Network Working Group
Internet-Draft
Obsoletes: 3490, 3491
(if approved)
Updates: 3492 (if approved)
Intended status: Standards Track
Expires: September 10, 2009

J. Klensin

March 9, 2009

Internationalized Domain Names in Applications (IDNA): Protocol
draft-ietf-idnabis-protocol-11.txt

Status of this Memo

Copyright Notice

Abstract

This document supplies the protocol definition for a revised and updated specification for internationalized domain names (IDNs). The rationale for these changes, the relationship to the older specification, and important terminology are provided in other documents. This document specifies the protocol mechanism, called Internationalizing Domain Names in Applications (IDNA), for registering and looking up IDNs in a way that does not require changes to the DNS itself. IDNA is only meant for processing domain names, not free text.

Table of Contents

1. Introduction

This document supplies the protocol definition for a revised and updated specification for internationalized domain names. Essential definitions and terminology for understanding this document and a road map of the collection of documents that make up IDNA2008 appear in [IDNA2008-Defs]. Appendix B discusses the relationship between this specification and the earlier version of IDNA (referred to here as "IDNA2003") and the rationale for these changes, along with considerable explanatory material and advice to zone administrators who support IDNs is provided in another documents, notably [IDNA2008-Rationale].

IDNA works by allowing applications to use certain ASCII string labels (beginning with a special prefix) to represent non-ASCII name labels. Lower-layer protocols need not be aware of this; therefore IDNA does not depend on changes to any infrastructure. In particular, IDNA does not depend on any changes to DNS servers, resolvers, or protocol elements, because the ASCII name service provided by the existing DNS is entirely sufficient for IDNA.

IDNA is applied only to DNS labels. Standards for combining labels into fully-qualified domain names and parsing labels out of those names are covered in the base DNS standards [RFC1034] [RFC1035] and their various updates. An application may, of course, apply locally-appropriate conventions to the presentation forms of domain names as discussed in [IDNA2008-Rationale].

While they share terminology, reference data, and some operations, this document describes two separate protocols, one for IDN registration (Section 4) and one for IDN lookup (Section 5).

1.1. Discussion Forum

[[anchor3: RFC Editor: please remove this section.]]

This work is being discussed in the IETF IDNABIS WG and on the mailing list idna-update@alvestrand.no

2. Terminology

General terminology applicable to IDNA, but with meanings familiar to those who have worked with Unicode or other character set standards and the DNS, appears in [IDNA2008-Defs]. Terminology that is an integral, normative, part of the IDNA definition, including the definitions of "ACE", appears in that document as well. Familiarity with the terminology materials in that document is assumed for

reading this one. The reader of this document is assumed to be familiar with DNS-specific terminology as defined in RFC 1034 [RFC1034].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119 [RFC2119].

3. Requirements and Applicability

3.1. Requirements

IDNA conformance means adherence to the following requirements:

1. Whenever a domain name is put into an IDN-unaware domain name slot (see Section 2 and [IDNA2008-Defs]), it MUST contain only ASCII characters (i.e., must be either an A-label or an NR-LDH-label), or must be a label associated with a DNS application that is not subject to either IDNA or the historical recommendations for "hostname"-style names [RFC1034].

2. Comparison of labels MUST be done on equivalent forms: either both A-Label forms or both U-Label forms. Because A-labels and U-labels can be transformed into each other without loss of information, these comparisons are equivalent. However, when a pair of putative A-labels are compared, the comparison MUST use an ASCII case-insensitive comparison (as with all comparisons of ASCII DNS labels). Comparisons on putative U-labels must test that the two strings are identical, without case-folding or other intermediate steps. Note that it is not necessary to verify that labels are valid in order to compare them. In many cases, verification of validity (that the strings actually are A-labels or U-labels) may be important for other reasons and SHOULD be performed.

3. Labels being registered MUST conform to the requirements of Section 4. Labels being looked up and the lookup process MUST conform to the requirements of Section 5.

3.2. Applicability

IDNA is applicable to all domain names in all domain name slots except where it is explicitly excluded. It is not applicable to domain name slots which do not use the LDH syntax rules.

This implies that IDNA is applicable to many protocols that predate

IDNA. Note that IDNs occupying domain name slots in those older protocols MUST be in A-label form until and unless those protocols and implementations of them are explicitly upgraded to be aware of IDNs in native character (Unicode, not encoded as A-labels) form. IDNs actually appearing in DNS queries or responses MUST be A-labels.

3.2.1. DNS Resource Records

IDNA applies only to domain names in the NAME and RDATA fields of DNS resource records whose CLASS is IN.

There are currently no other exclusions on the applicability of IDNA to DNS resource records. Applicability depends entirely on the CLASS, and not on the TYPE except as noted below. This will remain true, even as new types are defined, unless there is a compelling reason for a new type that requires type-specific rules. The special naming conventions applicable to SRV records (and "underscore names" more generally) are examples of type-specific rules that are incompatible with IDNA coding. Hence the first two labels (the ones required to start in "_") on a record with TYPE SRV MUST NOT be A-labels or U-labels (while it would be possible to write a non-ASCII string with a leading underscore, conversion to an A-label would be impossible without loss of information because the underscore is not a letter, digit, or hyphen and is consequently DISALLOWED in IDNs). Of course, those labels may be part of a domain that uses IDN labels at higher levels in the tree.

3.2.2. Non-domain-name Data Types Stored in the DNS

Although IDNA enables the representation of non-ASCII characters in domain names, that does not imply that IDNA enables the representation of non-ASCII characters in other data types that are stored in domain names, specifically in the RDATA field for types that have structured RDATA format. For example, an email address local part is stored in a domain name in the RNAME field as part of the RDATA of an SOA record (hostmaster@example.com would be represented as hostmaster.example.com). IDNA specifically does not update the existing email standards, which allow only ASCII characters in local parts. Even though work is in progress to define internationalization for email addresses [RFC4952], changes to the email address part of the SOA RDATA would require action in, or updates to, other standards, specifically those that specify the format of the SOA RR.

4. Registration Protocol

This section defines the procedure for registering an IDN. The

procedure is implementation independent; any sequence of steps that produces exactly the same result for all labels is considered a valid implementation.

Note that, while the registration and lookup protocols (Section 5) are very similar in most respects, they are different and implementers should carefully follow the steps they are implementing.

4.1. Input to IDNA Registration Process

[[anchor7: Note in Draft: This subsection is new in -09/, based on comments on the mailing list in January and February 2009. It replaces the previous first two subsections of this section and completely eliminates the discussion of local mapping for registration.]]

Registration processes are outside the scope of these protocols and may differ significantly depending on local needs. By the time a string enters the IDNA registration process as described in this specification, it is expected to be in Unicode and MUST be in Unicode Normalization Form C (NFC [Unicode-UAX15]). Entities responsible for zone files ("registries") are expected to accept only the exact string for which registration is requested, free of any mappings or local adjustments. They SHOULD avoid any possible ambiguity by accepting registrations only for A-labels, possibly paired with the relevant U-labels so that they can verify the correspondence.

4.2. Permitted Character and Label Validation

4.2.1. Input Format

The registry MAY permit submission of labels in A-label form and is encouraged to accept both the A-label form and the U-label one. If it does so, it MUST perform a conversion to a U-label, perform the steps and tests described below, and verify that the A-label produced by the step in Section 4.4 matches the one provided as input. In addition, if a U-label was provided, that U-label and the one obtained by conversion of the A-label MUST match exactly. If, for some reason, these tests fail, the registration MUST be rejected. If the conversion to a U-label is not performed, the registry MUST still verify that the A-label is superficially valid, i.e., that it does not violate any of the rules of Punycode [RFC3492] encoding such as the prohibition on trailing hyphen-minus, appearance of non-basic characters before the delimiter, and so on. Fake A-labels, i.e., invalid strings that appear to be A-labels but are not, MUST NOT be placed in DNS zones that support IDNA.

4.2.2. Rejection of Characters that are not Permitted

The candidate Unicode string is checked to verify that characters that IDNA does not permit do not appear in it. Those characters are identified in the "DISALLOWED" and "UNASSIGNED" lists that are specified in [IDNA2008-Tables] and described informally in [IDNA2008-Rationale]. Characters that are either DISALLOWED or UNASSIGNED MUST NOT be part of labels to be processed for registration in the DNS.

4.2.3. Label Validation

The proposed label (in the form of a Unicode string, i.e., a string that at least superficially appears to be a U-label) is then examined, performing tests that require examination of more than one character.

4.2.3.1. Rejection of Hyphen Sequences in U-labels

The Unicode string MUST NOT contain "--" (two consecutive hyphens) in the third and fourth character positions when the label is considered in "on the wire" order.

4.2.3.2. Leading Combining Marks

The first character of the string (when the label is considered in "on the wire" order) is examined to verify that it is not a combining mark (or combining character) (see The Unicode Standard, Section 2.11 [Unicode] for an exact definition). If it is a combining mark, the string MUST NOT be registered.

4.2.3.3. Contextual Rules

Each code point is checked for its identification as a character requiring contextual processing for registration (the list of characters appears as the combination of CONTEXTJ and CONTEXTO in [IDNA2008-Tables] as do the contextual rules themselves). If that indication appears, the table of contextual rules is checked for a rule for that character. If no rule is found, the proposed label is rejected and MUST NOT be installed in a zone file. If one is found, it is applied (typically as a test on the entire label or on adjacent characters within the label). If the application of the rule does not conclude that the character is valid in context, the proposed label MUST BE rejected. (See the IANA Considerations: IDNA Context Registry section of [IDNA2008-Tables].)

These contextual rules are required to support the use of characters that could be used, under other conditions, to produce misleading

labels or to cause unacceptable ambiguity in label matching and interpretation. For example, labels containing invisible ("zero-width") characters may be permitted in context with characters whose presentation forms are significantly changed by the presence or absence of the zero-width characters, while other labels in which zero-width characters appear may be rejected.

4.2.3.4. Labels Containing Characters Written Right to Left

Special tests are required for strings containing characters that are normally written from right to left. The criteria for classifying characters in terms of directionality are identified in the "Bidi" document [IDNA2008-BIDI] in this series. That document also describes conditions for strings that contain one or more of those characters to be U-labels. The tests for those conditions, specified there, are applied. Strings that contain right to left characters but that do not conform to the IDNA Bidi rules MUST NOT be inserted as labels in zone files.

4.2.4. Registration Validation Summary

Strings that contain at least one non-ASCII character, have been produced by the steps above, whose contents pass all of the tests in Section 4.2, and are 63 or fewer characters long in ACE form (see Section 4.4), are U-labels.

To summarize, tests are made in Section 4.2 for invalid characters, invalid combinations of characters, for labels that are invalid even if the characters they contain are valid individually, and for labels that do not conform to the restrictions for strings containing right to left characters.

4.3. Registry Restrictions

Registries at all levels of the DNS, not just the top level, are expected to establish policies about the labels that may be registered, and for the processes associated with that action. While exact policies are not specified as part of IDNA2008 and it is expected that different registries may specify different policies, there SHOULD be policies. Even a trivial policy (e.g., "anything can be registered in this zone that can be represented as an A-label - U-label pair") has value because it provides notice to users and applications implementers that the registry cannot be relied upon to provide even minimal user-protection restrictions. These per-registry policies and restrictions are an essential element of the IDNA registration protocol even for registries (and corresponding zone files) deep in the DNS hierarchy. As discussed in [IDNA2008-Rationale], such restrictions have always existed in the

DNS. That document also contains a discussion and recommendations about possible types of rules.

The string produced by the above steps is checked and processed as appropriate to local registry restrictions. Application of those registry restrictions may result in the rejection of some labels or the application of special restrictions to others.

4.4. Punycode Conversion

The resulting U-label is converted to an A-label. The A-label, more precisely defined elsewhere, is the encoding of the U-label according to the Punycode algorithm [RFC3492] with the ACE prefix "xn--" added at the beginning of the string. The resulting string must, of course, conform to the length limits imposed by the DNS. This document updates RFC 3492 only to the extent of replacing the reference to the discussion of the ACE prefix. The ACE prefix is now specified in this document rather than as part of RFC 3490 or Nameprep [RFC3491] but is the same in both sets of documents.

The failure conditions identified in the Punycode encoding procedure cannot occur if the input is a U-label as determined by the steps above.

4.5. Insertion in the Zone

The A-label is registered in the DNS by insertion into a zone.

5. Domain Name Lookup Protocol

Lookup is conceptually different from registration and different tests are applied on the client. Although some validity checks are necessary to avoid serious problems with the protocol (see Section 5.5ff.), the lookup-side tests are more permissive and rely on the assumption that names that are present in the DNS are valid. That assumption is, however, a weak one because the presence of wild cards in the DNS might cause a string that is not actually registered in the DNS to be successfully looked up.

[[anchor13: Note in Draft: Try to reorganize and renumber Section 5 (Lookup) so that it exactly parallels Section 4 (Registration). This has not been done in drafts -10 or -11 because the task will be much easier if the local mapping material is pulled from here (and there is no point trying to align the section numbers twice).]]

5.1. Label String Input

The user supplies a string in the local character set, typically by typing it or clicking on, or copying and pasting, a resource identifier, e.g., a URI [RFC3986] or IRI [RFC3987] from which the domain name is extracted. Alternately, some process not directly involving the user may read the string from a file or obtain it in some other way. Processing in this step and the next two are local matters, to be accomplished prior to actual invocation of IDNA, but at least the two steps in Section 5.2 and Section 5.3 must be accomplished in some way.

5.2. Conversion to Unicode

The string is converted from the local character set into Unicode, if it is not already Unicode. The exact nature of this conversion is beyond the scope of this document, but may involve normalization identical to that discussed in Section 4.1. The result MUST be a Unicode string in NFC form.

5.3. Character Changes in Preprocessing or the User Interface

[[anchor14: Note in Drafts -10 and -11. As of the time this draft was posted, the WG was continuing to discuss various alternatives to this section, which was pragmatic relative to various options and behavior but that seems to make no one happy from a predictability or transition standpoint. Please see the (temporary) first appendix to this document for a first cut at possible alternate formulations.]]

The Unicode string MAY then be processed to prevent confounding of user expectations. For instance, it might be reasonable, at this step, to convert all upper case characters to lower case, if this makes sense in the user's environment, but even this should be approached with caution due to some edge cases: in the long term, it is probably better for users to understand IDNs strictly in lower-case, U-label, form. More generally, preprocessing may be useful to smooth the transition from IDNA2003, especially for direct user input, but with similar cautions. In general, IDNs appearing in files and those transmitted across the network as part of protocols are expected to be in either ASCII form (including A-labels) or to contain U-labels, rather than being in forms requiring mapping or other conversions.

Other examples of processing for localization might be applied, especially to direct user input, at this point. They include interpreting various characters as separating domain name components from each other (label separators) because they either look like periods or are used to separate sentences, mapping halfwidth or

fullwidth East Asian characters to the common form permitted in labels, or giving special treatment to characters whose presentation forms are dependent only on placement in the label. Such localization changes are also outside the scope of this specification.

Recommendations for preprocessing for global contexts (i.e., when local considerations do not apply or cannot be used) and for maximum interoperability with labels that might have been specified under liberal readings of IDNA2003 are given in [IDNA2008-Rationale]. It is important to note that the intent of these specifications is that labels in application protocols, files, or links are intended to be in U-label or A-label form. Preprocessing MUST NOT map a character that is valid in a label as specified elsewhere in this document or in [IDNA2008-Tables] into another character. Excessively liberal use of preprocessing, especially to strings stored in files, poses a threat to consistent and predictable behavior for the user even if not to actual interoperability.

Because these transformations are local, it is important that domain names that might be passed between systems (e.g., in IRIs) be U-labels or A-labels and not forms that might be accepted locally as a consequence of this step. This step is not standardized as part of IDNA, and is not further specified here.

5.4. A-label Input

If the input to this procedure appears to be an A-label (i.e., it starts in "xn--"), the lookup application MAY attempt to convert it to a U-label and apply the tests of Section 5.5 and the conversion of Section 5.6 to that form. If the label is converted to Unicode (i.e., to U-label form) using the Punycode decoding algorithm, then the processing specified in those two sections MUST be performed, and the label MUST be rejected if the resulting label is not identical to the original. See the Name Server Considerations section of [IDNA2008-Rationale] for additional discussion on this topic.

That conversion and testing SHOULD be performed if the domain name will later be presented to the user in native character form (this requires that the lookup application be IDNA-aware). If those steps are not performed, the lookup process SHOULD at least make tests to determine that the string is actually an A-label, examining it for the invalid formats specified in the Punycode decoding specification. Applications that are not IDNA-aware will obviously omit that testing; others MAY treat the string as opaque to avoid the additional processing at the expense of providing less protection and information to users.

5.5. Validation and Character List Testing

As with the registration procedure described in Section 4, the Unicode string is checked to verify that all characters that appear in it are valid as input to IDNA lookup processing. As discussed above and in [IDNA2008-Rationale], the lookup check is more liberal than the registration one. Putative labels with any of the following characteristics MUST BE rejected prior to DNS lookup:

- Labels containing code points that are unassigned in the version of Unicode being used by the application, i.e.,in the UNASSIGNED category of [IDNA2008-Tables].

- Labels that are not in NFC form as defined in [Unicode-UAX15].

- Labels containing prohibited code points, i.e., those that are assigned to the "DISALLOWED" category in the permitted character table [IDNA2008-Tables].

- Labels containing code points that are identified in [IDNA2008-Tables] as "CONTEXTJ", i.e., requiring exceptional contextual rule processing on lookup, but that do not conform to that rule. Note that this implies that a rule much be defined, not null: a character that requires a contextual rule but for which the rule is null is treated in this step as having failed to conform to the rule.

- Labels containing code points that are identified in [IDNA2008-Tables] as "CONTEXTO", but for which no such rule appears in the table of rules. Applications resolving DNS names or carrying out equivalent operations are not required to test contextual rules for "CONTEXTO" characters, only to verify that a rule is defined (although they MAY make such tests to give better information to the user).

- Labels whose first character is a combining mark (see Section 4.2.3.2.

In addition, the application SHOULD apply the following test. The test may be omitted in special circumstances, such as when the lookup application knows that the conditions are enforced elsewhere, because an attempt to look up and resolve such strings will almost certainly lead to a DNS lookup failure except when wildcards are present in the zone. However, applying the test is likely to give much better information about the reason for a lookup failure -- information that may be usefully passed to the user when that is feasible -- than DNS resolution failure information alone. In any event, lookup applications should avoid attempting to resolve labels that are

invalid under that test.

- Verification that the string is compliant with the requirements for right to left characters, specified in [IDNA2008-BIDI].

For all other strings, the lookup application MUST rely on the presence or absence of labels in the DNS to determine the validity of those labels and the validity of the characters they contain. If they are registered, they are presumed to be valid; if they are not, their possible validity is not relevant. A lookup application that declines to process a string that conforms to the rules above and does not look it up in the DNS is not in conformance with this protocol.

5.6. Punycode Conversion

The string that has now been validated for lookup is converted to ACE form using the Punycode algorithm (with the ACE prefix added). With the understanding that this summary is not normative (the steps above are), the string has either

- been determined to be in Unicode and in NFC form with no leading combining marks, to contain no DISALLOWED or UNASSIGNED code points, to have rules associated with any code points in CONTEXTJ or CONTEXTO, and, for those in CONTEXTJ, to satisfy the conditions of the rules.

- satisfied the conditions for A-label input in Section 5.4 under circumstances in which the U-label conversions and tests have not been performed

5.7. DNS Name Resolution

That resulting validated string is looked up in the DNS, using normal DNS resolver procedures. That lookup can obviously either succeed (returning information) or fail.

6. Security Considerations

Security Considerations for this version of IDNA, except for the special issues associated with right to left scripts and characters, are described in [IDNA2008-Defs]. Specific issues for labels containing characters associated with scripts written right to left appear in [IDNA2008-BIDI].

7. IANA Considerations

IANA actions for this version of IDNA are specified in [IDNA2008-Tables] and discussed informally in [IDNA2008-Rationale]. The components of IDNA described in this document do not require any IANA actions.

8. Contributors

While the listed editor held the pen, the original versions of this document represent the joint work and conclusions of an ad hoc design team consisting of the editor and, in alphabetic order, Harald Alvestrand, Tina Dam, Patrik Faltstrom, and Cary Karp. This document draws significantly on the original version of IDNA [RFC3490] both conceptually and for specific text. This second-generation version would not have been possible without the work that went into that first version and its authors, Patrik Faltstrom, Paul Hoffman, and Adam Costello. While Faltstrom was actively involved in the creation of this version, Hoffman and Costello were not and should not be held responsible for any errors or omissions.

9. Acknowledgments

This revision to IDNA would have been impossible without the accumulated experience since RFC 3490 was published and resulting comments and complaints of many people in the IETF, ICANN, and other communities, too many people to list here. Nor would it have been possible without RFC 3490 itself and the efforts of the Working Group that defined it. Those people whose contributions are acknowledged in RFC 3490, [RFC4690], and [IDNA2008-Rationale] were particularly important.

Specific textual changes were incorporated into this document after suggestions from the other contributors, Stephane Bortzmeyer, Vint Cerf, Mark Davis, Paul Hoffman, Kent Karlsson, Erik van der Poel, Marcos Sanz, Andrew Sullivan, Ken Whistler, and other WG participants. Special thanks are due to Paul Hoffman for permission to extract material from his Internet-Draft to form the basis for Appendix B.

10. References

10.1. Normative References

[IDNA2008-BIDI]
Alvestrand, H. and C. Karp, "An updated IDNA criterion for right-to-left scripts", July 2008, <https://datatracker.ietf.org/drafts/draft-ietf-idnabis-bidi/>.

[IDNA2008-Defs]
Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", February 2009, <https://datatracker.ietf.org/drafts/draft-ietf-idnabis-defs/>.

[IDNA2008-Tables]
Faltstrom, P., "The Unicode Codepoints and IDNA", July 2008, <https://datatracker.ietf.org/drafts/draft-ietf-idnabis-tables/>.

A version of this document is available in HTML format at http://stupid.domain.name/idnabis/draft-ietf-idnabis-tables-02.html

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC1123] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3492] Costello, A., "Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)", RFC 3492, March 2003.

[Unicode-PropertyValueAliases]
The Unicode Consortium, "Unicode Character Database: PropertyValueAliases", March 2008, <http://www.unicode.org/Public/UNIDATA/PropertyValueAliases.txt>.

[Unicode-RegEx]
The Unicode Consortium, "Unicode Technical Standard #18: Unicode Regular Expressions", May 2005, <http://www.unicode.org/reports/tr18/>.

[Unicode-Scripts]
The Unicode Consortium, "Unicode Standard Annex #24: Unicode Script Property", February 2008, <http://www.unicode.org/reports/tr24/>.

[Unicode-UAX15]
The Unicode Consortium, "Unicode Standard Annex #15: Unicode Normalization Forms", 2006, <http://www.unicode.org/reports/tr15/>.

10.2. Informative References

[ASCII] American National Standards Institute (formerly United States of America Standards Institute), "USA Code for Information Interchange", ANSI X3.4-1968, 1968.

ANSI X3.4-1968 has been replaced by newer versions with slight modifications, but the 1968 version remains definitive for the Internet.

[IDNA2008-Rationale]
Klensin, J., Ed., "Internationalizing Domain Names for Applications (IDNA): Issues, Explanation, and Rationale", February 2009, <https://datatracker.ietf.org/drafts/draft-ietf-idnabis-rationale>.

[RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.

[RFC2535] Eastlake, D., "Domain Name System Security Extensions", RFC 2535, March 1999.

[RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, August 1999.

[RFC3490] Faltstrom, P., Hoffman, P., and A. Costello, "Internationalizing Domain Names in Applications (IDNA)", RFC 3490, March 2003.

[RFC3491] Hoffman, P. and M. Blanchet, "Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN)", RFC 3491, March 2003.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform

Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, January 2005.

[RFC4690] Klensin, J., Faltstrom, P., Karp, C., and IAB, "Review and Recommendations for Internationalized Domain Names (IDNs)", RFC 4690, September 2006.

[RFC4952] Klensin, J. and Y. Ko, "Overview and Framework for Internationalized Email", RFC 4952, July 2007.

[Unicode] The Unicode Consortium, "The Unicode Standard, Version 5.0", 2007.

Boston, MA, USA: Addison-Wesley. ISBN 0-321-48091-0

Appendix A. Local Mapping Alternatives

The subsections of this appendix are temporary and represent different sketches of possible replacements for Section 5.3. They do not represent an assertion of WG consensus or any assertion about the possibility of including one of them as part of the WG's work program. Instead, they are supplied only for purposes of comparison, discussion, and, should it be relevant, refinement.

The first paragraph of each subsection describes how the material would be placed relative to the existing main document text. Subsequent paragraphs are the actual suggestions, although in incomplete sketch form.

A.1. Transitional Mapping Model

If this subsection were adopted, Section 5.3 would be deleted and this one would be inserted after, or integrated with, Section 5.7.

This specification does not support the extensive mappings from one character to another, including Unicode Case Folding and Compatibility Character mapping, of IDNA2003. It also changes the interpretations of a small number of characters relative to IDNA2003. Most applications, especially those with which IDNs have been used for some time, will need to maintain reasonable compatibility with files created under IDNA2003 and user interfaces designed for it. This section specifies additional steps to be taken to provide maximum IDNA2003 compatibility.

If an application requires IDNA2003 backward compatibility, it MUST execute the steps in one of the two subsections that immediately follow.

A.1.1. Fallback Lookup

If the string validates and the resolution attempt in Section 5.7 successfully returns a result, the lookup process terminates with that result. If validation succeeds but resolution fails, the validated string is proceeded through the ToASCII operation specified in IDNA2003 [RFC3490]. Assuming it produces a valid result, the resulting string is compared to the previous validated one. If they are not identical, a resolution attempt is made with the ToASCII output and the result of that attempt is returned as the result of the lookup operation.

Should IDNA2008 validation fail, the string is processed through ToASCII and, assuming the result is valid, the resulting string is resolved and the result of that attempt returned as the result of the lookup operation.

If ToASCII (IDNA2003) conversion is attempted and fails, the lookup operation behaves as if no name was found in the DNS.

Note that this procedure involves, at most, one DNS lookup (resolution attempt). If IDNA2008 string validation, conversion, and resolution succeed, no attempt is made to use IDNA2003 mechanisms. The procedure does, however, require that lookup applications fully support both IDNA2008 and IDNA2003 lookup operations so that the fallback can occur.

A.1.2. Two-step Lookup

Prior to the resolution attempt in Section 5.7, ACE strings are computed using both IDNA2003 (ToASCII) and IDNA2008 methods (as specified here). Assuming both validate, those strings are compared. If they are identical, or only one was valid, then a single DNS resolution is performed and its result is the result of the lookup operation. If both are valid but they are not identical, one resolution attempt is made with each of the two ACE strings.

If neither string is valid as an IDN, then the lookup operation fails.

When two resolutions are attempted, if one of the two is successful and the other is not, the successful value is used as the result of the lookup. If both are successful, the user or calling application must be presented with a choice in some way.

This procedure will require two DNS lookups (resolution attempts) in all cases except those in which the label string fails IDNA2008 validation, neither IDNA2003 or IDNA2008 can validate the string and translate it to ACE form, or the strings obtained from the two conversions are identical. As with the prior option, IDNA implementations will need to support both the IDNA2003 algorithm and tables and the IDNA2008 one. The question of how multiple results from different interpretations of the same input string should be handled by applications is a difficult one, with potential false positive and security attack vector implications as well as the possibility of general confusion.

In particular, if both interpretations of the name return values, the lookup application has no practical way to tell whether the relevant registry has applied "variant" or "bundling" techniques to ensure that both domain names are under the same control or not. From that perspective, the approach in the previous subsection assumes that has been done (if the IDNA2003-interpretation label is present at all) while this one assumes that such bundling is unlikely to have occurred.

[[anchor24: Note in Draft: If this appendix is used, RFC3490 must be moved from Informative to Normative.]]

A.2. Internationalized Resource Identifier (IRI) Mapping Model

This subsection is intended to be descriptive of an approach that lies outside IDNA, rather than a normative component of it. If it were adopted, Section 5.3 would be deleted and the material below would be referenced, either as a non-normative Appendix in Protocol or, more reasonably, as a section of Rationale.

IDNA2003 supported extensive mappings from one character to another, including Unicode Case Folding and Compatibility Character mapping. Those mappings are no longer supported on registration and are inconsistent with the "exact match" lookups that people expect from the DNS. Some mapping should still be supported, both for compatibility with applications that assume IDNA2003 and to avoid confounding user expectations. The specific mappings involved are not part of IDNA, but are expected to be specified as part of a revision to the IRI specification [RFC3987] and the conversion from IRI form to URI form. That change leaves mapping unspecified and prohibited for actual domain names, however, in practice, most domain names, especially in the web applications that appear to have been most important for IDNs between the publication of IDNA2003 and the release of this specification, are not interpreted as themselves but as abbreviated form of URIs or IRIs and hence subject to the transformation rules of the latter.

Appendix B. Summary of Major Changes from IDNA2003

1. Update base character set from Unicode 3.2 to Unicode version-agnostic.

2. Separate the definitions for the "registration" and "lookup" activities.

3. Disallow symbol and punctuation characters except where special exceptions are necessary.

4. Remove the mapping and normalization steps from the protocol and have them instead done by the applications themselves, possibly in a local fashion, before invoking the protocol.

5. Change the way that the protocol specifies which characters are allowed in labels from "humans decide what the table of codepoints contains" to "decision about codepoints are based on Unicode properties plus a small exclusion list created by humans".

6. Introduce the new concept of characters that can be used only in specific contexts.

7. Allow typical words and names in languages such as Dhivehi and Yiddish to be expressed.

8. Make bidirectional domain names (delimited strings of labels, not just labels standing on their own) display in a less surprising fashion whether they appear in obvious domain name contexts or as part of running text in paragraphs.

9. Remove the dot separator from the mandatory part of the protocol.

10. Make some currently-valid labels that are not actually IDNA labels invalid.

Appendix C. Change Log

[[anchor27: RFC Editor: Please remove this appendix.]]

C.1. Changes between Version -00 and -01 of draft-ietf-idnabis-protocol

- Corrected discussion of SRV records.

- Several small corrections for clarity.
- Inserted more "open issue" placeholders.

C.2. Version -02

- Rewrote the "conversion to Unicode" text in Section 5.2 as requested on-list.
- Added a comment (and reference) about EDNS0 to the "DNS Server Conventions" section, which was also retitled.
- Made several editorial corrections and improvements in response to various comments.
- Added several new discussion placeholder anchors and updated some older ones.

C.3. Version -03

- Trimmed change log, removing information about pre-WG drafts.
- Incorporated a number of changes suggested by Marcos Sanz in his note of 2008.07.17 and added several more placeholder anchors.
- Several minor editorial corrections and improvements.
- "Editor" designation temporarily removed because the automatic posting machinery does not accept it.

C.4. Version -04

- Removed Contextual Rule appendices for transfer to Tables.
- Several changes, including removal of discussion anchors, based on discussions at IETF 72 (Dublin)
- Rewrote the preprocessing material (Section 5.3) somewhat.

C.5. Version -05

- Updated part of the A-label input explanation (Section 5.4) per note from Erik van der Poel.

C.6. Version -06

- Corrected a few typographical errors.
- Incorporated the material (formerly in Rationale) on the relationship between IDNA2003 and IDNA2008 as an appendix and pointed to the new definitions document.
- Text modified in several places to recognize the dangers of interaction between DNS wildcards and IDNs.
- Text added to be explicit about the handling of edge and failure cases in Punycode encoding and decoding.
- Revised for consistency with the new Definitions document and to make the text read more smoothly.

C.7. Version -07

- Multiple small textual and editorial changes and clarifications.
- Requirement for normalization clarified to apply to all cases and conditions for preprocessing further clarified.
- Substantive change to Section 4.2.1, turning a SHOULD to a MUST (see note from Mark Davis, 19 November, 2008 18:14 -0800).

C.8. Version -08

- Added some references and altered text to improve clarity.
- Changed the description of CONTEXTJ/CONTEXTO to conform to that in Tables. In other words, these are now treated as distinction categories (again), rather than as specially-flagged subsets of PROTOCOL VALID.
- The discussion of label comparisons has been rewritten to make it more precise and to clarify that one does not need to verify that a string is a [valid] A-label or U-label in order to test it for equality with another string. The WG should verify that the current text is what is desired.
- Other changes to reflect post-IETF discussions or editorial improvements.

C.9. Version -09

- Removed Security Considerations material to Defs document.
- Removed the Name Server Considerations material to Rationale. That material is not normative and not needed to implement the protocol itself.
- Adjusted terminology to match new version of Defs.
- Removed all discussion of local mapping and option for it from registration protocol.
- Removed some old placeholders and inquiries because no comments have been received.
- Small editorial corrections.

C.10. Version -10

- Rewrote the registration input material slightly to further clarify the "no mapping on registration" principle.
- Added placeholder notes about several tasks, notably reorganizing Section 4 and Section 5 so that subsection numbers are parallel.
- Cleaned up an incorrect use of the terms "A-label" and "U-label" in the lookup phase that was spotted by Mark Davis. Inserted a note there about alternate ways to deal with the resulting terminology problem.
- Added a temporarily appendix (above) to document alternate strategies for possible replacements for Section 5.3.

C.11. Version -11

- Removed dangling reference to "C-label" (editing error in prior draft).
- Recast the last steps of the Lookup description to eliminate "apparent" (previously "putative") terminology.
- Rewrote major portions of the temporary appendix that describes transitional mappings to improve clarity and add context.
- Did some fine-tuning of terminology, notably in Section 3.2.1.

Author's Address

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

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