Network Working Group
Internet-Draft
Updates: 5982 (if approved)
Intended status: Standards Track
Expires: January 20, 2015

J.C. Klensin
P. Faltstrom
Netnod

July 21, 2014

IDNA Update for Unicode 7.0.0
draft-klensin-idna-5892upd-unicode70-00.txt

Abstract

The current version of the IDNA specifications anticipated that each new version of Unicode would be reviewed to verify that no changes had been introduced that required adjustments to the set of rules and, in particular, whether new exceptions or backward compatibility adjustments were needed. That review was conducted for Unicode 7.0.0 and identified a problematic new code point. This specification updates RFC 5982 to disallow that code point and provides information about the reasons why that exclusion is appropriate. It also applies an editorial clarification that was the subject of an earlier erratum.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 20, 2015.

Copyright Notice

Table of Contents

1. Introduction

The current version of the IDNA specifications, known as "IDNA2008" [RFC5890], anticipated that each new version of Unicode would be reviewed to verify that no changes had been introduced that required adjustments to IDNA's rules and, in particular, whether new exceptions or backward compatibility adjustments were needed. When that review was carefully conducted for Unicode 7.0.0 [Unicode7], comparing it to prior versions including the text in Unicode 6.2 [Unicode62], it identified a problematic new code point (U+08A1, ARABIC LETTER BEH WITH HAMZA ABOVE). Section 2 of this specification updates the portion of the IDNA2008 specification that identifies rules for what characters are permitted [RFC5892] to disallow that code point. It also provides information about the reasons why that exclusion is appropriate.

As anticipated when IDNA2008, and RFC 5892 in particular, were written, exceptions and explicit updates are likely to be needed only if there is disagreement between the Unicode Consortium's view about what is best for the Standard and the IETF's view of what is best for IDNs, the DNS, and IDNA. It was hoped that a situation would never arise in which the the two perspectives would disagree, but the possibility was anticipated and considerable mechanism added to RFC 5890 and 5982 as a result. It is probably important to note that a disagreement in this context does not imply that anyone is "wrong", only that the two different groups have different needs and therefore criteria about what is acceptable. For that reason, the IETF has, in the past, allowed some characters for IDNA that active Unicode Technical Committee members suggested be disallowed to avoid a change in derived tables [RFC6452]. This document describes a case where the IETF should disallow a character that the various properties would otherwise treat as PVALID.

This document provides the "flagging for the IESG" specified by Section 5.1 of RFC 5892. As specified there, the change itself requires IETF review because it alters the rules of Section 2 of that document.

Readers of this document are expected to be familiar with Unicode terminology [Unicode62] and the IETF conventions for representing Unicode code points [RFC5137].

As a convenience to readers of RFC 5892 and to reduce the risks of confusion, this document also formally applies the content of an erratum to the text of the RFC (see Section 3) and so brings that RFC up to date with all agreed changes.

> [[RFC Editor: please remove the following comment and note if they get to you.]]
>
> [[IESG: It might not be a bad idea to incorporate some version of the following into the Last Call announcement.]]
>
> NOTE IN DRAFT to IETF Reviewers: The issues in this document, and particularly the extended discussion below of why this change to RFC 5892 is necessary and appropriate, are fairly esoteric. Understanding them requires that one have at least some understanding of how the Arabic Script works and the reasons the Unicode Standard gives various Arabic Script characters a fairly extended discussion. It also requires understanding of a number of Unicode principles, including the Normalization Stability rules as applied to new precomposed characters and guidelines for adding new characters. References are provided for those who want to pursue them, but potential reviewers should assume that the

background needed to understand the reasons for this change is no less deep in the subject matter than would be expected of someone reviewing a proposed change in, e.g., the fundamentals of BGP, TCP congestion control, or some cryptographic algorithm.

2. Change to RFC 5892 for new character U+08A1

With the publication of this document, Section 2.6 ("Exceptions (F)") of RFC 5892 [RFC5892] is updated by adding 08A1 to the rule in Category F so that the rule itself reads:

```
F: cp is in {00B7, 00DF, 0375, 03C2, 05F3, 05F4, 0640, 0660,
             0661, 0662, 0663, 0664, 0665, 0666, 0667, 0668,
             0669, 06F0, 06F1, 06F2, 06F3, 06F4, 06F5, 06F6,
             06F7, 06F8, 06F9, 06FD, 06FE, 07FA, 08A1, 0F0B,
             3007, 302E, 302F, 3031, 3032, 3033, 3034, 3035,
             303B, 30FB}
```

and then add to the subtable designated
"DISALLOWED -- Would otherwise have been PVALID"
after the line that begins "07FA", the additional line:

```
   08A1; DISALLOWED # ARABIC LETTER BEH WITH HAMZA ABOVE
```

This has the effect of making the cited code point DISALLOWED independent of application of the rest of the IDNA rule set to the current version of Unicode. Those wishing to create domain name labels containing Beh with Hamza Above may continue to use the sequence

```
   U+0628, ARABIC LETTER BEH
```

followed by

```
   U+0654, ARABIC HAMZA ABOVE
```

which was valid for IDNA purposes in Unicode 5.0 and earlier and which continues to be valid.

3. Editorial clarification to RFC 5892

Verified RFC Editor Erratum 3312 [RFC5892Erratum] provides a clarification to Appendix A and Section A.1 of RFC 5892. This section of this document updates the RFC to apply that clarification.

1. In Appendix A, add a new paragraph after the paragraph that begins "The code point...". The new paragraph should read:

"For the rule to be evaluated to True for the label, it MUST be evaluated separately for every occurrence of the Code point in the

label; each of those evaluations must result in True."

2. In Appendix A, Section A.1, replace the "Rule Set" by

```
  Rule Set:
    False;
    If Canonical_Combining_Class(Before(cp)) .eq.  Virama Then True;
    If cp .eq. \u200C And
           RegExpMatch((Joining_Type:{L,D})(Joining_Type:T)*cp
      (Joining_Type:T)*(Joining_Type:{R,D})) Then True;
```

4. Explanation

[[NOTE IN DRAFT: Given the nature of this document, we believe this material belongs here. It could, however, be moved to an appendix if anyone felt strongly about that.]]

This section summarizes some of the discussions and reasoning that led to the conclusion and change in Section 2. It should not be considered as either normative or authoritative.

As the Unicode Standard points out at some length [Unicode62-Arabic], Hamza is a problematic abstract character and the "Hamza Above" construction even more so [Unicode62-Hamza]. Those sections explain a distinction made by Unicode between the use of a Hamza mark to denote a glottal stop and one used as a diacritic mark to denote a separate letter. In the first case, the combining sequence is used. In the second, a precombined character is assigned.

Unlike Unicode generally and because of concerns about identifier spoofing and attacks based on similarities, character distinctions in IDNA are based much more strictly on the appearance of characters; pronunciation distinctions are not considered. So, for IDNA, BEH WITH HAMZA ABOVE is not-quite-tautologically the same as BEH WITH HAMZA ABOVE, even if one of them is written as U+08A1 (new to Unicode 7.0.0) and the other as the sequence \u’0628’\u’0654’ (feasible with Unicode 7.0.0 but also available in versions of Unicode going back at least to the original publication of RFC 5892). Because the two are, for IDNA purposes, the same, IDNA expects that normalization (specifically the requirement that all U-labels be in NFC form) will cause them to compare equal.

If Unicode also considered them the same, then the principle would apply that new precomposed ("composition") forms are not added unless one of the code points that could be used to construct it did not exist in an earlier version (and even then is discouraged)[UAX15-Versioning]. When exceptions are made, they are expected to conform to the rules and classes in the "Composition Exclusion Table", with class 2 being relevant to this case [UAX15-Exclusion]. That rule essentially requires that the normalization for the old combining sequence to itself be retained (for stability) but that the newly-added character be treated as canonically decomposable and decompose back to the older sequence even under NFC. That was not done for this particular case, presumably because of the distinction about prounciation modifiers versus separate letters noted above. Because, for IDNA and the DNS, there is a possibility that the composing sequence \u'0628'\u'0654' already appears in labels, the only choice other than allowing an otherwise-identical, and identically-appearing, label with U+08A1 substituted to identify a different DNS entry is to DISALLOW the new character.

4.1. A related historical problem

At least three other grapheme clusters have been present for many version of Unicode and can be seen as involving issues similar to those for the newly-added ARABIC LETTER BEH WITH HAMZA ABOVE. ARABIC LETTER HAH WITH HAMZA ABOVE (U+0681) and ARABIC LETTER REH WITH HAMZA ABOVE (U+076C) do not have decomposition forms and are preferred over combining sequences using HAMZA ABOVE (U+0654) [Unicode62-Hamza]. By contrast, ARABIC LETTER ALEF WITH HAMZA ABOVE (U+0623) decomposes into \u'0627'\u'0653' and ARABIC LETTER YEH WITH HAMZA ABOVE (U+0626) decomposes into \u'064A'\u'0654' so the precomposed character and combining sequences compare equal when both are normalized, as this specification prefers.

There are other variations on this theme. For example, ARABIC LETTER U WITH HAMZA ABOVE (U+0677) has a compatibility decomposition into the combining sequence \u'06C7'\u'0674'.

Had the issues outlined in this document been better understood at the time, it probably would have been wise for RFC 5892 to disallow either the precomposed character or the combining sequence of each pair unless Unicode normalization rules cause the right thing to happen. Failure to do so at the time places an extra burden on

registries to be sure that conflicts (and the potential for confusion and attacks) do not exist. Oddly, had the exclusion been made part of the specification at that time, the preference noted above would probably have dictated excluding the combining sequence, something not otherwise done in IDNA2008. Today, the only thing that can be excluded without the potential disruption of disallowing a previously-PVALID combining sequence is the newly-added code point so whatever is done, or might have been contemplated with hindsight, it would be somewhat inconsistent.

4.2. How this is being done

Questions have arisen has to why this specification makes the change to RFC 5892 by DISALLOWing U+08A1 as a simple exception (IDNA Category F, RFC 5892 Section 2.7) rather than either a backward-compatibility case (IDNA Category G, RFC 5982 Section 2.8) or modifying IDNA Category F to make Hamza (or Hamza Above, or combining Hamza generally) into CONTEXTO cases and specifying appropriate limitations in a new entry in the IANA IDNA Context Registry (as specified in RFC 5892 Section 5.2). The subsections below explain why neither of those alternatives was chosen despite some discussion of each.

4.2.1. Backward compatibility and normalization

The "BackwardCompatible" category (IDNA Category G, RFC 5892 Section 5.3) is described as applying only when "property values in versions of Unicode after 5.2 have changed in such a way that the derived property value would no longer be PVALID or DISALLOWED". Because U+08A1 is a newly-added code point in Unicode 7.0.0 and no property values of code points in prior versions have changed, that category G does not apply. If that section of RFC 5892 is replaced in the future, perhaps consideration should be given to adding Normalization Stability and other issues to that description but, at present, it is not relevant.

4.2.2. A new contextual rule

As the Unicode Standard points out at some length [Unicode62-Arabic], Hamza is a problematic abstract character and the "Hamza Above" construction even more so. IDNA has historically associated characters whose use is reasonable in some contexts but not others with the special derived property "CONTEXTO" and then specified specific, context-dependent, rules about where they may be used. Because Hamza Above is problematic (and spawns edge cases, as discussed in the Unicode Standard section cited above), it was suggested that a contextual rule might be appropriate. There are at least two reasons why a contextual rule would not be suitable for the present situation.

1. As discussed above, the present situation is a normalization stability and predictability problem, not a contextual one. Had the same issues arisen with a newly-added precomposed character that could previously be constructed from non-problematic base and combining characters, it would be even more clearly a normalization issue and, following the principles discussed there and particularly in UAX 15 [UAX15-Exclusion], might not have been assigned at all.

2. The contextual rule sets are designed around restricting the use of code points to a particular script or adjacent to particular characters within that script. Neither of these cases applies to the newly-added character even if one could imagine rules for the use of Hamza Above (U+0654) that would reflect the considerations of Chapter 8 of Unicode 6.2. Even had the latter been desired, it would be somewhat late now -- Hamza Above has been present as a combining character (U+0654) in many versions of Unicode. While that section of the Unicode Standard describes the issues, it does not provide actionable guidance about what to do about it for cases going forward or when visual identity is important.

5. Acknowledgements

The Unicode 7.0.0 changes were extensively discussed within the IAB's Internationalization Program. The authors are grateful for the discussions and feedback there, especially from Andrew Sullivan and David Thaler. Additional information was requested and received from Mark Davis and Ken Whistler and while they probably do not agree with the necessity of excluding this code point as their responsibility is to look at the Unicode Consortium requirements for stability, the decision would not have been possible without their input. Several experts and reviewers who prefer to remain anonymous also provided helpful input and comments on preliminary versions of this document.

6. IANA Considerations

When the IANA registry and tables are updated to reflect Unicode 7.0.0, code point U+08A1 should be identified as DISALLOWED, consistent with the change made in Section 2.

7. Security Considerations

This specification excludes a code point for which the Unicode-specified normalization behavior could result in two ways to form a visually-identical character within the same script not comparing equal. That behavior could create a dream case for someone intending to confuse the user by use of a domain name that looked identical to another one, was entirely in the same script, but was still considered different (see, for example, the discussion of false negatives in identifier comparison in Section 2.1 of RFC 6943 [RFC6943]). This exclusion therefore should improve Internet security.

8. References

8.1. Normative References

[RFC5137] Klensin, J., "ASCII Escaping of Unicode Characters", BCP 137, RFC 5137, February 2008.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, August 2010.

[RFC5892Erratum] "RFC5892, "The Unicode Code Points and Internationalized Domain Names for Applications (IDNA)", August 2010, Errata ID: 3312", Errata ID 3312, August 2012, <http://www.rfc-editor.org/errata_search.php?rfc=5892>.

[RFC5892] Faltstrom, P., "The Unicode Code Points and Internationalized Domain Names for Applications (IDNA)", RFC 5892, August 2010.

[RFC6943] Thaler, D., "Issues in Identifier Comparison for Security Purposes", RFC 6943, May 2013.

[UAX15-Exclusion] Davis, M., Ed., "Unicode Standard Annex #15: Unicode Normalization Forms, Section 5", June 2014, <http://www.unicode.org/reports/tr15/#Primary_Exclusion_List_Table>.

[UAX15-Versioning] Davis, M., Ed., "Unicode Standard Annex #15: Unicode Normalization Forms, Section 3", June 2014, <http://www.unicode.org/reports/tr15/#Versioning>.

[Unicode62-Arabic] "The Unicode Standard, Version 6.2.0, ob.cit., Chapter 8", Chapter 8, 2012, <http://www.unicode.org/versions/Unicode6.2.0/ch08.pdf>.

Subsection titled "Encoding Principles", paragraph numbered 4, starting on page 251.

[Unicode62-Hamza] "The Unicode Standard, Version 6.2.0, ob.cit., Chapter 8", Chapter 8, 2012, <http://www.unicode.org/versions/Unicode6.2.0/ch08.pdf>.

Subsection titled "Combining Hamza Above" starting on page 263.

[Unicode62]

The Unicode Consortium, "The Unicode Standard, Version 6.2.0", ISBN 978-1-936213-07-8, 2012, <http://www.unicode.org/versions/Unicode6.2.0/>.

Preferred citation: The Unicode Consortium. The Unicode Standard, Version 6.2.0, (Mountain View, CA: The Unicode Consortium, 2012. ISBN 978-1-936213-07-8)

[Unicode7]
The Unicode Consortium, "The Unicode Standard, Version 7.0.0", ISBN 978-1-936213-09-2, 2014, <http://www.unicode.org/versions/Unicode7.0.0/>.

Preferred Citation: The Unicode Consortium. The Unicode Standard, Version 7.0.0, (Mountain View, CA: The Unicode Consortium, 2014. ISBN 978-1-936213-09-2)

8.2. Informative References

[RFC6452] Faltstrom, P. and P. Hoffman, "The Unicode Code Points and Internationalized Domain Names for Applications (IDNA) - Unicode 6.0", RFC 6452, November 2011.

Authors' Addresses

John C Klensin
1770 Massachusetts Ave, Ste 322
Cambridge, MA 02140
USA

Phone: +1 617 245 1457
Email: john-ietf@jck.com

Patrik Faltstrom
Netnod
Franzengatan 5
Stockholm, 112 51
Sweden

Phone: +46 70 6059051
Email: paf@netnod.se