Internationalizing Domain Names in Applications (IDNA): Protocol
draft-klensin-idnabis-protocol-01.txt

Status of this Memo

By submitting this Internet-Draft, each author represents that any
applicable patent or other IPR claims of which he or she is aware
have been or will be disclosed, and any of which he or she becomes
aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering
Task Force (IETF), its areas, and its working groups. Note that
other groups may also distribute working documents as Internet-
Drafts.

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."

The list of current Internet-Drafts can be accessed at
http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at

This Internet-Draft will expire on May 20, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2007).

Abstract

This document supplies the protocol definition for a revised and
updated specification for internationalized domain names. The
rationale for these changes and relationship to the older
specification and some new terminology is provided in other
documents. This document specifies a standard method using
characters outside the ASCII repertoire in domain names. This
document defines internationalized domain names (IDNs) and a
mechanism called Internationalizing Domain Names in Applications (IDNA) for handling them in a standard fashion. IDNs use characters drawn from a large subset of the Unicode repertoire, but IDNA allows the non-ASCII characters to be represented using only the ASCII characters already allowed in so-called host names today. This backward-compatible representation is required in existing protocols like DNS, so that IDNs can be introduced with no changes to the existing infrastructure. IDNA is only meant for processing domain names, not free text.
Table of Contents

1. Introduction .................................................. 4
   1.1. Discussion Forum .......................................... 4
2. Terminology .................................................... 4
3. Requirements and Applicability ............................... 5
   3.1. Requirements .............................................. 5
   3.2. Applicability ............................................. 5
      3.2.1. DNS Resource Records ................................ 5
      3.2.2. Non-domain-name Data Types Stored in the DNS ........ 6
4. Registration Protocol ......................................... 6
   4.1. Proposed label ............................................ 6
   4.2. Conversion to Unicode ..................................... 6
   4.3. Permitted Character Identification ........................ 6
   4.4. Additional Character String Checking and Processing ... 7
   4.5. Registry Restrictions ..................................... 8
   4.6. Punycode Conversion ...................................... 8
   4.7. Insertion in the Zone ..................................... 8
5. Domain Name Resolution (Lookup) Protocol .................... 8
   5.1. User input ................................................. 8
   5.2. Conversion to Unicode ..................................... 8
   5.3. User Interface Character Changes ........................ 9
   5.4. Pre-Normalization Validation and Character List Testing . 9
   5.5. Normalization ............................................. 10
   5.6. Post-Normalization Processing ............................ 10
   5.7. Punycode Conversion ...................................... 10
   5.8. DNS Name Resolution ...................................... 10
6. Name server Considerations ................................... 10
   6.1. Processing Non-ASCII Strings .............................. 10
   6.2. DNSSEC Authentication of IDN Domain Names ............... 11
   6.3. Root Server Considerations ................................ 11
7. Security Considerations ........................................ 11
8. IANA Considerations ........................................... 12
9. Contributors ................................................... 12
10. Acknowledgements ............................................. 12
11. References ................................................... 13
    11.1. Normative References .................................... 13
    11.2. Informative References .................................. 13
Author’s Address .................................................. 14
Intellectual Property and Copyright Statements .................. 16
1. Introduction

This document supplies the protocol definition for a revised and updated specification for internationalized domain names. The rationale for these changes and relationship to the older specification and some new terminology is provided in other documents, notably [IDNA200X-issues].

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 [RFC1035]. An application may, of course, apply locally-appropriate conventions to the presentation forms of domain names as discussed in [IDNA200X-issues].

A good deal of the background material that appeared in RFC 3490 has been removed from this update. That material is either of historical interest only or has been covered from a more recent perspective in RFC 4690 [RFC4690] and [IDNA200X-issues].

1.1. Discussion Forum

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

This work is being discussed 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 [IDNA200X-issues]. 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.

[[anchor4: May want to copy some definitions from "issues" back here to avoid or reduce normative dependencies. But, of course, that...]]
would risk the two sets of definitions becoming inconsistent. Tradeoff...]

The key words "MUST", "SHALL", "REQUIRED", "SHOULD", "RECOMMENDED", and "MAY" 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 [IDNA200X-issues]), it MUST contain only ASCII characters (i.e., must be either an A-label, an LDH-label, or a label associated with a DNS application that is not subject to IDNA.

2. Comparing two labels MUST be done on the A-label form, using an ASCII case-insensitive comparison, as with all comparisons of DNS labels.

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 protocols MUST be in A-label form.

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; it depends entirely on the CLASS, and not on the TYPE. 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. It is worth noting that the special naming conventions applicable to SRV records are precisely such type-specific rules.
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 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. Other work is under development to define internationalization for email addresses [RFC4952], but changes to that part of the SOA RDATA would require action in other standards, which could also specify IDNA interpretation of labels that follow the local part such as by permitting them to be A-labels.

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.

4.1. Proposed label

The registrant submits a request for an IDN. The user typically produces the request string by the keyboard entry of a character sequence.

4.2. Conversion to Unicode

Some system routine, or a localized front-end to the IDNA process, ensures that the proposed label is a Unicode string. As a local implementation choice, the implementation may choose to map some forbidden characters to permitted characters (for instance mapping uppercase characters to lowercase ones), displaying the result to the user, and allowing processing to continue.

4.3. Permitted Character Identification

The Unicode string is examined to prohibit characters that IDNA does not permit in input. Those characters are identified in the "NEVER" list that is discussed in [IDNA200X-issues] which characters specified in [IDNA200X-Permitted]. Characters that are specified as "NEVER" in those documents, and characters or sequences that are unassigned in Unicode, MUST NOT be part of labels being processed for
registration in the DNS.

4.4.  Additional Character String Checking and Processing

All characters produced as output of the preceding step are then verified for permissibility by IDNA. Conceptually, these tests are, in order:

1.  Each code point is verified to be assigned in the version of Unicode in use (See [IDNA200X-issues]). If verification fails, the proposed label containing the code point is not a U-label and MUST NOT be processed further.

2.  Each code point is checked for its presence as "YES" (always permitted) in the table of included characters for registration or, if appropriate for the specific registry, as "MAYBE" permitted (see [IDNA200X-Permitted]).

3.  Each code point is checked for its presence as "CONTEXTUAL RULE REQUIRED" in the table of included characters for registration. 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. If one is found, it is applied (typically as a test on the entire label or adjacent characters). 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 [IDNA200X-issues]).

4.  Additional special tests for right-to-left strings are applied (See [IDNA200X-BIDI]).

 Strings that have been produced by the steps above, and whose contents pass the above tests, are U-labels.

To summarize, tests are made here for invalid characters, invalid combinations of characters, and for labels that are invalid even if the individual characters they contain are all valid. 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. Additional transformations that do not occur as the result of the steps above may be specified at this point by IDNA200x. As the list of characters permitted to be registered expands, new rules, similar to those suggested for zero-width characters, may accompany them.
4.5. 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. As discussed in [IDNA200X-issues]), such restrictions have always existed in the DNS.

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.

4.6. Punycode Conversion

The resulting U-label is converted to an A-label (i.e., the encoding of that label according to the Punycode algorithm [RFC3492] with the prefix included, i.e., the "xn--..." form).

4.7. Insertion in the Zone

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

5. Domain Name Resolution (Lookup) Protocol

Resolution is conceptually different from registration and different tests are applied on the client. The resolution-side tests are more permissive and rely heavily on the assumption that names that are present in the DNS are valid. Among other things, this distinction facilitates expansion of the permitted character lists to include new scripts and accommodate new versions of Unicode.

5.1. User 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. Processing in this step and the next two are local matters, to be accomplished prior to actual invocation of IDNAbis, but at least this one and the next one must be accomplished in some way.

5.2. Conversion to Unicode

The local character set, character coding conventions, and, as necessary, display and presentation conventions, are converted to Unicode (without surrogates), paralleling the process above.
5.3. User Interface Character Changes

The Unicode string MAY then be processed, in a way specific to the local environment, to make the result of the IDNA processing match user expectations. For instance, at this step, it would be reasonable to case-fold all upper case characters to lower case, if this makes sense in the user’s environment.

Other examples of processing for localization that might be applied, if appropriate, at this point (but even further outside the scope of this specification) include interpreting the KANA MIDDLE DOT to separate domain name components from each other, standardizing different "width" forms of the same character, or giving special treatment to characters whose presentation forms are dependent only on placement in the label.

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, and not specified further here.

5.4. Pre-Normalization Validation and Character List Testing

In parallel with the registration procedure, the Unicode string is checked to verify that all characters that appear in it are valid for IDNA resolution input. As discussed in [IDNA200X-issues], the resolution check is more liberal than that of the registration one. Characters that fall into the "MAYBE" (see [IDNA200X-issues]) categories in the inclusion tables do not lead to label rejection on resolution. Putative labels containing code points with any of the following characteristics MUST BE rejected prior to DNS lookup:

- Code points that are unassigned in the version of Unicode being used by the application, i.e., in the "Unassigned" Unicode category.

- Prohibited code points, i.e., those that are assigned to the "NEVER" category in the permitted character table.

- Code points that are shown in the permitted character table as requiring a contextual rule ("CONTEXTUAL RULE REQUIRED"), but for which no such rule appears in the table of rules.

For all other strings, the resolver 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.

5.5. Normalization

The validated Unicode string is normalized (using NFC); no case-mapping is performed.

5.6. Post-Normalization Processing

Any necessary processing or filtering is applied to the normalized output string produced by the above. In the cases that can be anticipated, this step will be null. It is included in the model in case, e.g., full-label checks are needed on lookup.

5.7. Punycode Conversion

The validated string, a U-label, is converted to an A-label.

5.8. DNS Name Resolution

The A-label is looked up in the DNS, using normal DNS procedures.

6. Name server Considerations

6.1. Processing Non-ASCII Strings

Existing DNS servers do not know the IDNA rules for handling non-ASCII forms of IDNs, and therefore need to be shielded from them. All existing channels through which names can enter a DNS server database (for example, master files (as described in RFC 1034) and DNS update messages [RFC2136]) are IDN-unaware because they predate IDNA. Other sections of this document provide the needed shielding by ensuring that internationalized domain names entering DNS server databases through such channels have already been converted to their equivalent ASCII A-label forms.

Because of the design of the algorithms in Section 4 and Section 5 (a domain name containing only ASCII codepoints can not be converted to an A-label), there can not be more than one label for each domain name.

RFC2821 explicitly allows domain labels to contain octets beyond the ASCII range (0..7F), and this document does not change that. Note, however, that there is no defined interpretation of octets 80..FF as characters. If labels containing these octets are returned to...
applications, unpredictable behavior could result. The A-label form, which cannot contain those characters, is the only standard representation for internationalized labels in the current DNS protocol.

6.2. DNSSEC Authentication of IDN Domain Names

DNS Security [RFC2535] is a method for supplying cryptographic verification information along with DNS messages. Public Key Cryptography is used in conjunction with digital signatures to provide a means for a requester of domain information to authenticate the source of the data. This ensures that it can be traced back to a trusted source, either directly, or via a chain of trust linking the source of the information to the top of the DNS hierarchy.

IDNA specifies that all internationalized domain names served by DNS servers that cannot be represented directly in ASCII must use the A-label form. Conversion to A-labels must be performed prior to a zone being signed by the private key for that zone. Because of this ordering, it is important to recognize that DNSSEC authenticates a domain name containing A-labels or conventional LDH-labels, not U-labels. In the presence of DNSSEC, no form of a zone file or query response that contains a U-label may be signed or validated against.

One consequence of this for sites deploying IDNA in the presence of DNSSEC is that any special purpose proxies or forwarders used to transform user input into IDNs must be earlier in the resolution flow than DNSSEC authenticating nameservers for DNSSEC to work.

6.3. Root Server Considerations

IDNs are likely to be somewhat longer than current domain names, so the bandwidth needed by the root servers is likely to go up by a small amount. Also, queries and responses for IDNs will probably be somewhat longer than typical queries today, so more queries and responses may be forced to go to TCP instead of UDP.

7. Security Considerations

The general security principles and issues for IDNA appear in [IDNA200X-issues]. The comments below are specific to this protocol, but should be read in the context of that material and the specifications, identified there, on which this one depends.

This memo describes an algorithm which encodes characters that are not valid according to the base DNS specifications (STD13 [RFC1034] [RFC1035] and Host Requirements [RFC1123]) into octet values that are
valid. No security issues such as string length increases or new allowed values are introduced by the encoding process or the use of these encoded values, apart from those introduced by the ACE encoding itself.

Domain names (or portions of them) are sometimes compared against a set of privileged or anti-privileged domains. In such situations it is especially important that the comparisons be done properly, as specified in requirement 2 of Section 3.1. For labels already in ASCII form, the proper comparison reduces to the same case-insensitive ASCII comparison that has always been used for ASCII labels.

The introduction of IDNA means that any existing labels that start with the ACE prefix would be construed as U-labels, at least until they failed one of the relevant tests, whether or not that was the intent of the zone administrator or registrant. There is no evidence that this has caused any practical problems since RFC 3490 was adopted, but the risk still exists in principle.

8. IANA Considerations

IANA actions for this version of IDNA are specified in [IDNA200X-issues].

9. Contributors

While the listed editor held the pen, this document represents 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.

10. Acknowledgements

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 [IDNA200X-issues] were particularly important.

11. References

11.1. Normative References

[IDNA200X-BIDI]

[IDNA200X-issues]


11.2. Informative References


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

[IDNA200X-Permitted]
draft-faltstrom-idnabis-tables-02.txt).

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

[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


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

Resource Identifier (URI): Generic Syntax", STD 66,

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.


5.0", 2007.

Boston, MA, USA: Addison-Wesley. ISBN 0-321-48091-0
Author’s Address

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

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