26-Feb-2015

Dear ICANN,

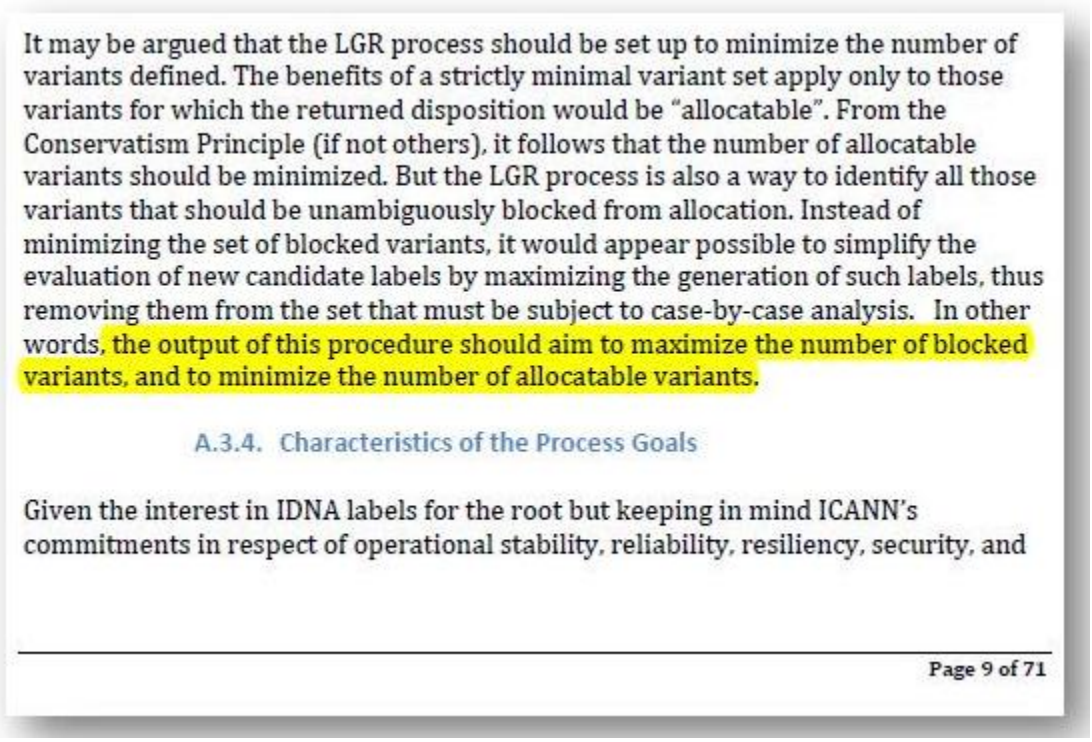
Please find attached our comments pertaining the announcement for public comments: "IDN TLDs - LGR Procedure Implementation - Guidelines for Designing Script-Specific Label Generation Rules (LGR)"

Sincerely Yours,
Raed Al-Fayez
Abdulrahman Al-Ghadir
Abdulaziz Al-Zoman

Comments on

IDN TLDs - LGR Procedure Implementation - Guidelines for Designing Script-Specific Label Generation Rules (LGR)

We DO entirely agree with the statement that was outlined on the document entitled: "Procedure to Develop and Maintain Label Generation Rules (LGR) for the Root Zone With Respect to IDN Labels" (page 9, highlighted in yellow in the following screen capture):



It may be argued that the LGR process should be set up to minimize the number of variants defined. The benefits of a strictly minimal variant set apply only to those variants for which the returned disposition would be "allocatable". From the Conservatism Principle (if not others), it follows that the number of allocatable variants should be minimized. But the LGR process is also a way to identify all those variants that should be unambiguously blocked from allocation. Instead of minimizing the set of blocked variants, it would appear possible to simplify the evaluation of new candidate labels by maximizing the generation of such labels, thus removing them from the set that must be subject to case-by-case analysis. In other words, the output of this procedure should aim to maximize the number of blocked variants, and to minimize the number of allocatable variants.

A.3.4. Characteristics of the Process Goals

Given the interest in IDNA labels for the root but keeping in mind ICANN's commitments in respect of operational stability, reliability, resiliency, security, and

Page 9 of 71

Based on our extensive experiences in the field of Arabic IDNs that have been accumulated since 2001, and base on our continuous efforts to find and reach an acceptable, realistic and workable solution (for all relevant entities: registries, registrants, and normal Internet users) to the hassle of managing and using enormous variant domains that might be generated due to character similarities within the whole Arabic script, we have indeed developed and implemented a Variants Management System that explicitly and constructively enforces the above goal. It does that by adopting the following concepts:

I. One Key for all Variants (Master key)

An Arabic domain name label may have many variants. For example, if a registrant wants to register a domain name label that contains more than 4 characters each of them has its own variants, then the required domain name may end up with hundreds or thousands of possible variants (e.g. كيكة has more than 200 variants, منطقة-مكة-مكرمة has more than 3,000 variants!). Therefore, storing all possible variants is not a visible nor a practical solution, especially for longer domain names as they generate larger variant list. Thus, a new identification mechanism has been developed and used to easily manage the whole variant list with one unique identifier, to speed up the lookup process, and to eliminate the need of saving all possible variants (hence save storage space). Our Variants Management System achieved that by adopting the "Master Key Algorithm" that we developed to generate a unique key for a domain name label and all of its possible variants, which then can be used in the lookup process for both domain name availability and variant allocation.

II. Variants base on character position

In Arabic script languages, characters may take different shapes depending on their position (standalone, beginning, middle, or end) within a word. Therefore, our Variants Management System considers a character position when deciding that two code points are variants or not. For example, let us consider the following variant table for the HEH Class:

Code Point	Possible shapes in context [Standalone, End, Middle, Beginning]			
0647	ه	ـه	ـهـ	هـ
06BE	ھ	ـھ	ـھـ	ھـ
06C1	ہ	ـہ	ـہـ	ہـ

The following is a list of all the permutations (total 16) of the Arabic word (هدهد) (meaning in English the bird Hoopoe):

هدهد	هدهد	ہدهد	ہدهد
هدهد	هدہد	هدہد	هدهد
هدہد	هدہد	ہدهد	ہدہد
ہدہد	ہدهد	ہدہد	ہدہد

If we were not considering character position when generating variants we will get a list of (16) variants. If we consider character position when generating variants we will get only 4 valid variants (i.e. 25%). So the other invalid 12 (75%) words will NOT be considered variants as they are totally different and do not present risk or any security issue.

III. A label is composed using a single input character set table

From practical and realistic point of view, it is safe to assume that a string (label) in Arabic script based language is typed using "one" keyboard layout (input device); i.e., there are no mixing between code points from different keyboards (Arabic Keyboard layout , Urdu keyboard layout ..etc). For example, when typing the word (کلی), the possible valid ways that can be typed subject to the selected input device (keyboard) will be the following (total 3):

LANGUAGE	UNICODE	LABEL
Arabic	(U+0643) (U+0644) (U+0649)	كلى
Persian/Urdu	(U+06A9) (U+0644) (U+06CC)	کلی

Arabic	(U+0643) (U+0644) (U+064A)	كلي

Other possible combinations such as the following:

(U+0643) (U+0644) (U+06CC)	كلی
(U+06A9) (U+0644) (U+0649)	کلی
(U+0643) (U+0644) (U+06D2)	كلے

are not realistic nor practical, as each word is composed of characters that are not available in one input device (i.e., you need more than one input device to be able to compose the whole word). Therefore, out of 18 total possible variants for the word (كلى), only 3 are allocate-able; the rest (which represents %83) are blocked, see Appendix I.

To further illustrate the strength and efficiency of our Variants Management System in significantly minimizing the number of allocate-able variants and maximizing the number of blocked variants, please consider the following example:

- Domain label: منطقة-مكة-مكرمة
- Total number of possible variants: 3,888
- Possible number of valid input (subject to the input keyboard): 4
- List of Possible valid variants (subject to input keyboards):
  - ✓ منطقة-مكة-مكرمة (Arabic)
  - ✓ منطقة-مکة-مکرمة (Persian)
  - ✓ منطقۃ-مکۃ-مکرمۃ (Urdu)
  - ✓ منطقه-مكه-مكرمه (Arabic)

It is clear from this example that our system noticeably cuts down the number of allocate-able variants and increases the number of blocked variants. Hence, it definitely and positively achieves the abovementioned goal of the "Procedure to Develop and Maintain LGR".

IV. Step-by-step adoption

The Arabic script serves many languages (50+ languages). Most of them are either: historical languages (not used any more) , not mature from linguistic and technical point of view (e.g. no electronic existence) or have changed their writing script to Latin. Our Variants Management System handles this issue straightforwardly by giving the mature languages (which have their language tables and variant tables already defined) a quick start with protection to the registry and registrants. Later, any other language becomes ready (by having their language and variant tables being defined) it can be easily added to our system without the need to regenerate the keys for the registered domain names.

V. Study variants across the whole Arabic script

When building any variant table a full study should be conducted across the whole Arabic Script in order to identify all possible variants against code points in the supported language table (not like other solutions: who only check the variants between code points within only

the support language tables). This way whenever a new language is added there will be no need to restudy the previous supported languages and change their variant tables. The result is less key regeneration when adding new languages to our system.

VI. Variants types

Variants are tagged in our system based on different categories or types. This makes the process of identifying them easy and later it assists selecting the right action that could be done on them based on their types. The supported types are:

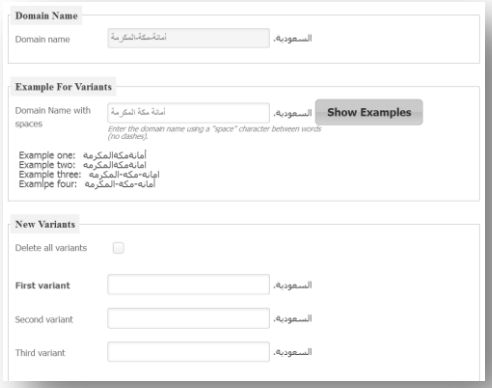
- Exact: The similarity between the concerned characters is visually identical (as mirror).
- Typo: The concerned characters are look-alike but not identical (typo/style match).
- Interchangeable: The concerned characters can be used interchangeably by many users (e.g. In the Arabic language, at the end of words, ARABIC LETTER TEH MARBUTA (U+0629) and ARABIC LETTER HEH (U+0647) are used interchangeably in writing. That is because they sound similar when pronounced at the end of phrase, and hence the LETTER TEH MARBUTA sometimes is written as LETTER HEH and the two are considered "confusable" in that context. See RFC 6365.).

VII. International Reachability

One of the main principles for the stability of the Internet and Internationalized domain names that the end user should be able to reach his/her domain name regardless of location. In order to inforce this principle the input devices (language table) that the user may use to reach a domain name (based on the user location) should be carefully considered when defining variants. Consider, for instance, the case where a suitable variant is not allocated to the registrant this may cause a reachability problem and reduce the user acceptance. For example, if someone registered the domain name "مكة" (all characters from the Arabic language) and a user try to reach that domain name from an Internet café in Pakistan, he/she will not be able to reach it unless this variant "مکة" (Urdu variant) is allocated and delegated. In summary, variants need to be studied from both similarity point of view (by language community) and reachability pointy of view (based on input devices used by other language communities). We believe that variants which are generated by the latter should be automatically allocated to the registrant since they are needed for domain name stability and reachability.

VIII. Simple User Interface

A registrant should not be shocked by the complexity of the interface and the huge size of variant list to choose which variants to allocate. The registry should not assume that regular user may know the differences between Arabic KAF (U+0643) and KEHEH (U+06A9) just by displaying the different variant labels. Also, it is unpractical to list all allocate-able variants because the list may contain hundreds of allocate-able variants. Thus, the registry should help the registrant to generate and distinguish some variants (as helping examples) and then the registrant may choose from them or manually type the desired allocate-able variants. For example, the Registry could provide a separate web interface (and/or EPP command) for listing the possible variants in a clever way (i.e. using multiple filters to minimize the generated list) to help both the Registrar and Registrant generating and managing variants. The following is a snapshot of our Variant Management System (old version):



To test our solution please visit and use the following online tools. It also generates the complete variant table for the considered languages (Arabic, Persian, Urdu): http://arabic-domains.org/adn_tools/mk/index.php

The technical description of the Master Key algorithm that are used in our proposal can be found in the following links:

- http://arabic-domains.org/docs/Master_Key_Algorithm.pdf
- http://nic.sa/en/view/doc64

Please note, the above “public” tools and documentations were developed and written some time ago (2007-2010), therefore some common terminologies are not yet developed at that time so we use our own terminologies that you may have difficult to grasp from the first reading. SaudiNIC is continuously modifying these tools and incorporating within them all the new developed concepts and ideas but they are kept for internal usage until it reach acceptable and stable stage when they become ready to publish for the public. Hence, SaudiNIC in the process of customizing and polishing these tools to put them for public use as well as updating the related documents to use the new terminologies and to support the proposed format outlined in the “Representing registration policy for IDNs using XML” that will be used in IANA Repository.

In conclusion, we DO understand and appreciate the goal and the statement outlined in the document and we have proposed and implemented a solution to achieve this remarkable goal in a straightforward manner that we are happy to share with others without any restrictions whatsoever.

Appendix I (Variant List for "کلی"):

Variant size: 18

Exact(s) from same language (2)

LANGUAGE	UNICODE	LABEL
Persian, Urdu	(U+06A9) (U+0644) (U+06CC)	کلی
Arabic	(U+0643) (U+0644) (U+0649)	کلی

Typo(s) from same language (2)

LANGUAGE	UNICODE	LABEL
Arabic	(U+0643) (U+0644) (U+064A)	كلي
Urdu	(U+06A9) (U+0644) (U+06D2)	کلے

Exact(s) from different language(s)(Mixed) (2)

UNICODE	LABEL
(U+06A9) (U+0644) (U+0649)	کلی
(U+0643) (U+0644) (U+06CC)	كلی

Typo(s) from different language(s)(Mixed) (12)

UNICODE	LABEL
(U+06AA) (U+0644) (U+0649)	ڪلی
(U+06A9) (U+0644) (U+064A)	کلي
(U+06AA) (U+0644) (U+064A)	ڪلي
(U+06AA) (U+0644) (U+06CC)	ڪلی
(U+0643) (U+0644) (U+06CD)	كلۍ
(U+06A9) (U+0644) (U+06CD)	کلۍ
(U+06AA) (U+0644) (U+06CD)	ڪلۍ
(U+0643) (U+0644) (U+06D0)	كلې
(U+06A9) (U+0644) (U+06D0)	کلې
(U+06AA) (U+0644) (U+06D0)	ڪلې
(U+0643) (U+0644) (U+06D2)	كلے
(U+06AA) (U+0644) (U+06D2)	ڪلے