that not everything is allowed, right? There are many punctuation marks, many other symbols or special characters which are not allowed in domain name labels. This is for the second level. If you go at the top level, it's even more constricted, as we shared earlier, that only the letters are allowed, and not even the digits and hyphens are also not allowed, for example, in the ASCII TLDs.

So moving on, we get to the Internationalized Domain Names. For Internationalized Domain Names as well, we need some rules which can govern what kind of labels, how to formulate a label. So this is an example of a Thai domain name. Even in Thai domain name we'll have multiple labels in a domain name we'll have second level, third level, possibly, and of course the top level.

What's happened is that the same spirit which was applied to the ASCII domain names has been in a way applied to the Internationalized Domain Names as well. The reason I say spirit is because you can't apply the same formula because some script, even not all scripts are alphabetic in the context of other scripts beyond Latin. There are other types of scripts so you can't really just put the exact formula across all the scripts. But in any case, for second level domain names, what IDNA 2008 does is it says based on Unicode, a valid U-label can be formulated by using normally letters, digits, and combining marks. Whereas for the top level, again, since it is reduced to alphabetic, basically, it says that the top-level domains of a root should be restricted to just the characters, which is letters and marks in Unicode but cannot contain digits.

You can imagine that we have similar tables for scripts. In Unicode for all the different scripts, we have Thai table, we have the Chinese table, Cyrillic table, and so on. For each of those tables, we'd have to define a subset of characters, and then rules. That formulation which we use to define repertoire variants and so on for the second level are normally referred to as second level IDN table or LGR. And the rules which we formulated for top-level domains, we call that the Root Zone LGR. So that's sort of how the whole thing is set up.

What's an objective of an IDN table? At least the way we see it, there are two clear objectives. One is the usability objective and the other is the security and stability objective. They obviously need to be balanced with each other for designing the "right" IDN table. So the usability aspect is, of course, that second level domain names should be enabled in local languages and script. But to be used for communities globally, but on the other hand, these need to be really designed in a way that they keep the Internet domain names secure and stable. So that's sort of the trade off which we have to play with when we are actually designing these IDN tables.

So what does an IDN table do looking at it at a very high level? So if you have a label, any level, t1 in this case, what you would want to do is eventually you will define an IDN table and once you actually have integrated that IDN table in your system, a potential registrant will send you a request that "I want to register t1" and will say, "Okay. I want to you register this t1 as an Arabic language label." What the system will do is it will load up the IDN table for the Arabic language and it will run t1 through that IDN table to see and it will find out first that whether it is a valid label as per the IDN table or not. If it is a valid label, then it will also try to find out whether it has any variants. If it has any variants, so in this case, there are four variants, it also determines whether any of those variants are usable, meaning allocatable, or they're not really usable but they need to be blocked for security reasons so that nobody else can get to them. So that's the difference between allocatable and blocked variants. All that information, the validity of a label and the set of variants and the disposition of those variants as either allocatable or blocked, all that is somehow provided through an IDN table. So that's the information which is codified in IDN table to enable those answers to a request.

Okay. So looking at it a little more deeply. As we've already seen, IDN tables define which labels can be registered for a particular language or script at the second level under a top-level domain. So let me just explain that a little more. It says a few things and I just want to point that out.

First, that an IDN table can be defined either for a language or for a script. It is up to the registry operator to decide whether they want to design an IDN table for a language or for a script or for both. So that configuration really is determined by the registry operator based on their business models. But it could potentially be defined at two different layers: language level or the script level.

Second is that it is up to the top-level domain or the registry operator to choose which languages and scripts to support through the IDN tables under the TLD. So some may just not do any IDNs at all, just do ASCII. Some may do three languages, some may do five languages and five scripts, some may do a hundred languages. So it really varies significantly across a top-level domain or registry operator to make a decision on who they want to serve, and that's a business decision, and that's really their decision to make. The motivation of again designing these IDN tables is that we want to beyond business also manage or ensure that the labels which are being offered make sure they give a secure and stable experience to the end users. So we are balancing security, stability, and usability.

Going into the IDN table itself, at a higher level, it contains five pieces of information. The first piece of information is for a particular IDN table, it tells you what is the language and/or script of that IDN table. So you can, for example, define a French IDN table using Latin script. You can define Arabic language table using Arabic script. You can define Russian language table using Cyrillic script. Or you could actually skip the language and say that "I'm just going to define a table for Cyrillic script," and it will just cover all the languages which are used to write Cyrillic script. So again, as I said, that decision is done by the registry operator because that's a business-oriented decision. But whatever decision they make, they have to specify the language and/or the script inside the IDN table to clarify the scope of that IDN table.

Then there is some metadata and description. It's a free flow text. So the language and script tag is not free flow text, by the way. It uses standard ISO standards to specify which language and which script. So there’s ISO 639 and there’s also a standard for scripts. So those ISO standards are used to specify language and scripts. That part’s pretty formal. The script tags are normally taken up from the IANA script language tag registry.

The metadata description part is pure text. So it’s a free flow text normally, but it also has some other fields like version number, date of publication, and those kinds of details in addition to just a pure description of what the table is doing and what it is for.

Then comes the repertoire. We’ve already talked about the repertoire. It is a list of code points for that particular script or language which are allowed to form labels. Normally, it’s just a list of code points either from ASCII or Unicode.

Then, in addition to repertoire, it also has a list of variants. Variants, as we’ve all been talking about, says that these are two different code points which are considered the same by that particular script community, and that those are really defined by the script community. Of course, what we do is we codify that inside the IDN tables.

So here are two examples. 629 and 663 are identified as variants by the Arabic script community. And even if you look closer, those two characters actually look identical. That’s one of the reasons it is actually identified as variants, because if they’re not variants, obviously they're indistinguishable by end users and that if not identified as variant labels, they can potentially cause security problems.

Then there are rules and there are two kinds of rules. There are rules which are linguistically motivated and then there are rules which are technically motivated. An example of a linguistic rule is Lao language or Lao script is used. It is one of the tonal languages and it actually has, therefore, tone marks in its writing system. But the way it works is that in the writing system, the tone always comes on a syllable. So, the tone mark, the way the writing system works can only, for example, be written on top of a main consonant. Without a consonant in writing the tone, you can't really write at all. So it's sort of an inherent property of that script. And that's codified in the rule, which says the tone mark follows a main consonant in Lao script, for example, but cannot occur by itself.

On the other hand, the technical rules are obviously not linguistic in nature but more motivated by standards and technical considerations. For example, RFC 5891, which is part of our IDNA 2008, says that the Unicode string must not contain hyphen in the third- and fourth-character positions. That's a purely technical requirement. It's got nothing to do with linguistics.

So there can be these rules which apply at a label level. They're not rules which are on specific concerns, specific characters. In any case, those collectively formulate what is an IDN table. So IDN table would generally contain most of this or all of this information. Sometimes rules or variants could be skipped because not all scripts and languages actually have variants and rules. But repertoire would certainly be there and language and script tag would be minimally there as well.

So moving on, we can represent or codify this data in multiple ways. There are multiple formats; they're based on different RFCs. They were initially RFC 3743, and then eventually 4290, which have been used to codify this data. Now, those are all informational RFCs. More recently, RFC 7940 has been formulated, which has also introduced the term Label Generation Rules, which was initially IDN tables. This is a much more formal mechanism. We'll look at it in a little more detail later. But interestingly, another thing which is different in this RFC from the others is that RFC 7940 is not an informational RFC. It is a Standard Track RFC, which, of course, has a different implication on usage. So looking at these three—

DONNA AUSTIN: Sarmad, sorry. Just before we move on, can you just go back to the previous slide, please?

SARMAD HUSSAIN: Yes.

DONNA AUSTIN: So just a question. Whereas the Root Zone LGR has been developed to apply across TLDs and provide some consistency, IDN tables at the moment are at the discretion of registry operators to develop and implement themselves and they only apply to the second level. Is that correct?

SARMAD HUSSAIN: Yes. That's exactly correct.

DONNA AUSTIN: Okay, all right. So there may be IDN tables that are similar across the registry operators but there's no requirement that they have the same-same across all the registry operators.

SARMAD HUSSAIN: Yes. So for the same language or script, two different registry operators can actually have different IDN tables. That depends on their business needs. So just to give an example, you could have a country or you're targeting a community which is speaking a particular language but it actually also has population which are immigrants, a large population of immigrants from another country. So even though the community you're focusing on may have a particular character set, but because they have a large population of immigrants which are using some additional characters in the same script, from a business perspective, you may want to expand or have a different design of the IDN table vis-à-vis which characters to include depending on how you want to target that community.

DONNA AUSTIN: Okay, all right. Thanks, Sarmad.

SARMAD HUSSAIN: Sorry I didn't see your hand. But if you have any questions, please feel free to jump in and I'd be more than happy to take questions. It's better to do the questions during the presentation rather than at the end. Hadia, please.

DONNA AUSTIN: Okay. Thanks, Sarmad. Of course, that applies to everybody else in the room or on the call, but we'll be prudent in the use of the time. But I see we have some other hands. So do you want to manage that queue, Sarmad?

SARMAD HUSSAIN: Sure. Hadia, please.

HADIA ELMINIAWI: Thank you, Sarmad. So my understanding that also the registry would provide the IDN table for the second level domains to

ICANN, right? If it's correct, I'm thinking here of when applicants are applying, I don't know if the registry will set a tool for them, or they just apply in the registry, looks if it's consistent with the table or not, or whether this could happen through ICANN, though it's a second level domain and it's not really in the remit of ICANN. Thank you.

SARMAD HUSSAIN: As far as the registrant interface is concerned, that's managed by the Registries and the Registrars. I think we can maybe potentially park that question in the second part of the session today when Registries and Registrars actually have the floor, because then they could actually answer, for example, how the different mechanisms which exist for that purpose, if that's okay. Okay. So we also have Yaovi in the queue. Yaovi, please.

YAOVI ATOHOUN: It was a follow-up question. But based on your answer, I may wait for the next session, because my question was the impact of the fact that we can have different IDN tables at the second level from the registry side, what can be predictable impact on the whole DNS system? But I think that this can be in the other session as you mentioned. Thank you.

SARMAD HUSSAIN: Right. I think after we complete this presentation, the next part of the session does invite Registry and Registrar representatives. So sure, I guess we can take that question at that time. But as I said, this is largely motivated by their own business requirements. So that's not really a technical requirement per se. What we are focusing on this particular part of the presentation is more from a technical standpoint.

So I don't see any other hands. Hadia, is that a new hand? Okay. Then we'll continue. Then going into a little more detail, basically RFC 3743, RFC 4290, those are the previous formats. As I shared, both of these are informational RFCs. Both of these are actually based on text format, having these in text format means that you cannot automatically process these by machines. You will actually need to write code and/or process it if you're just taking a look at it and you'll probably do some manual processing. That was sort of a limitation which was identified. Because of that limitation, this new format was actually developed, which is a Standards Track. What this new format in RFC 7940 does is that it actually codifies all the information in an XML format and allows to define code points, variants, as well as rules in what is machine readable. So that provides a very significant advantage over some of the earlier format which are just purely text-based.

Just to show some examples. You will see that there's a list of code points in IDN tables. In 3743, the variants are, in a way, separated by semicolon. But the important thing is that if you look at the top there, the rules are actually written in plain English. So if you look at that, there is actually a hyphen rule which says, “Label must neither start nor end with hyphen.” 002D is hyphen. So, that’s just plain English rule. And it’s just not possible for a machine to actually parse that and understand that and actually apply that rule on to a label.

Similarly, if you look at RFC 4290 format, here also you have variants. For example, there’s an Arabic table and it’s saying 0030. If you look at the second line there in the list of code points, it says 0030 is a variant of 0660. Basically, what that is the digit zero in what we use in ASCII, and the digit zero which we use in Arabic script, it’s saying both zeros are variants of each other. So is 1 and so is 2, and so is 3, and so on.

Then there is a rule at the top which says that you can actually have a label myname123 and then you can also have myname123 with 123 in Arabic version. But you cannot mix labels so you cannot have 1 in Arabic, and 2 in ASCII, and then 3 in Arabic as well. So that’s a rule which is captured in text, but again, it is not possible to machine parse it. Whereas when we actually come to the RFC 7940 format, everything is actually codified in XML variants and rules are also codified in XML which become eventually machine possible.

We're not going to go into too much detail. But just to give you a bit of an overview. So I guess the question is why XML format? And then we've talked about that a little bit as well. But I think when you're writing things in English, there's always possibility of misinterpretation or non-consistent interpretation of the rules. If I write a paragraph and I want to say a particular rule, somebody else reading that paragraph may interpret the rule slightly differently. Therefore, the implementation could potentially vary. What happens in LGRs is that since it's actually done in an algorithmic fashion, when it's implemented, there's less likely chance that it actually will be inconsistent across different implementations.

Also, since the whole thing is machine readable, the LGR format can be developed, reviewed, compared automatically by tools, and manual inspection is not required. So not only does it allow you to process it but you can actually compare two different IDN tables and say whether they're the same or not very easily. Whereas if some parts of these IDN tables were in English language, then you can't really compare it and so on.

It's straightforward to test LGRs on labels automatically if they're in RFC format. It allows flexibility in encoding. It allows almost equal flexibility to encode any kinds of rules, as we've seen, because we've actually encoded already 26 scripts and rules on LGR in XML format. It is has been reasonable to capture the rules which are required across those scripts. So it's reasonably flexible as text.

XML format, of course, if you look here, this is not human readable, right? We don't expect people to just go through and say that, "Okay, I understand that." But there is a very easy mechanism to convert this data into an HTML format like a web page. We actually have tools which do it, so it makes it very human. There's easy ways of making it human readable as well. So let me stop here. There's a question, a raised hand. Anil, please go ahead.

ANIL KUMAR JAIN: Thank you, Sarmad. As you have said, Root Zone LGR is the latest development in these tables. My question is that once a script gets a RZ-LGR, they have the old manual IDN tables also prepared previously. So do we have to delete the IDN table previous once the RZ–LGR is prepared, or both of them may exist together? Thank you.

SARMAD HUSSAIN: Actually, again, that's a business decision. These formats are available at ICANN. We support all the three formats still, and we will obviously continue to support them. So it is up to the Registries to decide which format they want to take forward. Again, I think as far as the operational details are concerned on what happens on the ground, maybe that's also something I'll defer to the next part of the session where Registries and Registrars can actually share their experience on whether they're preferring one format or the other, or if they're transitioning, and so on so that those are again operational, more operational kinds of decisions. If that's okay, we'll come back to it. Satish, please.

ANIL KUMAR JAIN: Thank you.

SATISH BABU: Thanks, Sarmad. I was wondering if the database, this XML schema for the Root Zone LGR, is the standardized schema across all languages and scripts?

SARMAD HUSSAIN: It is a standardized schema. It is what RFC 7940 actually defines that schema. All the Root Zone LGR tables or LGRs, which have been defined for all the 26 different scripts, are strictly obviously adhering to that schema. Okay. So I don't see any more questions. So let's move on.

Who develops the IDN tables? I guess some of this is already answered through the questions, but let's still go through it. So we've already seen that we have IDN tables for the second level, and then we have obviously IDN table for the root zone, which we call the Root Zone LGR. IDN tables at the second level developed and used by the Registries for the second level, and also possibly for the third level. So they are actually developed by the Registries themselves.

What we do at ICANN is that we've done some homework in collaboration with the communities which use these scripts. So we've given some sample, three pre-vetted for security IDN tables in XML format, which we call reference LGRs, which are available for the registry operators to consult while they're designing their own IDN tables. The advantage, again, as I said, of looking at these reference LGRs is because we've already worked with the target community and they've already vetted for security and stability considerations based on the input from that script community. So, those are also available.

The only constraint in IDN table design from a technical perspective is that IDN table should be secure and stable. The linguistic content actually is more motivated by the business needs. So, from an ICANN perspective or when we are taking a look at it for second level IDN tables, we're more obviously taking a look at it from a security and stability point of view.

Root Zone LGR is of course for the root zone, not for the second level or the third level and so on. This is actually developed by the script communities and it is anticipated for use by ICANN for root zone, of course dependent on eventually the policy recommendations. So SubPro of course already has suggested use of Root Zone LGR for the top-level domains. And that policy, as you all know, is under consideration. And IDN EPDP Working Group has also, of course, considered Root Zone LGR for that purpose for the existing gTLDs as well. But those recommendations are still with the working group and under consideration.

The Root Zone LGR, this was a very well defined procedure in which the communities were really involved which use the script to define what the solution is. I see many of people here have actually been part of that process. But in case you're interested, there's a link here to see the detailed procedure. So just to give you an example of how IDN tables work.

DONNA AUSTIN: Sarmad?

SARMAD HUSSAIN: Sorry.

DONNA AUSTIN: We have a couple of hands up. Thank you.

SATISH BABU: Thanks, Sarmad. Can we go back to the previous? I refer to the last line of the first set of points there. IDN table should not have any security and stability problems. That is a prescriptive statement. How is it enforced? Is there any validation of these tables at some level?

SARMAD HUSSAIN: Yes. The way it works for gTLDs—and that process will be covered later in the slides—but the process is that whenever the gTLD has to offer an IDN table, it basically designs and develops its IDN table. It is actually shared with ICANN for review. We review it for security and stability considerations. And if there are any, we identify any issues. We share that back with the registry operator. So there's a discussion which we have with the registry operator and then come to common conclusion. And based on that, then once it is actually finalized, it actually is added to Exhibit A of the contract for that registry. So it is actually a very formal process because it has contractual implications. Dennis, please.

DENNIS TAN: Thank you, Sarmad. On the same point about the IDN which should not have any security and stability problems. Without elaboration and context, it looks like a really strong statement and requirement. So perhaps we should elaborate, in the context of IDN table, what secure and stable means. Because any security, it just opens up the Pandora's Box, if you will, any sort of security issues that might some people might think of.

SARMAD HUSSAIN: Sure. Again, I think as I shared earlier that in some specific cases—that's a good point, Dennis—it does require some discussion with the registry operators. The way I think we will look at it is we normally work not only with the registry operators but also with the script communities to advise us on what they think is a security, we know what is a secure and stable solution for that script. We use that as a potential guideline for us as we move forward, but then when we receive IDN tables from the registry operators, there is further discussion in case there is some concern identified around security and stability, there is a discussion with the registry operator. As I said, we eventually would like to come to a common solution, and then move forward. Any comments? Otherwise, let's move on.

Okay. I wanted to give a couple of examples on security on how IDN tables are designed. So if you look at this, it's like a mini Latin script IDN table which has a hyphen and four characters: b, c, i, and ı (dotless i). It has a variant definition that i and dotless i actually are variants and the variant is blocked in one direction and allocatable in the other direction.

Then there is a rule that hyphen cannot be at the start of the first position of a label or start of a label. So then, if a registrant gives a label "bbc" to this IDN table, the IDN table says that yes, this is a valid label because it has all the code points are in the table. It has no variants, of course, identified. And the rule that hyphen cannot be the first position of the label is also satisfied.

If a registrant, for example, enter "cio" to it, this label is rejected because o is not part of the repertoire. And so it says that "There is an out of repertoire character, and therefore, this label is not allowed."

Give it a hyphen "-bbb" as a label, it is also rejected even though bbb and hyphen are all in the repertoire. But there is a rule which says that hyphen cannot be the start of a label. And based on that rule, the label is rejected, and therefore, that label cannot proceed for registration.

Looking at the same example a little more, from a variant label perspective, if we have "bbc" that has no variant labels, if somebody enters "cbı," you can see that ı (dotless i) has a variant label so it creates cbı and says that that's valid and it identifies that cbı is a blocked variant based on the definition of variants.

Then if you have label like "cicı," that creates four possible variants or actually one main label, which is good to go. And then it creates one allocatable variant where ı (dotless i) is changed to i, and that’s an allocatable change. Whereas in the other two cases, the i's can change to ı and that’s a blocked case. So two blocked variants are generated and one allocatable variant is generated in addition to the original label which is marked as valid through this IDN table. Let me see. Does everybody understand that example and how this works? So this is how an IDN table works.

So then, once you have allocatable variants, those allocatable variants can actually be activated. So the IDN tables may generate both allocatable and blocked variant labels. Allocatable variant labels can be activated, meaning they can actually be turned on. Blocked variants should not be turned on. They’re just blocked and not available for anybody else. Whether allocatable variants are turned on for use is really determined by both the registry policy which are offering those IDN tables, and also by the request of the registrant.

So there may be actually a registry which says that “We are not going to offer any variants and we will just block all of them.” In that case, a registrant cannot even turn on allocatable variants. But the registry may say that “We are going to allow for registrants to activate variants which are allocatable.” And in that case, if there are five allocatable variants generated for a label, the registrant may request that “Can you please activate this one label for me or these two labels for me?”

In IDN Guidelines version 4.0, which are guidelines which determine or guide how IDNs should be implemented at the second level—that's a separate discussion maybe for another day—but the latest version also suggests that there could be some script communities like Chinese, for example, where some variants should be automatically activated by the registry rather than asking the registrants. That's a script community decision. That's like an additional party to this process. Basically, IDN Guidelines also suggest all activated variant labels should be registered to the same registrant or same entity to prevent user confusion. So those are some of the details which obviously become relevant.

Another thing, which I guess this working group will be also considering, is harmonizing multiple IDN tables under a TLD. So I wanted to take some time to actually explain what that means as well so that eventual discussion can happen. So we've seen that an IDN table is offered under a TLD. We've also seen that a top-level domain can offer multiple IDN tables under it. So it can actually have 10 IDN tables which it offers based on its own business model. Basically, IDN Guidelines version 4.0 say that if there are multiple tables which are being offered under the same TLD, then those IDN tables should be harmonized. Also in the IDN variant TLD recommendations, it says that not only IDN tables under a TLD should be harmonized but all IDN tables under a TLD and its variant TLDs should also be harmonized. So it sort of extends what was done or what was suggested in the IDN Guidelines which were more applicable to a single TLD to TLD and all its variant TLDs as well.

So the question is what is harmonization? And for that we go back to With the IDN Guidelines version 4.0 which actually explains what harmonization is. So this is text which is actually quoted from IDN Guidelines version 4.0. It says that harmonization is two measures. These two measures are suggested to prevent cases of IDN variant labels being generated by different IDN tables under the same TLD to be allocated to different registrants.

So if one variant is being created by one table and another variant is being created by another table, both those variants are actually being created under the same TLD. So there must be a mechanism to actually manage that. Those two variants actually still go to the same registrant. That's what harmonization is actually aiming to achieve. So it's trying to plug this potential security issue from end user perspective.

There are two parts to it. Basically, two IDN variant code points or IDN variant code point sequences. One IDN table cannot be a non-IDN variant code point in another IDN table. So it's saying that if two code points are variants in one IDN table, they shouldn't be. So if a registry is offering both, let's say it's using Devanagari script and it's offering Hindi table at a Nepali table, both are written in Devanagari script. In the Hindi table, they say there are two characters which are variants, and in the Nepali table, they're saying those two characters are not variants, then that creates an inconsistency within that TLD. Of course, the first point says that that should not be the case.

The second says that all code points should be considered when determining variants under a TLD. I guess this second point gets more clear in the next slide where I have actually a concrete example. But let me stop here and go to Steve. Steve, please.

STEVE CHAN: Thanks very much, Sarmad. This is Steve from staff. And it's really just a practical question and a time check, actually. So we allocated about 60 minutes for this and we're actually coming up on time in four minutes. So just curious. I guess I just want to let you know. Thanks.

SARMAD HUSSAIN: So we'll pace up and finish in four minutes, hopefully. So this is an example of two IDN tables being offered. One is Arabic language table and one is Urdu language table. If you see, the third letter is slightly different in both. You can create two different labels, 0628, 064A, and 0641 in one case, and 0628, 06CC, and 0641 in the second case, but they look identical. If the two tables are offered independently of each other under the same TLD, then you will have these two labels dot TLD under the same TLD actually occurring and will go to two different registrants, and that can potentially create a security issue.

So the solution to that is that you need to harmonize those tables. There could be multiple ways of doing it, the way we've been doing it in the Root Zone LGR, for example, is that we create one large table which merges all the tables. When you merge all the tables, then you have 064A and 06CC both in the same table. And you say you identify that those two are actually variant labels. Once you define those as variant labels, then irrespective of whether you're creating one label from one table from another table, it identifies both those labels still as variants. And therefore, after harmonization, that potential issue actually is resolved, something which is probably a little more understandable for those who do not understand Arabic script.

So the previous example was within script example. But this is a cross-script example. So you could actually have a Latin table and Cyrillic table and you have different code points, C, E, I, P in English and then their corresponding Cyrillic letters. You can see that if those two tables are offered independently, it creates the string on the right which are identical looking, but they can be actually created under the same TLD and go to different registrants, and that potentially creates a security issue. But if you harmonize all those tables under the same merged IDN table, one way of managing it, and then define these variant code points, if you see at the bottom of the table, and then use this merge table to determine cross-script variants, for example, this allows to identify both those as variants, and then you could ensure that they all go to the same registrant or one blocks the other one so that the security issue is managed. So that's what harmonization is.

So moving on, we've already shared that we've actually developed reference LGRs. We are also in the process of creating a merged LGR for reference LGRs to support registry operators who look at that as well. The list is here, we are also in the process of developing more. These are based on community input on at least the script level. Work done by Root Zone LGR, we've actually gone back to the same Generation Panels, community experts, to give us feedback on second level LGRs. And then we also take them through public comment process. So we try to make sure that this has input from all the different stakeholders. Hadia, please.

HADIA ELMINIAWI: Thank you. Going back to your example, so you develop a merged IDN table or you keep them two separate IDN tables and add the code points from one language to the other, like the code points of one language to the table of the other and vice versa, and you still have two, right?

SARMAD HUSSAIN: We actually have three. So what we do is we have a Latin table, a Cyrillic table, and then a third merged table. At least for Root Zone LGR, that's the mechanism which we are following, that each language has its own table to determine its variants and validity of labels. But then we use the merged table to identify all the possible blocked variants.

HADIA ELMINIAWI: But you could also have only two with each of them.

SARMAD HUSSAIN: Right, exactly.

HADIA ELMINIAWI: It won't reflect much what's—

SARMAD HUSSAIN: Yes. That's exactly what I said when I was talking about merged table. I said this is one of the ways you could implement it. But there are actually other ways, and one of that is what you're suggesting that you actually in a way add these definitions back into the original two tables and extend them. So this can be implemented in multiple ways. This is one way of just implementing it. Thank you.

Just to close this off, generic, we talked about this, that a generic top-level domain registry operator which is intending to offer registrations in different languages and script. The gTLD must be approved to offer IDN service for languages and scripts, and that is done by them requesting for that so that it can be added as an IDN service in their contract.

Then they actually develop IDN tables and share these IDN tables with ICANN for review. Then once the review process is completed and the IDN tables are approved, these are updated in Exhibit A of the Registry Agreement. The way this can be done is at the application time or even after the application time when the gTLD is in operation, they can also add update, delete additional existing tables, or add additional tables. And that's done through an IDN service in RSEP process. And then there is also possibility of updating tables when a registry operator changes its RSP at the back end.

Finally, all these IDN tables are published in the IANA repositories so that other registry operators also can see what is being implemented. This provision was done so that one could develop a consistent solution across registry operators to the extent possible so that users can have a consistent user experience.

I think that brings me to the end. Well, last slide, we actually have an IDN table review tool as well, which is available for registry operators to use in case they want to see how we review the IDN table internally. They can actually see the review even before they submit it. So it's an effort we've done to make sure that the reviews we do for the registry operators are consistent and transparent. With that, back to you, Donna. Thank you.

DONNA AUSTIN: Thank you, Sarmad. I know that was a lot of information to get through but it is going to be important as we get through to our discussions. So we are a little bit behind schedule. I think we have Michael from the Registrars and Dennis from the Registries. They're going to share some practical experience with us on how the IDN tables work in practice.

But what I want to do, if everyone that's in the room or if you're at home, just stand up, have a stretch for 60 seconds, and then we'll come back to Michael and Dennis. Sixty minutes is a long time to be sitting down and I know we've got 30 more to go. So get up, stretch your legs, and we'll start this again in one minute. Thanks.

Writing in chat so there can't be standing and stretching. All right. So I think we will kick back in. Ariel … Dennis and Michael … Sorry I'm not sure how we plan to do this session so I guess we'll just throw it over to Michael and Dennis. Who wants to go first?

DENNIS TAN: Donna?

DONNA AUSTIN: Yes, Dennis?

DENNIS TAN: I'm here.

DONNA AUSTIN: Okay. Did you stretch, Dennis?

DENNIS TAN: I did. Yes.

DONNA AUSTIN: Excellent. All right. Okay. So we'll hand it over to you. Thanks, Dennis.

DENNIS TAN: All right. Is there a specific question that we need to react to the structure of the conversation, I guess?

DONNA AUSTIN: No particular question, Dennis. Just what we're hoping to get a sense of—given Sarmad's presentation, from a practical level, how does a registry actually implement the IDN tables? From your perspective, does it matter that there's different IDN tables from different registry operators? So just a little bit of a practical example experience of how as a registry operator, you operate IDN tables at the second level or register IDNs at the second level. So just practical information about how it works in practice, if that makes sense.

DENNIS TAN: Yeah. Thank you, Donna. I think that's a fair question. Let's see how it goes. I'm happy to follow up on that. For the record, Dennis Tan, Verisign. As a gTLD registry operator and you can see the portfolio of IDN tables that we offer, the .com, .net, and other TLDs that we support. There are north of 100 tables between languages and scripts. So throughout the years, the original Verisign IDN implementation predates me. I think it goes back 20 years ago. So it has been evolving throughout this time and there has been many iterations of IDN tables and reviews and based on the knowledge gained through community work and also our own research and investigation.

For example, we have certain languages, tables which require certain specific rules as it pertains to variants. For example, the Chinese language, that's one where we calculate variants of the simplified and traditional strings based on the work of—I'm trying to remember the exact name. Is it the JET Working Group or something along those lines? It was a very long time ago, they define a baseline table for Han script, so Verisign followed that implementation.

Since we're talking about variants, our policy is to block the registration for the variant. It just remains blocked for the registrant or any registrant. It was just a policy that's been in place for many, many years. But we're open to see whether that's still the case or we should revisit that policy.

I don't know if there's any questions that you specifically want me to answer but—

DONNA AUSTIN: Dennis, something that you and Edmon discussed in the chat earlier that there's quite a number of TLD operators now but there's not a lot of difference between IDN labels at the second level that the registry operators use. So does that reflect that there is that community consultation, or whether there's a sharing of IDN tables amongst registry operators? So how does that work?

DENNIS TAN: Thank you, Donna, for the point. So far as collaboration consistency across different gTLD operators and how they manage or define the IDN tables, yes. I think for the most part, as Edmon said earlier, and I think I agree that for the most part, IDN tables are consistent and very similar. I mean, there are more commonalities than there are differences. Difference might come just based on the sources that each registry operator is consulting during the development of the IDN tables. And in my DBA from perhaps a few code points, meaning letters or potentially some rules as they apply, and maybe that's something as a data point that this working group might want to dig into. Not so sure how we would structure such research but at least from numbers of tables that are shared by registry operators. I mean, the IANA repository is something to look at. So yeah. Back to you, Donna.

DONNA AUSTIN: Satish has a question. And then I have one that I'll read out from Justine. Satish?

SATISH BABU: Thanks, Donna. I'd like to know if there is any formal interface between registry operators and language communities, because this is ultimately a language issue. So what is the process of consultation? Thank you.

DENNIS TAN: Thank you. I'll take a stab, and then others, please feel free to chime in. This is going to sound repetitive, but the implementation of our tables predates me and I'm already 10 years in Verisign. So it’s been a long time ago. So the consultation existed but it was a very long time. Trying to remember we did some of it.

I think as far as when we launched our new gTLDs, the Korean and the Japanese TLDs, we did some outreach to these communities and see what their revision might be in place. Maybe we did but it was not as extensive. It’s not the creating the table from the ground up but it was an adjustment.

But again, it continued to evolve. For example, we went through the registry testing through the New gTLD Program. That process informed us of changes that we adjusted in our own table. So that’s one of the examples that those tables get feedback and get developed into the new implementation and new registrations are applied to those new rules.

DONNA AUSTIN: Thanks, Dennis. The question from Justin in chat is how well does the discussion between registry operator and ICANN Org regarding the security and stability requirements of the of the registry operator’s IDN tables? I’m not sure I’ve said that correctly. But I guess what’s the discussion between the two in relation to security and stability requirements for IDN tables?

DENNIS TAN: Thank you, Justine, for the question. I think as far as security and stability, that's going back to my remarks about IDN tables should not have any secure and stable issues. There is an ongoing conversation and I'm just looking at Sarmad here. We were just talking earlier as a follow-up to an ongoing conversation between the ICANN Org and the registry operators as far as how these IDN table review processes are going. How can we improve the transparency and consistency of those reviews, and when it comes to security and stability issues, what exactly are we measuring against? As Sarmad pointed out, those definitions, the rules are those recommendations or advice or whatever you want to call them that come from the specific script communities that ICANN, we as a whole, have gained through the work of the Root Zone LGR project and the Generation Panels that they have gone through many, many cycles, working through many years, establishing those script rules. So looking through those lenses, we can get a glimpse of what secure and stable means. So there is no one single definition. It basically is based on the scripts. So hopefully that answers your question, Justine.

DONNA AUSTIN: Oh, Rick, you can't hear me because I'm on mute. Rick, go ahead, please.

RICK WILHELM: Thank you, Donna. Rick Wilhelm, PIR for the transcript. Just a brief statement to follow up on something that Dennis Tan said regarding the security and stability. The thing I would add is that its security and stability, as many folks apprehend, is a dynamic property. It can be secure and stable at one point, and then new things can emerge that cause something that was formally thought to be secure and stable to be found to be then no longer secure and stable. So that's something to just keep in mind. Because sometimes there are security stability things that are literally discovered related to IDNs that were previously thought to be secure and stable. Thank you.

DONNA AUSTIN: Thanks, Rick. Michael, I think you're with us. I assume you're in the room with everybody else. Okay. So can I hand it over to you? I guess what will be interesting to understand is obviously it seems the registry operator has the IDN tables where registrars over names at the second level, what's the applicability of the IDN tables, if you've got multiple registries with different IDN tables? So how does that work from a registrar perspective?

MICHAEL BAULAND: Thanks, Donna. I cannot speak for all of us, of course, but in our registrar software, we actually do not implement any IDN table because it does not really make sense. Whenever a domain is applied for or we do a domain check, we just send that domain label to the registry. And then the registry software will tell us whether that's valid label or not. And in that sense, the registry is always authoritative regarding the IDN table. So even if we implemented the same IDN tables to do a check beforehand, the best case, it would say the same thing as the registry so there's no use. And worst case, there was some error in the implementation of the IDN table, especially if the rules Sarmad showed in the old styles were just given in English language, there's a big chance that the implementations would be different from the registry ones. So from my perspective, it makes no sense for the registrar to do any work in the implementation of those IDN tables. Thanks.

DONNA AUSTIN: Thanks, Michael. Is what you're saying is that the registrar doesn't develop its own IDN tables but it has the ability to do a technical check with the registry to ensure that whatever the name is is consistent with their IDN tables before the sale goes ahead, I suppose.

MICHAEL BAULAND: Yes. Basically, that's the case. The EPP protocol allows for a so-called domain check, which, on the one hand, tells you whether the domain name is still available for registration, or whether it's taken or maybe blocked due to some variant relationship. This is also telling you whether the label itself is valid, or whether you used some invalid code points that are not supported by the registry, or whether you combined code points in a way that does not support it. So yeah. If ever a customer of ours wants to register a domain name, we send that domain name to the registry via EPP domain check command. That is the result we then hand back to the customer. Thanks.

DONNA AUSTIN: Terrific. Thanks, Michael. So any questions from folks? I know that there were a few early on in the piece that Sarmad suggested it might be best for this session. So I don't know if those questions have been satisfied or not. So if anyone has questions for our Registry and Registrar colleagues, now would be a good time to ask them. Or if any of our other Registry or Registrar colleagues wanted to share a little more about their experience, that would be great, too. Michael?

MICHAEL BAULAND: Thanks. While my registrar perspective is probably quite boring because we didn't do anything, our company is also providing the registry backend service for a few of the new gTLDs. In that context, we also had to deal with the question what IDN tables our customers want to offer and how we develop those IDN tables.

For this, we mainly just looked at the existing IDN tables in the IANA repository of TLDs that existed before the 2012 round. We then contacted some of those registries and asked them whether it would be okay to use their IDN table. One advantage is of course we don't have to put work into development. And second is that then our TLDs are using IDN table that's already known to other registrars so they're the same, which makes life easier.

So most of the tables we used from existing ones, there was just one IDN table which is for the Arabic script TLD. For that, the customer wanted to have an Arabic script IDN table covering many of the languages using this Arabic script. And for this, we actually developed our own table. The main work has been done by Siavash Shahshahani and Alireza. I also supported them but just in computer science way, so to say, not in linguistic way because I don't know any Arabic. But I was able to make sure the rules they wanted to implement were consistent in themselves and not contradictory. So in that context, we developed our own IDN table and then used that for the Arabic script TLD. Thanks.

DONNA AUSTIN: Thanks, Michael. It's an important perspective to have from a registry backend service provider. So I really appreciate you providing that perspective as well. So thank you for that.

So we're three minutes from time and I don't see any more hands. So I guess you can have another couple of minutes for another stretch before you move off to your next meeting or whatever it is you're going to.

Just a reminder to our EPDP team that we won't be meeting next week, but we will kick off again the week after that. So safe travels home, everybody. Sorry, we couldn't see you in person. But I assume Justine will be okay because she lives in Kale. So maybe we'll see you at ICANN75. So thanks, everybody.

JULIE BISLAND: Thank you, Donna. Thanks, everyone, for joining. This meeting is adjourned. And you can end the recording.

**[END OF TRANSCRIPTION]**