



Internationalized Domain Names Variant Issues Project
Arabic Case Study Team Issues Report

L2/11-367



Internationalized Domain Names Variant Issues Project
Arabic Case Study Team Issues Report



Arabic Variant TLD Issues and Requirements

1. Background

This document identifies issues relevant to definition, management and implementation of variants at the root level. The scope of variants in the context of the Arabic Script Case Study is across the strings that, when registered as TLDs, would cause confusion that may result in:

- Unexpected end-user experience
- DNS security and stability issues

In 2009, an independent implementation working team was formed after discussions during the ICANN meetings in Mexico City and Sydney to study these issues. The team included linguistic and technical experts from various language communities, and was co-chaired by two members of ICANN Board of Directors, who are well-versed in the fields of IDN and DNS. The team recommended that variants of TLDs not be delegated at that time, and that if desired variants are to be delegated, certain conditions must be fulfilled.

To develop potential solutions for the delegation of IDN variant TLDs, the ICANN Board in its 2010 meeting¹ in Norway directed the CEO to:

... develop (in consultation with the ICANN Board ES-WG) an issues report identifying what needs to be done with the evaluation, possible delegation, allocation and operation of IDN gTLDs containing variant characters, as part of the new gTLD process in order to facilitate the development of workable approaches to the deployment of gTLDs containing variant characters IDNs. The analysis of needed work should identify the appropriate venues (e.g., ICANN, IETF, language community, etc.) for pursuing the necessary work. The report should be published for public review.

Accordingly, ICANN in consultation with community has proposed to conduct as many as six case studies -- on the Arabic, Chinese (Traditional and Simplified), Cyrillic, Devanagari, Greek, and Latin scripts -- to investigate the set of issues that need to be resolved to facilitate a good user experience for IDN variant TLDs. Each Case Study Team has been requested to submit an Issues Report for the script it is looking into. From these Issues Reports, a single Issues Report will be created.

This document presents the Issues Report developed by the Arabic Case Study Team, and is intended for the TLD level².

¹ See ICANN Board of Directors. (2010) Adopted Board Resolutions. Trondheim, Norway. Retrieved November 30, 2010, from <http://www.icann.org/en/minutes/resolutions-25sep10-en.htm#2.5>

² Though some issues may apply to second and other levels, that discussion is not in scope of the current work.

2. Introduction to Arabic Script

Arabic script has been used across North Africa, Middle East, Central Asia, South Asia and South East Asia to write multiple languages from Semitic, Indo-Iranian, Indo-European, Dravidian, Turkic, African and Austronesian language families. Some salient areas in which Arabic script is currently being used are highlighted in the illustration below.



Figure 1: Writing Systems of the World

(Source: <http://en.wikipedia.org/wiki/File:WritingSystemsoftheWorld4.png>)

The Arabic writing system is also referred to as an Abjad system, in which consonantal sounds are represented as base characters and vowels are normally represented by optional combining marks on these base characters (except for long vowels, which may also be represented by base characters). The writing system is cursive and each letter may have multiple shapes, generally categorized as initial, medial, final and isolated forms, based on where it occurs within the connected portion of a (sub)word, called a ligature, or whether it occurs by itself (not joined with any other letter). Though logically a letter may have these four shapes, in reality the shape of a letter may also vary with other letters it joins with (not just its position within a ligature) and may take up many different shapes³.

An additional complexity in the Arabic script is the bi-directionality of the script. The script is generally written from right to left; however, the digits are written from left to right. Though this does not change the key-press order of input into computing devices, it may have significant effects on how the input is displayed. Further complexity is introduced if the (right to left) Arabic script is mixed with other (left to right) scripts, like Latin.

³ For an illustration of context dependent shaping see http://www.cle.org.pk/Publication/papers/2006/context_sensitive_shape_substitution.pdf

Arabic script has a variety of writing styles, with Naskh generally used to write Arabic, Sindhi and Persian languages and Nastaliq being used for many of the other languages, including Urdu, Pashto, Kashmiri etc. Additional writing styles, which are used for stylistic reasons, include Thuluth, Diwani, Kufi, Riqua, etc. Some of these writing styles are illustrated in Figure 2 (with same Arabic language phrase written in different styles). The various styles show both the base characters and the optional combining marks. Thuluth and Diwani also include non-linguistic ornate marks.

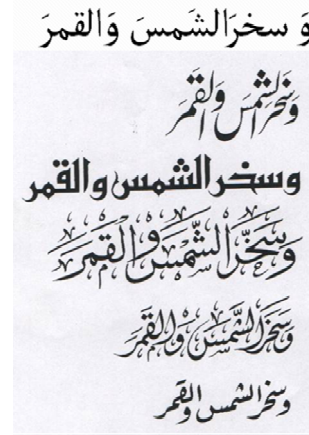


Figure 2: Arabic Script Writing Styles: Naskh, Nastaliq, Kufi, Thuluth, Diwani and Riqua

3. General Principles about the Report

During the discussion, the Arabic Script Case Study Team deliberated on many issues and has agreed on the following three general principles for interpreting the eventual document.

As per the scope of the study, the Arabic Script Case Study Team agreed to talk generally about TLDs, without making the distinction between ccTLD or gTLD labels at the root.

The team also agreed to limit the scope of work to TLDs (not second or other level labels), even though some of the issues raised may apply to all levels.

Finally, though the team is generally confident on the identification of the issues and any corresponding recommendations, some issues may still need to be discussed with representatives of languages communities not represented in the committee (e.g. use of Arabic script in African languages). Such issues have been explicitly identified for further consultation.

4. Terminology

Specific terminology has been used in this document, as presented in Appendix E. These terms are based on the initial set of definitions circulated by the IDN Variant Issues Project Team of ICANN, but have been tailored for the Arabic Script Case Study, where necessary. Any relevant but missing terms have also been added to the list. It is recommended that the readers familiarize themselves with these terms, as defined, before reading the current document any further.

5. TLD Label Valid Code Points for Arabic Script

The Arabic script characters are encoded from U+0600 – U+06FF and U+0750 – U+077F⁴ in the Unicode standard. Only a subset of these characters are PVALID, i.e. allowed for use in labels

⁴ Unicode standard also include Arabic Presentation forms in the ranges U+FB50 – U+FDFF and U+FE70 – U+FEFF, which are not recommended for use and are not PVALID.



(e.g. see RFC 5892). The team suggests further limiting the use of these characters for TLDs, as per the following details.

1. U+0610 - U+061A: an issue as they are PVALID but should not be allowed for TLDs as these are combining marks
2. U+0620- U+063F: OK, PVALID and needed for TLDs
3. U+0641- U+064A: OK, PVALID and needed for TLDs
4. U+064B - U+0659: an issue as they are PVALID but should not be allowed for TLDs as these are combining marks
5. U+065A - U+065F: an issue as they are PVALID but may not be allowed for TLDs as these are combining marks
6. U+0660 - U+0669: an issue as they are permitted under a context rule, but should not be allowed for TLDs, because digits are not allowed in TLDs
7. U+066E - U+066F: an issue as they are PVALID but should not be allowed for TLDs because Archaic and not used in Arabic script based languages now
8. U+0670: an issue as it is PVALID but should not be allowed for TLDs as it is a combining mark
9. U+0671 - U+0673: OK, PVALID and needed for TLDs
10. U+0674: an issue as it is PVALID but resembles a combining mark
11. U+0679 - U+06D3: OK, PVALID and needed for TLDs
12. U+06D5: OK, PVALID and needed for TLDs
13. U+06D6 - U+06DC: an issue as they are PVALID but should not be allowed for TLDs as they are Quranic marks which are not used in writing contemporary Arabic script based languages and are combining marks
14. U+06DF - U+06E8: an issue as they are PVALID but should not be allowed for TLDs as they are Quranic marks which are not used in writing contemporary Arabic script based languages and are combining marks
15. U+06EA - U+06ED: an issue as they are PVALID but should not be allowed for TLDs as they are Quranic marks which are not used in writing contemporary Arabic script based languages and are combining marks
16. U+06EE - U+06EF: OK, PVALID and needed for TLDs



17. U+06F0 - U+06F9: an issue as they are permitted under a context rule, but should not be allowed for TLDs because digits are not allowed in TLDs
18. U+06FA - U+06FF: OK, PVALID and needed for TLDs
19. U+0750 - U+077F: OK, PVALID and needed for TLDs
20. U+FE73: an issue as it is PVALID but should not be allowed in any label (TLDs and other labels) as it is not a letter or character used in writing Arabic script based languages
21. U+200C (ZWNJ): Zero-Width-Non-Joiner (ZWNJ) is code point needed in Arabic script to allow for the final shape of a dual-joining letter to appear in the middle of a word for correct orthographic representation of some words in some languages, e.g. when, in languages like Farsi and Urdu, a suffix of a word does not join with the root, as in فيض آباد (instead of فيضآباد, “Faiz-Abad” in Urdu and Farsi). The team discussed both the need to allow ZWNJ and issues that may arise by using it in TLDs in great detail. The following is a summary of the arguments which came out of the discussion:

- a. Arguments in favor of allowing ZWNJ for TLDs

It is needed by some languages (including Farsi, Urdu, Kurdish, etc.) to correctly render a label where a non-final dual-joining letter occurs in the final form (making the word to appear disjoint, where it would be joined otherwise).

- i) Linguistically for some languages (e.g., Farsi, Urdu) this can occur in multiple cases:

a prefix which is disjoint with the root; a root which is disjoint with the suffix; arbitrary disjoint word due to orthographic convention (many times this occurs for borrowed words from English and other languages); transliteration of abbreviations like UN; one word followed by another word, where space is not allowed to separate these words (e.g. in labels for Urdu, Farsi, Kurdish, Pashto, etc.)⁵.

- ii) The ZWNJ is visible in most cases due to the significant change in letter shaping (see examples of exceptions listed later in this discussion; a thorough analysis of the Arabic script charts for Unicode is needed to determine the complete set)

⁵ Hyphen is used for Arabic language community for this purpose in the domain names. See Point 23 in this section.

- iii) As discussed, this character is needed for many languages, including Farsi, Urdu, Kurdish, etc. It is already in use by some language communities, e.g. Kurdish⁶.
- iv) It is being proposed by various language communities to be used and added to their keyboards to write IDN labels, similar to adding '@' sign to write email addresses. For e.g. national consensus and policy for .پاکستان IDN ccTLD requires ZWNJ to be added and available for use in the labels⁷. It is available on some other keyboards⁸
- v) It is required to render the brand names properly, e.g. پیپسی کولا (“Pepsi Cola”) for many languages using Arabic script, including Urdu, Pashto, Farsi, Kurdish, etc.

b. Arguments against allowing ZWNJ for TLDs

- i) Labels need not capture linguistic conventions and may be treated as a string instead of a linguistic word, which undermines word-based arguments.
- ii) At Script level, the ZWNJ is considered by UNICODE to be an invisible join control character and listed in the "Unicode Security Considerations" document, which warns that incorrect usage can expose programs or systems to possible security attacks. This is especially relevant for IDNs.
- iii) The ZWNJ in a few cases is still not visible to all users (e.g., U+0637, U+0638, U+069F, U+06BE, and U+06FF). A comprehensive analysis of Unicode Arabic Script Code Charts is needed to find any additional cases. This process should be repeated as the Unicode gets updated.
- iv) The ZWNJ concept and behavior are not known to many Arabic script users, who do not use it or know how to type it.
- v) ZWNJ is not conveniently available on the keyboard, where typing it requires multiple simultaneous key-presses, which is complicated for users⁹. ZWNJ is also inconsistently placed on keyboards across various operating systems. In addition, it is not available on many keyboards, making it difficult for people to use ZWNJ when, for example, they travel.

⁶ E.g. in Kurdish there are 7 ZWNJ occurrences in 10 words in the first sentence at:
http://ckb.wikipedia.org/wiki/%D8%B2%D9%85%D8%A7%D9%86%DB%8C_%DA%A9%D9%88%D8%B1%D8%AF%DB%8C

⁷ See <http://www.cle.org.pk/IDN/IDN2009/download/minutesofsecondworkshop.pdf> Section 5 (b).

⁸ In Windows 7, activate the "Arabic (Saudi Arabia)" keyboard and try Ctrl+Shift+2. This is the same method people should type in using Microsoft's Persian keyboard layout

⁹ ZWNJ may be generated using three key strokes pressed simultaneously and with different combination depending on the operating system. Regardless, these combinations are not known to many users.

- vi) The users may not be able to type a domain name as they may think it is a <space> not ZWNJ, which may lead to reachability and usability problems, and therefore, mistrust of IDNs.
- vii) Based on the ZWNJ Contextual Rule (RFC 5892 Appendix A.1) for handling CONTEXTJ labels under the current IDNA2008, the Rule implementation¹⁰ does not totally resolve the non-visibility problem particularly in some cases as discussed above (e.g., U+0637, U+0638, U+069F, U+06BE, and U+06FF).
- viii) It is not one with the general category of {Ll, Ll, Lm, Mn}, as per the requirement defined by the gTLD Applicant Guidebook (v 2011-09-19, Module 2, page 2-13, Section 2.2.1.3.2, Part II, Item 2.1.3.).
- ix) Root policy should be more conservative than labels for other levels.
- x) Use of ZWNJ may cause additional bidirectional display issues.
- xi) Hyphen can be used instead of ZWNJ to break a string into ligatures¹¹.

c. Should ZWNJ be allowed at the TLD level?

The team has not been able to reach a consensus on this due to the different arguments, presented above. As this is an important decision, the team recommends that the ZWNJ issue should be further investigated through a comprehensive study. Following are a few questions to stimulate later the discussions that will be part of the comprehensive study¹². The right balance is needed, where variety in Label Generation Policy is desired without compromising the security and stability of the DNS.

- i) Shall ZWNJ be allowed provided that the string with it is considered a variant of the string without it?
- ii) Shall ZWNJ be allowed provided that whenever ZWNJ is allocated, then the variant without it must also be allocated?
- iii) Shall ZWNJ be allowed provided that whenever ZWNJ is allocated, then the variant without it must also be allocated, with the condition that the

¹⁰ Here are 2 examples of IDNA2008 implementations that depict the problem: two different A-labels generated from U-Labels (one of them has invisible ZWNJ):

<http://unicode.org/cldr/utility/idna.jsp?a=%D8%B7%D8%A8%D9%84%0D%0A%D8%B7%E2%80%8C%D8%A8%D9%84>

<http://demo.icu-project.org/icu-bin/idnbrowser?t=%D8%B7%E2%80%8C%D8%A8%D9%84>

¹¹ Hyphen has been agreed to be used by the Arabic language community as word separator as <space> is not allowed to be used in a label (see RFC 5564).

¹² These questions are neither exhaustive nor may present the best options. However, they are being documented to facilitate a more rigorous and comprehensive analysis later.

label with ZWNJ cannot be a fundamental label, but can only be a variant?

- iv) Shall ZWNJ be allowed only if the applicant can demonstrate convincingly that the string will not cause security risks, with additional restrictions as discussed above?
- v) Shall ZWNJ be not allowed in a TLD label?

d. Additional restrictions

Though there is a defined rule which allows ZWNJ only in contexts where its effect is visible, there are few contexts which ZWNJ may still not have a visible impact.

- i) This includes characters U+0637, U+0638 and U+069F. This is indicated by the two sequences, one with and one without the ZWNJ: ط ب (U+0637, U+0628 and U+0637, U+200C, U+0628 respectively; also see Appendix D). The ZWNJ should not be permitted following such characters, in addition to the constraint already put on its use by the IDNA 2008 protocol (see RFC 5893).
- ii) Similarly, the use of ZWNJ after HEH (e.g. HEH group in Appendix A.1; also see Appendix D) should also be carefully analyzed to restrict ZWNJ from occurring in cases where the shape change due to its occurrence is not visible. Careful analysis is also needed as sometimes the shape of HEH group rendering is also not correctly done in fonts (as confusion should be based on the canonical glyph shapes recommended by the Unicode standard).

22. U+200D (ZWJ): Not needed in Arabic script for TLDs

23. U+002D: Hyphen has been agreed to be used by the Arabic language community as word separator as <space> is not allowed to be used in a label (see RFC 5564). It is currently not allowed in TLDs (see RFC 1123).

A general rule may be extracted that combining marks should not be allowed for TLDs. Such a rule will help in reducing security threats in TLDs. However, before finalizing such a policy, consequences for African and other languages, which use these marks (especially U+065A - U+065F), should also be considered. In addition, the team did not have time to perform an exhaustive analysis of Unicode's Canonical Equivalence for any effect that depends on any of these characters. If a character in NFC form ends up depending on one of the combining marks, then it might be that the code point needs to be permitted in some circumstances. However, if a small subset of marks is still needed, it would still be better than allowing for all the marks. This will also significantly reduce the number of variants that can be generated.

This list may get updated as Unicode standard changes and more characters are added to Arabic script, as per the IDNA 2008 protocol specifications.

6. Character Variants in Arabic Script

Variant labels in Arabic script may occur due to reasons motivated by linguistics, writing styles and/or encoding. The current issues document lists all possible cases where variants may occur, for consideration and further stipulation of how variant sets may be constituted. The current analysis distributes character variants into four categories: identical, confusingly similar, interchangeable and optional.

1. Identical Cases

In these groups of letters, a letter exhibits an identical shape in at least one of the initial, medial, final or isolated forms with at least one other letter in the group. The groups are formed transitively, i.e. if A is a character variant of B, and B is a character variant of C, then A is also considered a character variant of C, even if the condition of being identical is not met between A and C (e.g. see the HEH set). These groups are defined in Appendix A.1.

- a. KAF group – limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).
- b. HEH group – limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).
- c. YEH group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).
- d. BEEH/E group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).
- e. FEH group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).
- f. VEH group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).
- g. TEH MARBUTA group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).
- h. HEH with HAMZA group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).
- i. TEH/RNOON group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).
- j. NOON group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).
- k. NOON/PEH group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request).

In addition to those which have exactly the same shape, the identical character variants may also be caused by a letter combining with a combining mark. Such cases are given in Appendix A.2 (A.2.1 and A.2.2). Some of these cases are addressed automatically through the IDNA 2008 protocol specifications, which require the characters to be normalized (in Normalized Form C) and treats them using Unicode's definition of Canonical Equivalence. However, it appears that in some cases pre-composed characters do not have Canonical Equivalence defined to a series of code points that produces the same or similar apparent (abstract) character. Cases where such Canonical Equivalence has not been defined by Unicode (for various reasons) would need to be explicitly

managed as variants. See the status listed in the final column of Appendix A.2 for examples.

As per the previous section, the team recommends that combining characters be disallowed for TLD labels. If they are disallowed, variant cases arising from combining characters may be ignored. However, if the final solution allows any combining characters in TLDs, Appendix A.2 must then be taken into account. As discussed, this also depends on feedback from communities using the Arabic script for African languages, as encoded by the Unicode standard. Appendix A.2 may still be useful reference for non-TLD label formation.

2. Similar Cases

In these cases, a letter is confusingly similar in shape in at least one of its initial, medial, final or isolated forms with at least one other letter in the group. The groups are formed transitively and cause user confusion (and are therefore considered a form of variants). These groups are listed in Appendix B.

- a. KAF group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request). This is needed as the SWASH KAF version is interchangeably used with other KAFs and the former is considered the same letter, with just stylistic variation in many languages. It is considered different from other KAFs in Sindhi language, for which it has been encoded.
- b. YEH group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request). The additional YEH in this case has a very slight tail at its end, which may be considered a stylistic variation and be confused with regular YEH by most users of Arabic script; but not by speakers of Pashto, who distinguish it from other YEHs.
- c. ALEF with HAMZA ABOVE group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request). The WAVY HAMZA is not distinguishable by most Arabic script users.
- d. ALEF with HAMZA BELOW group - limit as one at TLD level; all are possible for TLD registration (none preferred over any other, and the choice depends on registrant). The WAVY HAMZA is not distinguishable by most Arabic script users.
- e. ALEF with MADDA group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request). The two are considered similar in many (but not all) languages.
- f. YEH with HAMZA group - limit as one at TLD level; all are possible for TLD registration (none preferred over other, depends on registrant request). The two are considered similar in many (but not all) languages.

Dot orientation could cause confusion among Arabic script users and thus can be a potential candidate for character variants, i.e. labels only differing in these would be in the same variant label set. Such confusing cases have been listed in Appendix B.2. It should be investigated further with feedback from relevant language communities (not represented in the team).

Additionally there are some more cases, which are generally not confusing to Arabic script users, but may become confusable in small font size, especially in fonts that are not very accurate in shaping. These include the cases in Appendix B.3.

3. Interchangeable Cases

In some cases, either similar sound or common alternate spellings sometimes cause different characters to be used interchangeably in some languages. This could cause user confusion and should be considered as character variants.

- a. The letters (YEH, ALEF MAKSURA, WAW) with HAMZA may be confused with the corresponding letters without HAMZA and may be used interchangeably.
- b. In the Arabic language, all forms of ALEF (ALEF, ALEF with MADDA ABOVE, ALEF with HAMZA ABOVE, and ALEF with HAMZA BELOW) are used interchangeably. Specifically, ALEF is used instead of all of the other forms (e.g., ALEF with HAMZA ABOVE or ALEF with HAMZA BELOW), ALEF with HAMZA ABOVE is used instead of ALEF with MADDA ABOVE. Part of the writing confusion depends on how it is pronounced. However, ALEF with MADDA is a separate character in other languages, e.g. Urdu.
- c. In Arabic language and many other languages using Arabic script TEH MARBUTA (U+0629) and HEH (U+0647) are used interchangeably at the end of a word by speakers of these languages, because they sound same in the context. Thus, these may cause user confusion in the labels.
- d. There are two sets of digits, Arabic-Indic digits and Extended Arabic-Indic digits. They may be used interchangeably, with each other and with ASCII digits. Though this is not relevant to TLDs (in the root zone), as digits are not allowed in TLDs, this may be a relevant issue for other labels further down in the tree. The subsets of both Arabic sets from 1-3 and 8-9 are identical, but digits 4-7 have different shapes.

4. Optional Cases

There are different kinds of optional marks in Arabic script, and their usage differs across languages. As they are optional and there can be multiple optional marks per letter in a label, their use may create a very high number of variants. Due to their small size (as glyphs) and optional nature, they can also cause a high degree of user confusion and security issues. Therefore, if needed, their use should be very carefully regulated. It is recommended that they are not allowed for the TLDs. The list of optional marks is given below (reproduced from Section 5 above).

1. U+0610 - U+061A: an issue as they are PVALID but should not be allowed for TLDs
2. U+064B - U+0659: an issue as they are PVALID but should not be allowed for TLDs
3. U+065A - U+065F: an issue as they are PVALID but may not be allowed for TLDs
4. U+0670: an issue as it is PVALID but should not be allowed for TLDs
5. U+06D6 - U+06DC: an issue as they are PVALID but should not be allowed for TLDs
6. U+06DF - U+06E8: an issue as they are PVALID but should not be allowed for TLDs
7. U+06EA - U+06ED: an issue as they are PVALID but should not be allowed for TLDs

7. Label Generation Policy

A single Label Generation Policy¹³ needs to be defined for generating TLD labels in the root using the Arabic script. This policy needs to be developed and updated in consultation with the community.

The team agrees that the structure proposed in JET Guidelines (RFC 3743) is not appropriate for Arabic script. There are many differences, e.g. the three column format is not relevant and the relationship between variants is symmetric in Arabic script based labels. Thus a separate effort involving the Arabic Script community needs to be undertaken to consider the structure of Label Generation Policy for Arabic script. Defining the structure is important for effective use, reuse and derivative use of Label Generation Policy¹⁴.

The team further suggests that such policy at the TLD level, which defines the label at the root, needs to be more conservative compared with the other levels because it needs to support all languages using the Arabic script concurrently.

However, when this Arabic Script TLD Label Generation Policy is defined¹⁵, it may consider stipulating: (i) who is the point of contact and owner of this policy, (ii) who participates in defining this policy, (iii) how will this policy be maintained over time, e.g. as Unicode standard is updated, or there is an issue raised to change it by the community, (iv) what will be the process to raise issues to modify it, and (v) what will be the uses of this policy.

Though defining the structure is necessary for effective use of this information, it was considered beyond the scope of the current work on issues. However, it is suggested that the relevant protocol and/or policy development process which addresses this should consider including the following information:

1. List of label valid code points
2. Character variants
3. Additional rules and/or constraints on labels at that level, formally articulated using a context dependent syntax
4. Meta information, which may include optional information e.g. language(s) the policy supports, the contact information for the owner, version, and other relevant information

The community should also develop tools which allow for automatic verification of a suggested label and its variants as per the TLD Label Generation Policy and each label being requested should be evaluated against this policy before further processing.

¹³ This has been referred to as a “language table” elsewhere, but we are not using this terminology as “table” is an implementation detail and in the future such information may also be represented in other forms, e.g. using XML and may contain additional information. Further, the term “language” is also limiting as the current use may be for multiple languages or sometimes for the entire script.

¹⁴ The same structure may be valid for all levels of label generation, though the content may vary (as discussed) and lower level policies may become less conservative than TLDs.

¹⁵ Following appropriate policy development processes at ICANN and protocol development processes at IETF

Should variants be defined at the character positioning level (Initial, Medial, Final, Isolated) or at character level? This particular issue is relevant for both TLD and other levels. Position level variant policy would have dual advantages. First it will generate more label choices for the applicants, without making them any more confusable. And second, it will reduce the number of variants generated against a label. Though this policy may be a bit more complex to implement, it is still quite possible, with **السعودية** (IDN ccTLD of Saudi Arabia) already implementing it for Arabic language. This will have implications on the design of the structure for the Label Generation Policy. For the reasons given, it is a very important issue and needs to be thoroughly discussed.

8. Variant Management for TLD Applicants

Introduction of variants causes more complexity in the application process, as there are many more strings which are generated as variant labels and not all of them may eventually be allocated or delegated. Should there be a limit on the size of the Variant Label Sets (i.e. maximum number of Variant labels to a certain fundamental label), or else the list may be too long? If yes, how should the shorter list be determined?

An applicant for a TLD label would normally request a Fundamental Label. However, now a Variant Label Set will be generated as per the Label Generation Policy. All these labels, except the Blocked Variant Label Subset (which is also determined by the Label Generation policy), will be available for the applicant. As per the decision of the applicant and Label Generation Policy, the available labels will be divided into Allocated Variant Label Subset and Reserved Variant Label Subset. The former would be administratively associated with the applicant and the latter would not be available to anybody else but would not be administratively associated to the applicant. All the Allocated Variant Labels may eventually be activated through delegation or other mechanisms (e.g. DNAME or any other methods available). If only a subset is activated/delegated then this also needs to be defined. There may also be a Blocked Variant Label Subset, which is defined by the Label Generation Policy and not available to any applicant.

Also over time, the applicant may request for changing the status of one or more of the Variant Labels, from reserved to allocated, but not vice versa, etc., or the root registry may change status of some variants, e.g. from blocked to reserved, vice versa, etc.

It needs to be clearly defined what are the multiple states a Variant TLD Label may take and what are the processes to transition from one state to another, including administrative, technical and billing requirements. An associated question is whether history of such transitions should be maintained?¹⁶

Another important question is how this process will be different if the applicant requests change in status of a fundamental label? Will this be possible (should be, if the variants are symmetrical in Arabic script; are there cases where it may not be possible, e.g. in case ZWNJ is allowed)? This may be more complex to deal with as an alternate fundamental label will need to be defined and other variant labels will need to be administratively re-associated.

¹⁶ There may be additional issues for variants of labels at second and lower levels, e.g. the allocation and expiration dates for variants must be synchronized and the transition process should be within those bounds. Discussion of issues related to lower level labels is beyond the scope of this report.



If a TLD is re-delegated or transferred, does that affect the Variant Label Set status? Does the new TLD operator have the option to redefine the various variant subsets (fundamental, allocated, reserved, etc.) differently from those which exist at the time? What is the effect of these choices on the existing registrations?

Additionally, the solutions need to be clearly articulated for applicants for TLDs for them to make informed decisions.

As already highlighted, tools need to be developed or extended to automatically and consistently generate variant label sets against an applied for fundamental label, with no (or minimal) manual intervention.

Finally, it is possible that the Label Generation Policy may change. As it changes, it may modify the Variant Label Set (e.g. additional variant labels may be needed as Unicode expands its repertoire). How will such changes and their impact be communicated to the stakeholders, e.g. the relevant TLD registries?

9. Domain Name Registration Data (WHOIS)

As the domain names become available in Arabic script, there is a clear need for the Domain Name Registration Data to be in the same script. The current WHOIS Protocol (RFC 3912) does not support multilingual data. There are already some efforts in the community to come up with a more comprehensive protocol which also supports this aspect and this should be pursued and finalized.

In addition to the Domain Name Registration Data Access Protocol¹⁷, it should also be clearly defined what Domain Name Registration Data is needed from the applicant/registrant and how much of that will be made available to the public.

It is also crucial to define what Domain Name Registration Data Directory Services are needed to make this non-ASCII data available publically.

As the protocol, data and service needs around the IDNs are being finalized, the support for variants must also be considered. There is a paradigm shift here, where although currently there is a one-to-one lookup for WHOIS against the domain name, henceforth the look up will have to deal with the following scenario, where Label X_i represents a variant of Label X .

$$\left\{ \begin{array}{l} \text{Label } 3_1 \\ \dots \\ \text{Label } 3_n \end{array} \right\} \cdot \left\{ \begin{array}{l} \text{Label } 2_1 \\ \dots \\ \text{Label } 2_m \end{array} \right\} \cdot \left\{ \begin{array}{l} \text{Label } 1_1 \\ \dots \\ \text{Label } 1_n \end{array} \right\}$$

¹⁷ This section is using the terminology being recommended by SSAC Report on Domain Name (WHOIS) Terminology and Structure (SSAC 051)



There are multiple ways variants may be queried, and the possible domain names may have different statuses (e.g. reserved, allocated, delegated, activated, etc.).

It needs to be determined how a request for WHOIS information for a domain name $\text{Label3}_i.\text{Label2}_j.\text{Label1}_k$ may respond (request to information for $.\text{Label1}_k$ in case of TLDs).

The following are examples of some of the points needed to be considered when revising WHOIS services to support IDN:

- a. All allocated variants should have the same ownership
- b. It would be necessary to have the variant information available as part of the Domain Name Registration Data Directory Services
- c. A query on an allocated or reserved label should return the fundamental label; though it is not clear at this time whether the status of the queried label should also be returned along with the fundamental label.
- d. A query on the fundamental label would return the Domain Name Registration Data relevant to the query. Details of what is returned need to be defined (as discussed above).
- e. There is need for an additional query/service which returns the Label Variant Set against a requested domain name. Again it is not clear whether such a service should also return the status of each label in the set.
- f. Would the response against a blocked variant label be different from responses to labels with other status (reserved, allocated, etc.)?
- g. Will the creation and expiration dates of the Variant Label Set be inherited from the fundamental label, as suggested? If yes, then if a variant label is either added or changes its state, how will this information be part of the Domain Name Registration Data? Would history be maintained and communicated for such changes?

10. Fees

Variant TLD labels are required to address the wider (global) access, usability and security of the system. The Arabic Script Case Study team understands that TLD labels introduce a complex mechanism at the root, with a possible new process for TLD label status management. However, the team also notes that the variants are needed due to the inherent ambiguity in the script and its encoding, and is not motivated by extrinsic business interests. Therefore, as activation of variants is needed for a consistent, stable and secure user experience globally, the team recommends that no extra fees be added for variant activation.

As the fees for the gTLD process are already significant, it is worth noting that if extra cost is associated for deploying variants, it may force businesses not to activate them. This will have an adverse impact on accessibility of the TLDs globally and impact on the perceived stability of the Internet from a user perspective.

11. DNSSEC

The registry may need to think more about key management specially when they adopt variants, as multiple keys may be needed This may also cause different expiry dates for different labels

within the same Variant Label Set and may need to be looked into carefully when changing the status of a variant labels.

12. Dispute Resolution

This section addresses the impact on the dispute resolution process due to the activation of variant labels. This may have implications on the dispute resolution policies pre- and post delegation.

i) Filing a Complaint/Case

1. Are all members of a variant set treated as one also with regards to dispute resolution?
2. Could a case exist where a complainer claims right over a subset of a variant set? Will the dispute differ if the disputed domain name(s) is/are active, blocked or reserved? Will the whole variant set be included in this case?
3. Could a case exist where a complainer claims right over a whole variant set? Will all the members be consolidated as one case?
4. May registrants be forced into disputes as a result of registry variant practice such as automated generation of a variant label set? If so, does the present process provide adequate protection for the registrant?
5. Is it possible to have a single dispute spanning two different variant sets?

ii) Payment

1. If dispute resolution for a subset of a variant set is allowed, how does the number of labels under dispute affect the dispute resolution fee? Does consolidation apply?
2. Could the fee differ according to the status of the disputed domain name(s) whether they are active, blocked or reserved?

iii) Decision and Decision Process

1. Could a case exist where a decision (such as cancellation or transfer of ownership) is applied to only a subset of a variant set? If yes, what are the implications? In case the rest of the members were not active does this give the right to the TLD label owner to activate any one or more members of the variant set?
2. Could a case exist where, as a result of dispute resolution process, a TLD label is added to an existing variant set, for example, because it was overlooked in the pre-defined sets?
3. Can a dispute resolution decision have implications on the Label Generation Policy (i.e. can a dispute decision cause changes to an existing TLD Label Generation Policy?) If so, how will that be managed?
4. Should a change in a domain name status, due to its being under dispute, be inherited by the rest of the variants? What will be the process for this change?



iv) Technical Implementation of a Dispute Resolution Decision:

1. What are the implications of changes done on a TLD variant label set as a result of implementing a dispute resolution decision?
2. What are the implications of a dispute resolution decision on Label Generation Policy in terms of implementing changes to variant sets defined? Can a decision on a variant set have impact on other variant sets due to implied changes in the TLD Label Generation Policy? How can this be managed?

v) General

1. Do IDN variant considerations necessitate extending the purview of dispute resolution process beyond trademarks and service marks?

It must be noted here that there is an existing objection process to gTLD applications, and though it seems reasonably generic, the variants may still have an impact on it. The team has not been able to discuss this aspect in greater detail, but suggests this should be further investigated.

13. End-user Requirements

Labels and their variants will need to be configured and used by administrators and users respectively. Appropriate tools and applications need to be developed to assist the process.

a. Keyboard (soft) issues

- i) Lack of standard keyboard for many languages using Arabic script (e.g. for Kurdish, Urdu, Pashto, Sindhi and many other languages). This is also true when a user travels across different countries, trying to access, for example, a Farsi domain name from an Arabic language country (with an Arabic keyboard available locally). This causes variation in typing the same labels, and thus requires activation of variant labels for effective use for Arabic script.
- ii) Digits may also present an issue (not relevant to TLDs under current policies).

b. Font issues

- i) Two very different writing styles are in use by Arabic script community. The Arabic language uses Naskh writing style and Urdu, Pashto, and many other languages in South Asia use Nastaliq writing style. There are many other font styles in use, including Thuluth, Riqa, Kufi and others, for stylistic variation. Fonts may have implications on variant label sets.
- ii) There may be variation within a writing style across different fonts, e.g. see variety in rendering of HEH in Naskh style in Appendix C.

- iii) Font support may not be consistent across variant label set, as a font may not support all character variants, causing strings to break during display or boxes or other unexpected characters to show up. This may cause user confusion by not making different variants visible, and by making labels that are not variants to appear as same (e.g. two different letters mapped onto a box making them indistinguishable).

c. Concurrent Display of A-label and U-label for Administrative Purposes

Tools are needed for effective and easy operation. Though such tools are available for ASCII, they may not be available for Arabic script and may not handle variants conveniently.

- i) There is a need to develop tools to manage variant labels. For example, currently there are no EPP extensions to handle variants during the registration process.
- ii) Similarly the administration tools only deal with ASCII, meaning that server administration will only be possible using A-Labels. Tools need to be developed to view these A-Labels along with the U-Labels to avoid management errors and to better identify the variants.
- iii) The two cases above only present first examples of many other tools which need to be developed for different functions and which use domain names and labels. A more exhaustive list needs to be worked out and eventually addressed. Though these are not relevant to TLDs, they are certainly relevant to their eventual use in the domain name system.
- iv) Most of the DNS management and resolving software – if not all – can resolve ASCII characters only. Thus, when creating new IDN entries in zone files, they must be entered in ASCII rather than Unicode. The same will apply to variants and variant TLDs.

d. Bidirectional Issues

- i) Arabic script domain name may have characters of different directionality (Arabic letters, Arabic digits, Arabic Indic digits, ASCII LDH, etc.) mixed within a label or across labels or both. This causes inconsistent display across various applications and may confuse users. Such cases need to be investigated and displayed in a consistent manner in applications both within labels and across labels at different levels.

e. Operating System Support Issues

- i) Operating System support may not be consistent across variants of the same label, as an OS may not support all variant characters in a language table, causing strings to break during display or boxes or other unexpected characters to show up. This may cause user confusion by not making different variants visible, and by making labels which are not variants to appear as same (e.g. two different letters mapped onto a box making them

indistinguishable).

f. Application Issues¹⁸

- i) Applications may not be consistent across variants of the same label, as an application may not support all variant characters in a language table, causing strings to break during display or boxes or other unexpected characters to show up. This may cause user confusion by not making different variants visible, and by making labels which are not variants to appear as same (e.g. two different letters mapped onto a box making them indistinguishable).
- ii) While most of the latest web browsers have implemented IDNA2003, not many of them have implemented IDNA2008. Furthermore, while some old browsers do not support IDN extensions, some do support it, but do not display the U-Label format of an IDN (it breaks the label down into its ASCII format and displays it). Variants of TLDs should not be an issue for many of the web browsers as the user has to configure a language on the browser settings. However, some web browsers – such as Mozilla Firefox – require a special configuration in which a whitelist tag must be add as network.IDN.whitelist.<ASCII format of the TLD>. This means that for each variant TLD, a new tag must be added.
- iii) Currently, internationalized email is under heavy development, and the standards are not yet finalized much less deployed. A working group within the IETF – E-Mail Address Internationalization (EAI) – is currently working to resolve this issue. Since e-mails under different TLDs will require independent setups at the server end, the same would apply to variants since each variant TLD has a different A-Label representation.
- iv) Currently internationalized email is experimental. The same issue will apply to variants and variant TLDs. Furthermore, for each IDN e-mail variant, a new setup must be configured so that e-mails received from the all configured variants of an e-mail account can be received.
- v) Most of the office productivity tools – if not all – do not support IDNs; i.e. identifying and creating automatic hyperlinks for IDN domain names. The same issue will apply to the variants and variant TLDs.
- vi) Most – if not all – of online blogs and content management systems (CMS) do not support IDNs. Some might support IDNs for certain languages; related to language communities that are active in the production of online content in their native languages, most of the social networks still

¹⁸ Some of these issues are motivated through “Discussion Points: IDN Software Developer’s Consortium” report



does not recognize or display IDN domain names properly. The same applies to variants and variant TLDs.

vii) There are few IDNA-aware tools for network use and diagnostics. In particular, there are few tools available that use the U-label form of an IDN as their input. Performing diagnostics and doing simple operations on the U-label form of an IDN is still mostly manual, and network diagnostics always depend on having the A-label form.

viii) Most of the current mobile applications do not support IDNs. The same will apply to variants and variant TLDs.

g. Search Engines

i) Search engines must be optimized to cope with IDNs so that they offer IDN results. The same applies to variants and variant TLDs. Furthermore, even when search engines are optimized to provide IDN results, and assuming that an IDN has several variants across several languages within a script, which IDN would it return back?

h. Hosting Service Providers

i) Many of the hosting service providers cannot deal with IDN U-Label. Furthermore, web hosting tools, such as the control panels cannot deal with IDNs in U-label form. The same applies to variants and variant TLDs.

i. SSL Certificates

In order to configure a secure HTTP connection for a website, an SSL certificate must be added. If a website can be accessed using several domain names, each domain name will require an independent SSL certificate. And since each IDN variant holds a unique A-Label, each A-Label will require an independent SSL certificate. Thus, one will require $n+1$ SSL certificates if we assume an IDN and n variants for a certain domain name. This is equally true for all other TLS-using protocols.



Arabic Case Study Team

The Arabic Case Study Team convened for the first time at the ICANN meeting in Singapore in June 2011. Since then, the team has been conducting its work online and through weekly conference calls. The team met face-to-face in Doha, Qatar on 13-15 September 2011.

List of Contributors

Sarmad Hussain	Case Study Coordinator
Abdulaziz Al-Zoman	Team Member
Behnam Esfahbod	Team Member
Fahd A. Batayneh	Team Member
Huda Sarfraz	Team Member
Iftikhar H. Shah	Team Member
Khaled Koubaa	Team Member
Manal Ismail	Team Member
Mohamed El-Bashir	Team Member
Raed Al-Fayez	Team Member
Siavash Shahshahani	Team Member
Alireza Saleh	Team Observer

ICANN staff and consultants:

Baher Esmat	Case Study Liaison
Francisco Arias	Subject Matter Expert (Registry Operations)
Kim Davies	Subject Matter Expert (Security)
Andrew Sullivan	Subject Matter Expert (Protocols)
Nicholas Ostler	Subject Matter Expert (Linguistics)

Local Host Organization

The Supreme Council of Information and Communication Technology (ictQatar) is the local host organization of the Arabic Case Study Team. The team recognizes the support provided by ictQatar, particularly in hosting the face-to-face meeting in Doha on 13-15 September 2011.



Appendices

Important Notes:

- 1. The tables provided are based on the assumption that the variants will be defined at character level. If the variants are defined at position level, these tables will need to be re-analyzed and significantly revised*
- 2. The tables are dependent on, and may change with, the Unicode versions*
- 3. The tables are strictly proposed for TLD labels at root level. Registries may implement very different variant label generation policy at second and other levels*
- 4. Additional confusions between characters may occur in stylistically different fonts, especially those that do not adhere to Unicode standard. Such analysis is not in the current scope of work*
- 5. Though the tables are intended to be as comprehensive as possible, a thorough analysis is still recommended in the solutions phase*

Appendix A. Identical Character Variants

Appendix A.1. Same Shape in at least one Position

Unicode	Initial Form	Medial Form	Final Form	Isolated Form
KAF Group				
U+06A9 (ك)	ك	ك	ك	ك
U+0643 (ك)	ك	ك	ك	ك
HEH Group				
U+0647 (ه)	ه	ه	ه	ه
U+06BE (ه)	ه	ه	ه	ه
U+06C1 (ه)	ه	ه	ه	ه
U+06D5 (ه)	ه	ه	ه	ه
YEH Group				
U+064A (ي)	ي	ي	ي	ي
U+06CC (ي)	ي	ي	ي	ي
U+0649 (ي)	ي	ي	ي	ي
BEEH/E Group				
U+067B(ب)	ب	ب	ب	ب
U+06D0(ب)	ب	ب	ب	ب
FEH Group				
U+06A7 (ف)	ف	ف	ف	ف
U+0641 (ف)	ف	ف	ف	ف

VEH Group				
U+06A4 (ف)	ف	ف	ف	ف
U+06A8 (ق)	ق	ق	ق	ق
THE MARBUTA Group				
U+0629 (ة)	-	-	ة	ة
U+06C3 (ة)	-	-	ة	ة
HEH with HAMZA Group				
U+06C0 (ه)	-	-	ه	ه
U+06C2 (ه)	-	-	ه	ه
TTEY/RNOON Group				
U+06BB (ط)	ط	ط	ط	ط
U+0679 (ط)	ط	ط	ط	ط
NOON Group				
U+0646 (ن)	ن	ن	ن	ن
U+06BA (ن)	ن	ن	ن	ن
NOON/PEH Group				
U+06BD (ن)	ن	ن	ن	ن
U+067E (پ)	پ	پ	پ	پ

Appendix A.2. Same Shape in Composed and Decomposed Forms

Appendix A.2.1. Same Shape in Composed and Decomposed Forms using a single Combining Mark


Combining Mark	Composed Form	Decomposed Form	Unicode Normalized Form	
◌̣ U+0653	آ U+0622	◌̣ ا U+0627 U+0653	Defined	
◌̣ U+0654	أ U+0623	◌̣◌̣ ا U+0627 U+0654	Defined	
	ؤ U+0624	◌̣ و U+0648 U+0654	Defined	
	ئ U+0626		◌̣ ي U+064A U+0654	Defined
			ئ U+0649 U+0654	Not Defined
			ئ U+06CC U+0654	Not Defined
	ة U+06C0		◌̣ ه U+06D5 U+0654	Defined
			◌̣ ه U+0647 U+0654	Not Defined
	ة U+06C2		◌̣ ه U+06C1 U+0654	Defined
			◌̣ ه U+0647 U+0654	Not Defined
	ة U+06D3		◌̣ ه U+06D2 U+0654	Defined
	ح U+0681		◌̣ ح U+062D U+0654	Not Defined

Combining Mark	Composed Form	Decomposed Form	Unicode Normalized Form
	رُ U+076C	رُ U+0631 U+0654	Not Defined
◌ِ U+0655	إ U+0625	اِ U+0627 U+0655	Defined
◌ُ U+064F	ؤ U+06C7	وُ U+0648 U+064F	Not Defined
		وُ U+0648 U+0619	Not Defined
◌ِ U+0670	و U+06C8	وِ U+0648 U+0670	Not Defined
◌ِ U+06EC	و U+06CF	وِ U+0648 U+06EC	Not Defined
	غ U+063A	عِ U+0639 U+06EC	Not Defined
	ض U+0636	صِ U+0635 U+06EC	Not Defined
	خ U+062E	حِ U+062D U+06EC	Not Defined
	چ U+06BF	چِ U+0686 U+06EC	Not Defined
	ذ U+0630	دِ U+062F U+06EC	Not Defined
	ز U+0632	رِ U+0631 U+06EC	Not Defined
	ل U+06B6	لِ U+0644 U+06EC	Not Defined
	ف U+06A7	فِ U+066F U+06EC	Not Defined
	ف U+06A7	فِ U+066F U+06EC	Not Defined

Combining Mark	Composed Form	Decomposed Form	Unicode Normalized Form
	U+0641	U+06A1 U+06EC	
	ن U+0646	ن U+06BA U+06EC	Not Defined
	ك U+06AC	ك U+0643 U+06EC	Not Defined
	ك U+0762	ك U+06A9 U+06EC	Not Defined
	م U+0765	م U+0645 U+06EC	Not Defined
ط U+0615	ح U+0772	ح U+062D U+0615	Not Defined
	ط U+0679	ط U+066E U+0615	Not Defined
	ر U+0691	ر U+0631 U+0615	Not Defined
	د U+0688	د U+062F U+0615	Not Defined
	ز U+0771	ز U+0697 U+0615	Not Defined
	ن U+0768	ن U+0646 U+0615	Not Defined
	ب U+068B	ب U+068A U+0615	Not Defined
	س U+06BB	س U+0648 U+0615	Not Defined

Combining Mark	Composed Form	Decomposed Form	Unicode Normalized Form	
		U+06BA U+0615		
◌َ U+065B	ى U+063D	◌ِى U+06CC U+065B	Not Defined	
	و U+06C9	◌ِو U+0648 U+065B	Not Defined	
	س U+077E	◌ِس U+0633 U+065B	Not Defined	
	د U+06EE	◌ِد U+062F U+065B	Not Defined	
	ر U+06EF	◌ِر U+0631 U+065B	Not Defined	
	ه U+06FF		◌ِه U+06BE U+065B	Not Defined
			◌ِه U+0647 U+065B	Not Defined
◌ِ U+06DB	ى U+063F	◌ِى U+06CC U+06DB	Not Defined	
		◌ِى U+0649 U+06DB	Not Defined	
	س U+0634	◌ِس U+0633 U+06DB	Not Defined	
	پ U+069C	◌ِپ U+069B U+06DB	Not Defined	
	ت U+062B	◌ِت U+066E U+06DB	Not Defined	
	ح U+0685	◌ِح U+062D U+06DB	Not Defined	
	ر U+0698	◌ِر U+0631 U+06DB	Not Defined	

Combining Mark	Composed Form	Decomposed Form	Unicode Normalized Form
	ذ U+068E	د̇ U+062F U+06DB	Not Defined
	ع U+06A0	ع̇ U+0639 U+06DB	Not Defined
	ف U+06A4	ف̇ U+06A1 U+06DB	Not Defined
	ق U+06A8	ق̇ U+066F U+06DB	Not Defined
	ك U+06AD	ك̇ U+0643 U+06DB	Not Defined
	گ U+06B4	گ̇ U+06AF U+06DB	Not Defined
	ل U+06B7	ل̇ U+0644 U+06DB	Not Defined
	ن U+06BD	ن̇ U+06BA U+06DB	Not Defined
	ش U+0763	ك̇ U+06A9 U+06DB	Not Defined
◌̇ U+065C	ب U+0628	ب̇ U+066E U+065C	Not Defined
	د U+068A	د̇ U+062F U+065C	Not Defined
	ذ U+068B	ذ̇ U+0688 U+065C	Not Defined
	ر U+0694	ر̇ U+0631 U+065C	Not Defined
	ف U+06A3	ف̇ U+0641 U+065C	Not Defined
	ن U+06BD	ن̇ U+06BA U+065C	Not Defined

Combining Mark	Composed Form	Decomposed Form	Unicode Normalized Form
	U+06B9	U+0646 U+065C	
	غ U+06FC	غ U+063A U+065C	Not Defined
	ض U+06FB	ض U+0636 U+065C	Not Defined
	ث U+0751	ث U+062B U+065C	Not Defined
	م U+0766	م U+0645 U+065C	Not Defined
 U+065A	ل U+06B5	ل U+0644 U+065A	Not Defined
	و U+06C6	و U+0648 U+065A	Not Defined
	ي U+06CE	ي U+06CC U+065A	Not Defined
		ي U+0649 U+065A	Not Defined
	ب U+0756	ب U+066E U+065A	Not Defined
	ن U+0769	ن U+0646 U+065A	Not Defined

Appendix A.2.2. Same Shape in Composed and Decomposed forms using Two Combining Marks¹⁹

Composed Form	Decomposed Form
بَس U+069A	س َ ِ U+0633 U+065C U+06EC
فَب U+06A3	ف َ ِ U+06A1 U+065C U+06EC
بَسْ U+06FA	س ِ َ U+0633 U+06DB U+065C
جُص U+06FB	ص ِ َ U+0635 U+065C U+06EC
عُغ U+06FC	ع ِ َ U+0639 U+065C U+06EC
بُن U+06B9	ن ِ َ U+06BA U+065C U+06EC

¹⁹ Possibilities with more than two combining marks have not been investigated at this time

Appendix B. Confusable Similar Letters in Arabic Script

Appendix B.1. Similar Shape in at least One Position

Unicode	Initial Form	Medial Form	Final Form	Isolated Form
KAF Group				
U+06A9 (كـ)	ك	ك	ك	ك
U+06AA (ڪـ)	ڪ	ڪ	ڪ	ڪ
U+0643 (ك)	ك	ك	ك	ك
YEH Group				
U+064A (ي)	ي	ي	ي	ي
U+06CC (ى)	ي	ي	ى	ى
U+0649 (ى)	-	-	ى	ى
U+06D2 (ے)	-	-	ے	ے
U+06CD (ى)	-	-	ى	ى
ALEF with HAMZA ABOVE Group				
U+0623 (أ)	-	-	أ	أ
U+0672 (أ)	-	-	أ	أ
ALEF with HAMZA BELOW Group				
U+0625 (إ)	-	-	إ	إ
U+0673 (إ)	-	-	إ	إ
ALEF with MADDA Group				
U+0622 (آ)	-	-	آ	آ
U+0671 (آ)	-	-	آ	آ
YEH with HAMZA Group				
U+0626 (ئ)	ئ	ئ	ئ	ئ
U+06D3(ے)	-	-	ے	ے

Appendix B.2. Confusable Similar Shape with Difference in Dot Orientation



















Unicode		Characters	
i)	U+062A	i)	ﺍ
ii)	U+067A	ii)	ﺍٓ
i)	U+062B	i)	ﺏ
ii)	U+067D	ii)	ﺏٓ
i)	U+063C	i)	ﻙ
ii)	U+0764	ii)	ﻙٓ
i)	U+064A	i)	ﻱ
ii)	U+06D0	ii)	ﻱٓ
iii)	U+067B	iii)	ﺏٓ
i)	U+067E	i)	ﻱٓ
ii)	U+0752	ii)	ﻱٓ
i)	U+0683	i)	ﻥ
ii)	U+0684	ii)	ﻥٓ
i)	U+0686	i)	ﻥٓ
ii)	U+0758	ii)	ﻥٓ
i)	U+068E	i)	ﻱٓ
ii)	U+068F	ii)	ﻱٓ
i)	U+06A0	i)	ﻱٓ
ii)	U+075F	ii)	ﻱٓ
i)	U+06B2	i)	ﻱٓ
ii)	U+06B3	ii)	ﻱٓ
i)	U+075D	i)	ﻱٓ
ii)	U+075F	ii)	ﻱٓ
i)	U+0697	i)	ﺏ



















ii) U+076B	ii) ز
------------	-------

Appendix B.3. Possibly Confusable Similar Shape in Small Font Sizes





Unicode	Initial Form	Medial Form	Final Form	Isolated Form
GHAIN/FEH Group				
U+063A (غ)	غ	غ	غ	غ
U+0641 (ف)	ف	ف	ف	ف
QAF/AIN with 2 DOTS ABOVE Group				
U+0642 (ق)	ق	ق	ق	ق
U+065D (غ)	غ	غ	غ	غ
AIN/FEH/QAF with 3 DOTS ABOVE Group				
U+06A0 (غ)	غ	غ	غ	غ
U+06A4 (ف)	ف	ف	ف	ف
U+06A8 (ق)	ق	ق	ق	ق





Appendix C: Shape Variations with Fonts²⁰



















Representative shape in code chart	Possible shapes in context [Standalone, End, Middle, Beginning]				Font
 0647 HEH					Arabic Typesetting
 06BE KNOTTED HEH					Arabic Typesetting
 06C1 HEH GOAL					Arabic Typesetting
 06D5 AE					Arabic Typesetting

Representative shape in code chart	Possible shapes in context [Standalone, End, Middle, Beginning]				Font
 0647 HEH					Tahoma
 06BE KNOTTED HEH					Tahoma
 06C1 HEH GOAL					Tahoma
 06D5 AE					Tahoma

²⁰ The above is a result from the following tool (need to have the fonts in order to display them correctly): http://arabic-domains.org/adn_tools/compareChars/arabic-script-list-all.php

Representative shape in code chart	Possible shapes in context [Standalone, End, Middle, Beginning]				Font
 0647 HEH	◌	◌	◌	◌	DecoType Naskh
 06BE KNOTTED HEH	◌	◌	◌	◌	DecoType Naskh
 06C1 HEH GOAL	◌	◌	◌	◌	DecoType Naskh
 06D5 AE	◌	◌			DecoType Naskh

Representative shape in code chart	Possible shapes in context [Standalone, End, Middle, Beginning]				Font
 0647 HEH	◌	◌	◌	◌	DecoType Thuluth
 06BE KNOTTED HEH	◌	◌	◌	◌	DecoType Thuluth
 06C1 HEH GOAL	◌	◌	◌	◌	DecoType Thuluth
 06D5 AE	◌	◌			DecoType Thuluth

Representative shape in code chart	Possible shapes in context [Standalone, End, Middle, Beginning]				Font
 0647 HEH					Farsi Simple Bold
 06BE KNOTTED HEH					Farsi Simple Bold
 06C1 HEH GOAL					Farsi Simple Bold
 06D5 AE					Farsi Simple Bold

Appendix D: Additional Constraints for ZWNJ

<p>طِبل</p> <p>input[0] = U+0637 input[1] = U+200c input[2] = U+0628 input[3] = U+0644</p>	≠	<p>طِبل</p> <p>input[0] = U+0637 input[1] = U+0628 input[2] = U+0644</p>
<p>لهل</p> <p>input[0] = U+0644 input[2] = U+06BE input[1] = U+200c input[3] = U+0644</p>	≠	<p>لهل</p> <p>input[0] = U+0644 input[1] = U+06BE input[2] = U+0644</p>
<p>لهل</p> <p>input[0] = U+0644 input[2] = U+06FF input[1] = U+200c input[3] = U+0644</p>	≠	<p>لهل</p> <p>input[0] = U+0644 input[1] = U+06FF input[2] = U+0644</p>



Appendix E: IDN Variant Issues Project – Arabic Script Case Study Definitions

This section includes all the terms needed and were used in the IDN Variant Issues Project - Arabic Script Case Study. The terms included in this section comprise a complete set of all the definitions needed in the context of Arabic Script IDN Variants and are either newly introduced or referenced from other existing resources, as elaborated below:

Newly defined terms:

Those are terms that are being introduced to address a certain need within the Arabic-Script-based language communities. It's worth noting that this need may or may not exist for other scripts case studies.

Already defined terms:

Those are terms that are already defined elsewhere, namely:

- Draft Definitions for the ICANN Variant Issues Project Document
- Unicode website
- RFC 6365
- RFCs 5890, 5892, and 5893

They are either used as is where they accurately describe the meaning intended by the Arabic Script Case Study or edited to accurately describe the Arabic script case and footnoted as 'Motivated from original source'.

Scope of Variants in the context of the Arabic Script Case Study:

Strings that when registered as TLDs would cause confusion that may result in:

- Unexpected end-user experience
- DNS security and stability issues

Abstract Character²¹:

A unit of information used for the organization, control, or representation of textual data. (Unicode Standard, section 3.4, D7)

Code Point²²:

A value in the Unicode code space. The meaning here is restricted to meaning D10 in the Unicode Standard, section 3.4.

Assigned Code Point²³:

A mapping from an Abstract Character to a particular Code Point in the code space. See Unicode Standard, section 2.4. Not to be confused with Valid Code Point.

Arabic Letter²⁴:

Characters that are used to write Arabic script based languages and used to write words.

²¹ From Draft Definitions for the ICANN Variant Issues Project Document

²² From Draft Definitions for the ICANN Variant Issues Project Document

²³ From Draft Definitions for the ICANN Variant Issues Project Document

²⁴ Motivated from: <http://unicode.org/glossary/#L>



Non-Joining Characters²⁵:

Those characters that do not connect to letters before or after them; e.g. U+0621 LETTER HAMZA, U+0674 HIGH HAMZA, and U+200C ZWNJ.

Right-Joining Characters²⁶:

Those characters that connect to the letter before them; e.g. all letters based on ALEF, REH, DAL, and WAW, and a few other letters.

Dual-Joining Characters²⁷:

Those characters that connect to the letters before and after them; e.g. all other Arabic letters than those listed above.

Join-Causing Characters²⁸:

Those characters that connect to the letters before and after them, but do not change shape themselves; i.e. only U+200D ZWJ and U+0640 TATWEEL.

With respect to those categories, Arabic Script Letters could be defined as follows:

Non-Joining Letters:

The group of non-joining characters which are letters (by Unicode's definition); i.e. U+0621 LETTER HAMZA and U+0674 HIGH HAMZA.

Joining Letter:

The union of Right-Joining Letters and Dual-Joining Letters which cursively join with letters following them.

Right-Joining Letters:

The group of right-joining characters which are letters; i.e. all letters based on ALEF, REH, DAL, and WAW, and a few other letters.

Dual-Joining Letters:

The group of dual-joining characters which are letters; i.e. all other Arabic letters.

Arabic-Indic Digits²⁹:

Forms of decimal digits commonly used along with Arabic script and comprised of two sets of digits. The set <U+0660-9> is commonly used in Arabic-speaking world, while the set <U+06F0-9>, often referred to as Eastern Arabic-Indic, is used in Iran and Pakistan. Although [European digits](#) (1, 2, 3,...) derive historically from these forms, they are visually distinct and are coded separately. (Arabic-Indic digits are sometimes called Indic numerals; however, this nomenclature leads to confusion with the digits currently used with the scripts of India.)

²⁵ Unicode book, Chapter8, table 8-3, page 248 of latest edition

²⁶ Unicode book, Chapter8, table 8-3, page 248 of latest edition

²⁷ Unicode book, Chapter8, table 8-3, page 248 of latest edition

²⁸ Unicode book, Chapter8, table 8-3, page 248 of latest edition

²⁹ Motivated from: http://unicode.org/glossary/#arabic_digits



Arabic Digits³⁰:

The term "Arabic digits" may mean either the digits in the Arabic script (see above and [Arabic-Indic digits](#)) or the ordinary ASCII digits. When the term "Arabic digits" is used in Unicode specifications, it means Arabic-Indic digits.

Combining Marks³¹:

A commonly used synonym for [combining character](#); a character with the General Category of Combining Mark (M).

Label Valid Character:

An Abstract Character which can be used to form a label, and can be a Letter, Digit or another type.

Ligature³²:

A single glyph representing a combination of one or more Arabic Letters. This means that the isolated form of a character can be considered a ligature. It's worth noting that this is different from the Unicode use of the term.

Form of a Letter - Arabic Script:

A Letter in Arabic Script can occur in up to four different forms within a ligature. These include the following:

Isolated form:

It is the standalone form of a Letter, i.e. when the letter does not join with any other letter, forming a single letter.

Initial form:

It is the form of a right-joining-letter when it occurs in the beginning of a ligature, joined with at least one more letter after it, to form a ligature.

Medial form:

It is the form of a right-joining-letter when it occurs in the middle of a ligature, joined with at least one letter on either side, to form a ligature.

Final form:

It is the form of a joining-letter when it occurs at the end of a ligature, joined with at least one more character before it, to form a ligature.

Glyph³³:

(1) An abstract form that represents one or more glyph images. (2) A synonym for [glyph image](#). In displaying Unicode character data, one or more glyphs may be selected to depict a particular character. These glyphs are selected by a rendering engine during composition and layout processing.

³⁰ Motivated from: http://unicode.org/glossary/#arabic_digits

³¹ Source: <http://unicode.org/glossary/#C>

³² Motivated from: <http://unicode.org/glossary/#L>

³³ Source: <http://unicode.org/glossary/#G>



Glyph Image³⁴:

The actual concrete image of a glyph representation having been rasterized or otherwise imaged onto some display surface.

Writing style³⁵:

Conventions of writing the same script in different styles. Different communities using the script may find text in different writing styles difficult to read and possibly unintelligible. For example, the Perso-Arabic Nastaliq writing style and the Arabic Naskh writing style both use the Arabic script but have very different renderings and are not mutually comprehensible. Writing styles may have significant impact on internationalization; for example, the Nastaliq writing style requires significantly more line height than Naskh writing style.

Font³⁶:

A collection of glyphs used for the visual depiction of character data. A font is often associated with a set of parameters (for example, size, posture, weight, and seriffness), which, when set to particular values, generate a collection of imagable glyphs.

Valid Label:

A Label (U-Label or A-Label) which is valid as per the Label Generation Policy.

Valid TLD Label:

A Label which is valid for Top Level Domain as per a TLD Label Generation Policy.

Protocol Valid Code Point³⁷:

A Code Point that is allowed to be used in IDNs. Code points with this property value are permitted for general use in IDNs. However, that a label consists only of code points that have this property value does not imply that the label can be used in any given zone.

Label Valid Code Point³⁸:

The subset of Protocol Valid Code Point listed in the Label Generation Policy, and which may be used to form a label.

TLD Label Valid Code Point³⁹:

The subset of Label Valid Code Point that may be used to form a TLD label.

Arabic Script Character Variant⁴⁰:

A Label Valid Character which is replaceable with another Label Valid Character within a label, as defined by a Label Generation Policy. The relationship is symmetric in Arabic script.

³⁴ Source: <http://unicode.org/glossary/#G>

³⁵ Source: <http://www.rfc-editor.org/rfc/rfc6365.txt>

³⁶ Source: <http://unicode.org/glossary/#F>

³⁷ Motivated from <http://www.rfc-editor.org/rfc/rfc5892.txt>

³⁸ Motivated from Draft Definitions for the ICANN Variant Issues Project Document 'Valid Code Point'

³⁹ Motivated from Draft Definitions for the ICANN Variant Issues Project Document 'Valid Code Point'

⁴⁰ Motivated from Draft Definitions for the ICANN Variant Issues Project Document 'Character Variant'



Label Generation Policy⁴¹:

A formal specification that can be used to formulate or validate a label and determine whether two labels can be considered distinct for allocation. If two labels are not considered distinct for allocation, as per the policy, they are referred to as variants of each other. Variants are symmetric in Arabic script.

TLD Label Generation Policy

A formal specification that can be used to formulate or validate a TLD label and determine whether two TLD labels can be considered distinct for allocation.

Arabic Script Label Generation Policy

Policy specified to generate labels for Arabic script. For Arabic script, this would include at least a list of Protocol Valid Code Points allowed in forming labels, their Character Variants, additional Label formation constraints/rules and meta information (e.g. including script, owner, version, date, etc.).

Arabic Script TLD Label Generation Policy

Policy specified to generate TLD labels for Arabic script.

Variant Character Set⁴²

The set of code points consisting of a Valid Code Point and all of its variants.

Script Table⁴³:

A Script Table is a table of Unicode Code Points all having the same script property value. See Unicode Standard Annex #24.

A-label⁴⁴:

An ASCII-Compatible Encoding form of an IDNA-valid string. It must be a complete label: IDNA is defined for labels, not for parts of them and not for complete domain names. This means, by definition, that every A-label will begin with the IDNA ACE prefix, "xn--", followed by a string that is a valid output of the Punycode algorithm (RFC 3492) and hence a maximum of 59 ASCII characters in length. The prefix and string together must conform to all requirements for a label that can be stored in the DNS including conformance to the rules for LDH labels (See RFC 5390, Section RFC 2.3.1). If and only if a string meeting the above requirements can be decoded into a U-label is it an A-label. (RFC 5890)

U-label⁴⁵:

⁴¹ This term is being suggested as a replacement for “Language Table” as the latter has been found inappropriate as it “Language Table” is not limited to a single language and may represent many languages or an entire script, and “table” represents an implementation level decision (as eventually an XML based or other specification could also be used. Also, language table seems to be a binding, if not insufficient, mechanism to specify additional rules/constraints needed and does not clearly specify method or contents for meta information required to promote unambiguous use and reuse of the information.

⁴² Motivated from Draft Definitions for the ICANN Variant Issues Project Document 'Variant Character Collection'

⁴³ Motivated from Draft Definitions for the ICANN Variant Issues Project Document 'Script Table'

⁴⁴ Motivated from Draft Definitions for the ICANN Variant Issues Project Document 'A-label'



An IDNA-valid string of Unicode Code Points, in Normalization Form C (NFC) and including at least one non-ASCII character, expressed in a standard Unicode Encoding Form (such as UTF-8). It is also subject to the constraints about permitted characters that are specified in Section 4.2 of RFC 5891 and the rules in the Sections 2 and 3 of RFC 5892, the Bidi constraints in RFC 5893 if it contains any character from scripts that are written right to left, and the IDNA Symmetry Constraint. (RFC 5890)

Variant Label:

A U-label considered a variant of a Fundamental Label as per the Label Generation Policy.

Fundamental Label⁴⁶:

A Valid Label which is, in practice, the label received by the registry to be allocated and from which a registry or registrar may generate Variant Labels.

Activated Variant Label:

A Variant Label that is activated by a registry.

Allocated Variant Label:

A Variant Label that is allocated by a registry.

Reserved Variant Label:

A Variant Label that is reserved by a registry without allocation but may be allocated on request

Blocked Variant Label:

A Variant Label that is blocked by a registry (to avoid a conflict) and is not allowed to be allocated

Variant Label Set⁴⁷:

A set of U-labels consisting of one Fundamental Label and its zero or more Variant Labels.

Activated Variant Label Subset:

The subset of Variant Label Set that is activated, or alternatively, the set containing the Fundamental Label and all its Activated Variants.

Allocated Variant Label Subset:

The subset of Variant Label Set that is allocated, or alternatively, the set containing the Fundamental Label and all its Allocated Variants.

Reserved Variant Label Subset:

The subset of Variant Label Set that is reserved, or alternatively, the set containing all the Reserved Variants of a Fundamental Label (and the Fundamental Label, if it is not activated).

Blocked Variant Label Subset:

⁴⁵ Motivated from Draft Definitions for the ICANN Variant Issues Project Document 'U-label'

⁴⁶ Motivated from Draft Definitions for the ICANN Variant Issues Project Document 'Fundamental Label'

⁴⁷ Motivated from Draft Definitions for the ICANN Variant Issues Project Document 'Variant Label Set'



The subset of Variant Label Set that is blocked, or alternatively, the set containing all the Blocked Variants of a Fundamental Label.

Allocation⁴⁸:

In a DNS context, the first step on the way to Delegation. A registry (the parent side) is managing a zone. The registry makes an administrative association between a string and some entity that requests the string, making the string a label inside the zone, and a candidate for delegation. Allocation does not affect the DNS itself at all.

Delegation⁴⁹:

In a DNS context, the act of entering parent-side NS (nameserver) records in a zone, thereby creating a subordinate namespace with its own SOA (start of authority) record. See RFC 1034 for detailed discussion of how the DNS name space is broken up into zones.

Activation:

The process of making a domain name resolvable.

Reservation:

In Arabic Script IDN variants context, this is the process of having an unallocated variant label which relates to a Fundamental label that is allocated.

Blocking:

In Arabic Script IDN variants context, this is the process of having a variant label not allowed for allocation to anyone as long as its Fundamental label is allocated.

⁴⁸ From Draft Definitions for the ICANN Variant Issues Project Document

⁴⁹ From Draft Definitions for the ICANN Variant Issues Project Document